High-order Padé and Singly Diagonally Runge-Kutta schemes for linear ODEs, application to wave propagation problems

Hélène Barucq, Marc Duruflé, Mamadou N'Diaye

High-order Padé and Singly Diagonally Runge-Kutta schemes for linear ODEs, application to wave propagation problems

Hélène Barucq[1,3], Marc Duruflé[1,2] and Mamadou N'Diaye[1,3]

October 24, 2017

Abstract

In this paper we address the problem of constructing high-order implicit time schemes for wave equations. We consider two classes of one-step A-stable schemes adapted to linear Ordinary Differential Equation (ODE). The first class, which is not dissipative is based upon the diagonal Padé approximant of exponential function. For this class, the obtained schemes have the same stability function as Gauss Runge-Kutta (Gauss RK) schemes. They have the advantage to involve the solution of smaller linear systems at each time step compared to Gauss RK. The second class of schemes are constructed such that they require the inversion of a unique linear system several times at each time step like the Singly Diagonally Runge-Kutta (SDIRK) schemes. While the first class of schemes is constructed for an arbitrary order of accuracy, the second class schemes is given up to order 12. The performance assessment we provide shows a very good level of accuracy for both classes of schemes, and the great interest of considering high-order time schemes that are faster. The diagonal Padé schemes seem to be more accurate and more robust.

1 Introduction

The solution of wave propagation problems in electromagnetics, acoustics and elastodynamics has found important applications in many areas of engineering and science such as geophysics (seismic imaging), medicine (medical imaging), aerospace (radar), and telecommunication (antenna design, optical fibers). This wide range of applications has led to the development of many computational techniques for solving the partial differential equations (PDEs) governing wave propagation problems.

High-order finite element methods (FEM) have now demonstrated their strong capability for solving wave equations ([1], [2],...). In particular, they are well suited for considering complex geometries and heterogeneous media. The implementation of FEM requires to introduce an artificial boundary which is represented with an Absorbing Boundary Condition (ABC) or a Perfectly Matched Layers (PML). In practice, ABCs or PMLs are easier to handle when the wave equation is formulated as a first-order (in space and time) system as we consider herein (as in [1]). After space discretization, the obtained ODE can be discretized either with explicit or implicit time schemes. Explicit time schemes are very popular since they generate algorithms both cheap in memory and highly scalable. However, for stability purposes the time step is restricted by the size of the smallest element in the mesh and by the degree of the polynomials used in the FEM. As a consequence, even few small elements can make the maximal value of the time step (known as the Courant-Friedrichs-Lewy or CFL condition) so small that the computational cost becomes prohibitive.

A nice work has been done in [3] to increase the CFL, especially for high-order space approximations. The idea consists in applying a specific discretization in space in a way that the eigenvalues of the discrete matrix are modified leading to a maximal CFL number. Other works propose

[1]Inria Centre de Recherche Bordeaux Sud-Ouest, Team Magique-3D, 200 avenue de la vieille Tour, 33 405 Talence, FRANCE. email: helene.barucq@inria.fr, marc.durufle@inria.fr, mamadou.ndiaye@inria.fr

[2]Université de Bordeaux, Institut de Mathématiques de Bordeaux, Bordeaux, France. email: marc.durufle@inria.fr

[3]Université de Pau et des Pays de l'Adour, Laboratoire de Mathématiques et leurs Applications, Team Magique-3D Inria, avenue de l'université, 64012 Pau, France. email: helene.barucq@univ-pau.fr, mamadou.ndiaye@univ-pau.fr

local time-stepping techniques ([4], [5]) in order to have globally explicit schemes and a reduced computational cost with only few elements having a small time step. Another approach consists in applying globally implicit time-stepping techniques with unconditionally stability (Dahlquist's A-stable property [6]). This approach seems attractive especially for 1-D and 2-D simulations. But in the context of realistic applications (3-D heterogeneous media), it seems quite difficult to use a globally implicit time integration for wave equations due to the size of the linear system to be solved at each time step. This is why more recent investigations ([7], [8], [9]) deal with locally implicit schemes which provide methodologies involving the solution of linear systems only set on a small part of the computational domain. To ensure good levels of accuracy, such approach must involve time-integration schemes, whether explicit or implicit, which show robustness properties. For wave equations, robustness is characterized by dispersion and dissipation effects. In this work, we investigate implicit time-stepping techniques with the view of constructing very high-order unconditionally stable time discretization schemes, with low-dispersion and low-dissipation errors. By this way, we can dispose of high-order numerical methods that are increasingly relevant for practical applications involving wave equations. In fact, in the literature such time schemes seem not to be common beyond fourth-order.

Implicit Runge-Kutta schemes are very popular. They have the main advantage to be one-step schemes which do not need initialization schemes. Thus, this framework is mainly focused on the construction of high-order A-stable one-step methods. Linear multi-step schemes have not been considered because A-stable linear multi-step schemes are at most second-order schemes (known as the second Dahlquist's barrier [10]). Nevertheless, we have found some interesting investigations on multi-derivative multi-step schemes in [11] in which the author propose schemes up to order 5.

There are two main classes of implicit Runge-Kutta which are A-stable: Gauss Runge-Kutta schemes ([6]) that can be written at any order and some Singly Diagonally Implicit Runge Kutta (SDIRK) schemes that have been constructed up to order 5 ([6], [12], [13], [14]). Gauss Runge-Kutta schemes have the main drawback of requiring the solution of a very large linear system (the size increasing with the order in time), whereas SDIRK schemes require the solution of a unique linear system but the extension to higher order is not easy.

In this work, we are concerned with the solution of linear Ordinary Differential Equation (ODE) of the form

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

where A is a linear operator. We consider one-step schemes adapted to this class of ODE. We study schemes based on the diagonal Padé approximant of exponential (see [15]). In [16], these schemes are detailed for fourth-order. In this work, we have written them for order $2m$, $m \in \mathbb{N}^*$. It turned out that these schemes are equivalent to Gauss Runge-Kutta schemes. The main advantage of Padé schemes lies in the fact that they involve the solution of m successive linear systems of size N instead of solving a large system of size $m \times N$ with Gauss Runge-Kutta algorithm, m being the number of stages and N being the number of unknowns. In order to have a "fair" comparison with SDIRK schemes, we have developed schemes that require the inversion of the same linear system several times and that can be used only for linear ODEs. We called these schemes Linear-SDIRK. They are constructed by approximating the exponential with a fraction containing a unique pole (as initially studied in [17]). By adding extra-stages, we construct Linear-SDIRK up to order 12. It is possible to construct higher-order by adding more extra-stages. These schemes seem attractive when the memory is a critical issue (e.g. when a direct solver is used).

This paper is organized as follows. In Section 2, we describe the problem we are solving. In Sections 3 and 4 we present two classes of schemes that we are interested in: the diagonal Padé schemes and Linear-SDIRK schemes. We propose an efficient approach to implement them. In particular, for inhomogeneous ODEs, to address the implementation of the source field, we provide a method to correctly approximate the right hand side without losing the order of the schemes. An analysis of dispersion and dissipation is done to compare the developed schemes. Finally, these schemes are compared in 1-D and 2-D in Section 5 for the wave equation discretized with high-order finite elements using the C++ code Montjoie [18].

2 Preliminaries statement

2.1 General setting

We consider the following Ordinary Differential Equation (ODE)

$$\begin{cases} M_h \dfrac{dX(t)}{dt} + K_h X(t) = F(t) \quad t \in (0, T] \\ X(0) = X_0 \end{cases} \tag{1}$$

obtained after spatial discretization, where M_h is the mass matrix and K_h is the stiffness matrix. As usual h denotes the mesh size. $F(t)$ is a source term obtained after discretizing the continuous source term in space and X_0 is the initial condition. In the subsection 5.1, it is detailed how this ODE is obtained in the case of the solution 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) after one step is given by

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

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

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

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

Herein R is a rational function where both the numerator and denominator are polynomials of $\Delta t A$. The numerical schemes studied in this paper will consist of computing a sequence X_n, which is an approximation of the analytical solution $X(t_n)$, with the following numerical scheme:

$$X_{n+1} = R(\Delta t A) X_n + \widetilde{\phi}_n \tag{3}$$

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

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

R is called the stability function of the corresponding numerical scheme and its stability region is defined as

$$S = \{z \in \mathbb{C} \text{ such that } |R(z)| \leq 1\}.$$

In fact, since the analytical solution is stable if and only if

$$|\mathrm{e}^{\Delta t z}| = |\mathrm{e}^{\Delta t Re(z)}| \leq 1, \quad \forall z \in sp(A),$$

where $sp(A)$ represents the spectrum of the matrix A, the same condition must be satisfied by R for the numerical solution to be stable. For this reason the spectrum of A must be included in the negative half plane ($sp(A) \subset \mathbb{C}^-$). We then recall the following definition of Dahlquist's A-stability condition [6].

Definition A numerical scheme, whose stability region satisfies

$$S \supset \mathbb{C}^- = \{z \in \mathbb{C},\ Re(z) \leq 0\} \tag{4}$$

is called A-stable.

Let us define the stability function as:

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

where $N(z)$ and $D(z)$ are polynomials of z such that R is irreducible. As shown in [6], the corresponding numerical scheme is implicit when the degree of D is greater than one otherwise the numerical scheme is explicit. Since the A-stable requirement excludes rational functions that tend towards infinity when z tends to infinity, the degree of D must be greater or equal to the degree of N. In this work, we limit our-selves to the case where the degrees of D and N are equal and we will focus on two cases:

- $D(z)$ has distinct poles : we will choose the best rational approximation of the exponential known as the Padé approximation. The obtained schemes will be called Padé schemes. These schemes are detailed in Section 3, the numerator N and denominator D are given in (17).
- $D(z)$ has only one pole : we will construct the "best" approximation of the exponential satisfying the A-stable property. The obtained schemes will be called Linear-SDIRK schemes. These schemes are detailed in Section 4.

The second case deserves a particular interest because it induces the factorization of a unique linear system, whereas the first case implies to compute the solution to several linear systems. The stability function for Padé schemes is the same as the stability function for Gauss-Runge-Kutta schemes (see [6]). Gauss-Runge-Kutta schemes handle non-linear ODEs, whereas Padé schemes can be seen as a simplification of Gauss-Runge-Kutta schemes in the case of a linear ODE. The second case (Linear-SDIRK schemes) is well-known for non-linear ODEs as Singly Diagonal Runge-Kutta schemes (SDIRK). That is why we call them "Linear-SDIRK" since the constructed schemes will only apply to linear ODE.

Introducing $C = \Delta t A$, we define $R(C) = [D(C)^{-1}]N(C)$ as an approximation of e^C of order p ($\mathrm{e}^C = R(C) + O(\Delta t^{p+1})$) and we assume that $D(C)^{-1}$ is well defined. The analytical solution (2) can then be written as

$$D(C)X(t_{n+1}) = N(C)X(t_n) + \phi, \tag{5}$$

where

$$\phi = N(C)\int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u + O(\Delta t^{p+1}). \tag{6}$$

For homogeneous ODEs ($F(t) = 0$), it follows from (6) that $\phi = O(\Delta t^{p+1})$. Then, the numerical solution (3) satisfies

$$D(C)X_{n+1} = N(C)X_n \tag{7}$$

For inhomogeneous ODEs, i.e. $F(t) \neq 0$, we need to compute the quantity ϕ. However the integral in the expression (6) of ϕ is tedious to compute. We will rather compute the following equivalent quantity obtained from (5):

$$\phi = D(C)X(t_{n+1}) - N(C)X(t_n) \tag{8}$$

Finally, we propose the following numerical scheme:

$$\boxed{D(C)X_{n+1} = N(C)X_n + \phi_n} \tag{9}$$

where ϕ_n is an approximation of ϕ (up to a term in $O(\Delta t^{p+1})$). By using a Taylor expansion of $X(t_n)$ and $X(t_{n+1})$ around the time $t_n + \dfrac{\Delta t}{2}$ and using derivatives of the equation (1), we are able to compute ϕ_n in the following form:

$$\phi_n = \sum_{r=1}^{m} A^{r-1}\Delta t^r \sum_{i=0}^{n_w-1} \omega_i^r\, F\left(t_n + \Delta t\, c_i\right) \tag{10}$$

where m is the degree of the polynomial N or D and n_w is a number that depends on the scheme. This procedure will be detailed in subsections 3.2.2 and 4.6 for the two types of studied schemes.

2.2 Numerical dissipation and dispersion

> For time dependent problems, especially acoustic problems, a consistent, stable and convergent high order scheme does not guarantee a good quality numerical wave solution.
>
> *Christopher K. W. Tam and Jay Webb* [19]

A numerical scheme is dispersive if the numerical solution and exact solution have different phase speed while it is dissipative if they have difference in amplitude.

Following the analysis in [14] we present the explicit expression for the amplitude (dissipation) and phase (dispersion) errors of the numerical scheme knowing its stability function R. To

illustrate the dissipation and dispersion effects we will consider the following linear test equation

$$y' = i\lambda y, \quad y(t_0) = y_0 \text{ and } \lambda \in \mathbb{R}. \tag{11}$$

At each step of discretization, the numerical solution then reads

$$y_{n+1} = R(iz)y_n, \tag{12}$$

where $z = \lambda\,\Delta t$. The exact solution to the test equation (11) is given by

$$y_{n+1} = \mathrm{e}^{iz}y_n \tag{13}$$

The dispersion and dissipation errors can be measured by considering the ratio between the exact amplification factor ($R_e = \mathrm{e}^{iz}$) and numerical amplification factor ($R(iz)$).

Therefore, we define the dissipation and dispersion error, after each step, as follows:

Definition The leading dissipation error of a numerical scheme applied to (11) is measured by the function

$$\psi(z) = |R(iz)| - 1, \tag{14}$$

and the leading dispersion error of a numerical scheme applied to (11) is measured by the function

$$\Phi(z) := \arg\left[\frac{\mathrm{e}^{iz}}{R(iz)}\right] = z - arg[R(iz)]. \tag{15}$$

It is clear that a non-dissipative and non-dispersive scheme should ensure $|R(iz)| = 1$ and $\Phi(z) = 0$.

Remark In the previous definition, the function Φ is called the homogeneous dispersion. The dispersion error introduced by the homogeneous dispersion is linear in time and causes the numerical solution to become out of phase with respect to the exact solution. We refer to [14] for more details. In particular, [14] provides the definition of the error due to the inhomogeneous dispersion. It is constant in time and negligible regarding the error due to the homogeneous dispersion.

3 Padé Schemes for ODEs

The motivation of this section is to obtain a numerical solution for the ODE (1) using a Padé approximation of the exponential. Introduced by Henri Padé, Padé approximation is known to be an accurate approximation of a function by a rational function. For the exponential function, the general form of this approximation [20] is:

$$R_{r,s}(z) = \frac{N_{r,s}(z)}{D_{r,s}(z)} \tag{16}$$

where

$$N_{r,s}(z) = \sum_{i=0}^{s} \frac{s!\,(r+s-i)!}{(r+s)!\,i!\,(s-i)!}(z)^i \quad and \quad D_{r,s}(z) = \sum_{i=0}^{r} \frac{r!\,(r+s-i)!}{(r+s)!\,i!\,(r-i)!}(-z)^i. \tag{17}$$

By definition $R_{r,s}(z)$ is an approximation of order $(r+s)$ of e^z.

In [15] and [23], Ehle showed that the cases $r = s$, $r = s+1$ and $r = s+2$ are A-stable.

In the following, we will mainly focus our study on the case $r = s$ which corresponds to approximation that are commonly called diagonal Padé approximation to the exponential function. For convenience we now set $r = s = m$, $m \in \mathbb{N}$ and we note:

$$R_m(z) = R_{m,m}(z) = \frac{N_{m,m}(z)}{D_{m,m}(z)} = \frac{N_{m,m}(z)}{N_{m,m}(-z)}.$$

3.1 Numerical stability, dissipation and dispersion

From the Theorem $353A$ in [20] and the fact that $|N_m(ib)| = |N_m(-ib)|, \forall b \in \mathbb{R}$, we have the following result:

Proposition 3.1 *The stability function of numerical schemes obtained using the diagonal Padé approximation satisfies:* $\forall z \in \mathbb{C}^-$

$$|R_m(z)| \leq 1, \quad \forall m \in \mathbb{N}. \tag{18}$$

Furthermore, if $z = ib$, $b \in \mathbb{R}$,

$$|R_m(z)| = 1, \quad \forall m \in \mathbb{N}. \tag{19}$$

As a consequence,

- the diagonal Padé schemes are always A-stable,
- the diagonal Padé schemes when applied to the test equation (11) are not dissipative.

In addition, all the zeros of N_m are located in the negative half plane (see [15] and [21]). Since $D_m(z) = N_m(-z)$ by definition of R_m, D_m has all its zeros in the positive half plane. Furthermore, D_m has at most one real root which does exist when m is odd only, the other roots are complex conjugate. We have observed this result numerically and it can be deduced from the paper [22] in which the authors showed that all the poles of R_m converge to a curved right-side section of Szegö's curve.

Then, the only thing we have to worry about is the dispersion. The dispersion error can be represented quite faithfully by its Taylor expansion:

$$\frac{z - arg(R(iz))}{z} = \begin{cases} \dfrac{z^2}{12} - \dfrac{z^4}{80} + O(z^6), & \text{for } m = 1 \\ \dfrac{z^4}{720} - \dfrac{z^6}{12,096} + O(z^8), & \text{for } m = 2 \\ \dfrac{z^6}{100,800} - \dfrac{z^8}{2,592,000} + O(z^{10}), & \text{for } m = 3 \\ \dfrac{z^8}{25,401,600} - \dfrac{z^{10}}{869,299,200} + O(z^{12}), & \text{for } m = 4 \end{cases}$$

In Figure 1, we present the relative dispersion error of diagonal Padé schemes from order 2 to 10. In the x-coordinate, we have chosen to represent $\dfrac{z}{m}$, because m represents the computational complexity of the scheme. Indeed, $m = 1$ corresponds to the Crank-Nicolson scheme, where only one real linear system must be solved. $m = 2$ is a fourth-order scheme where only one complex linear system must be solved roughly assuming that the solution of a complex systems costs two solutions of a real system. $m = 3$ is a sixth-order scheme where one complex and one real linear system must be solved. The advantage is that we can compare fairly the different orders, we see clearly that the dispersion error is much smaller with higher order schemes.

3.2 Efficient implementation

3.2.1 Homogeneous case

In the homogeneous case, we consider (7) which is equivalent to

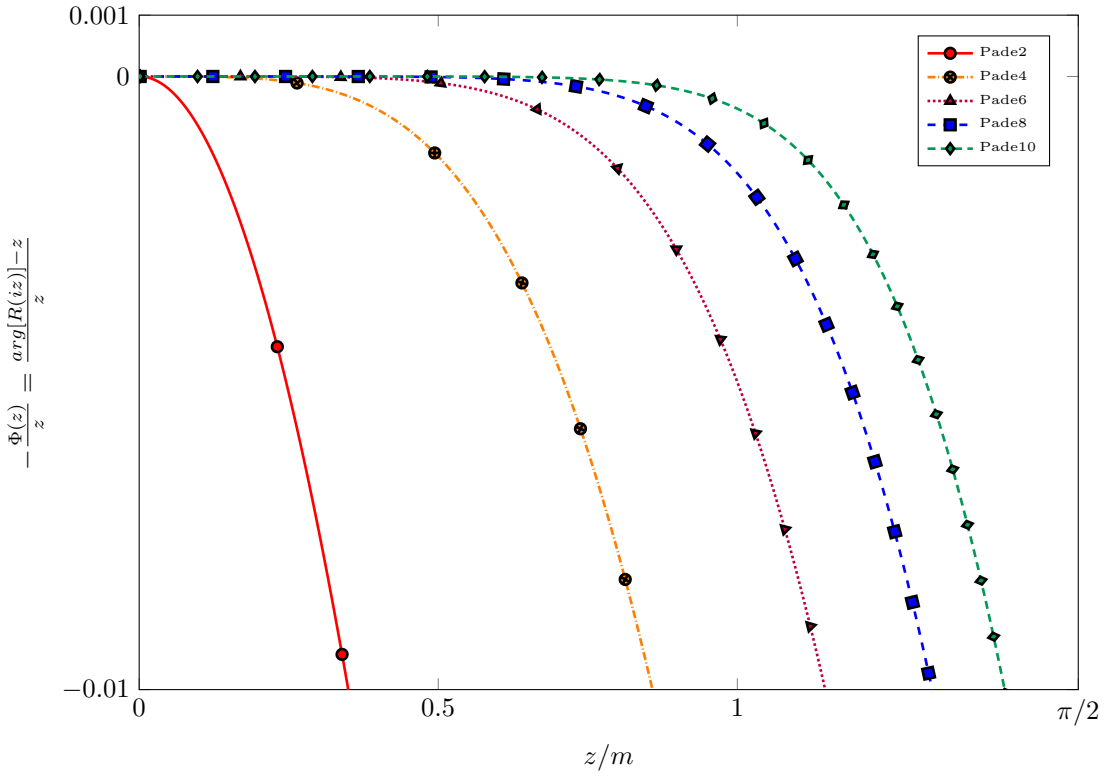
$$D_m(C)(X_{n+1} - X_n) = (N_m - D_m)(C)X_n.$$

Since $D_m(C) = N_m(-C)$, the polynomial $N_m - D_m$ contains non zero coefficients only for odd degree. It is then easy to compute the right hand side $(N_m - D_m)(C)X_n$ using Horner's algorithm. We note $G = (N_m - D_m)(C)X_n$ Now we have to solve the real linear system

$$D_m(C)(X_{n+1} - X_n) = G, \tag{20}$$

for each time step.

Figure 1: Dispersion of diagonal Padé schemes of order 2, 4, 6, 8 and 10 when applied to the test equation (11).



To perform this, we propose to factorize the polynomial D_m. Letting λ_k be zeroes of D_m and $Y_m = X_{n+1} - X_n$, we find

$$\left[\prod_{k=1}^{m}\left(I - \frac{C}{\lambda_k}\right)\right] Y_m = G. \tag{21}$$

Then we can solve successively the linear systems

$$\left(I - \frac{C}{\lambda_k}\right) Y_k = Y_{k-1} \quad k = 1, \ldots, m \tag{22}$$

$$Y_0 = G. \tag{23}$$

The final result Y_m is the desired solution. The next iterate X_{n+1} is then obtained as:

$$X_{n+1} = X_n + Y_m$$

Another way is to use a decomposition of $1/D_m$ as a sum of fractions with denominators of degree one which will lead to many independent linear systems that is convenient for parallelization [24].

Using the algebraic properties of D_m, we can optimize the computation algorithm. Indeed, when the degree m of the polynomial D_m is even, all its roots are complex conjugate. We write D_m as a product of second degree polynomial factors. Each second degree polynomial is the product of first degree polynomials obtained using complex conjugate roots of D_m:

$$\prod_{k=1}^{m/2}\left(I - \frac{C}{\lambda_k}\right)\left(I - \frac{C}{\overline{\lambda}_k}\right) Y_m = G. \tag{24}$$

When the degree m is odd, there is only one real root of D_m, other roots being complex conjugates. The real root λ is treated at the first step, as follows:

$$\begin{aligned} &\left(I - \frac{C}{\lambda}\right) Y_0 = G, \\ &\prod_{k=1}^{(m-1)/2}\left(I - \frac{C}{\lambda_k}\right)\left(I - \frac{C}{\overline{\lambda}_k}\right) Y_m = Y_0. \end{aligned} \tag{25}$$

We solve the second degree equation using the following algorithm proposed in [16]: Let λ_k be a complex root of D_m, $a_k = -\dfrac{1}{\lambda_k}$ and $\bar{a}_k = -\dfrac{1}{\overline{\lambda_k}}$. Then we take

$$P_2(C) = (I + a_k C)\,(I + \overline{a_k} C) = I + 2Re(a_k)C + a_k \bar{a}_k C^2.$$

A partial fraction decomposition of the polynomial $P_2^{-1}(x) = \dfrac{1}{(1 + a_k x)(1 + \bar{a}_k x)}$ allows to write

$$P_2^{-1}(C) = b_k (I + a_k C)^{-1} + \bar{b}_k (I + \bar{a}_k C)^{-1}, \tag{26}$$

with $b_k = \dfrac{a_k}{a_k - \bar{a}_k}$. To compute $Y_k = P_2^{-1}(C) Y_{k-1}$, we can compute

$$\begin{aligned} (I + a_k C)\, u &= Y_{k-1}, \\ (I + \bar{a}_k C)\, v &= Y_{k-1}, \\ Y_k &= b_k u + \bar{b}_k v. \end{aligned}$$

When the iterates X_n are real vectors, we have $v = (I + \bar{a}_k C)^{-1} Y_{k-1} = \overline{u}$. As a result, it suffices to solve only one system, giving the following algorithm:

$$\begin{aligned} (I + a_k C)\, u &= Y_{k-1}, \\ Y_k &= b_k u + \bar{b}_k \overline{u} = 2\, Re(b_k u). \end{aligned} \tag{27}$$

The case where iterates X_n are complex can be addressed by solving system (24) or (25) twice, for the real part and imaginary part of G.

3.2.2 Computation of the right hand side (RHS) ϕ

In the inhomogeneous case ($F(t) \neq 0$), we need to compute the coefficients ω_i^r involved in (10). This is done by using the Taylor expansion around $t_n + \dfrac{\Delta t}{2}$ of ϕ at order $p = 2m$. This expansion is completed with the expression (8) that we recall here:

$$\phi = D(C) X(t_{n+1}) - N(C) X(t_n) \tag{28}$$

We introduce the following notations

$$\rho_i^m = \frac{m!\,(2m-i)!}{(2m)!\; i!\; (m-i)!} = \binom{m}{i} \frac{(2m-i)!}{(2m)!} \quad \text{and} \quad C_k = \frac{1}{k!\; 2^{k-1}}. \tag{29}$$

Then, we have:

$$N_m(z) = \sum_{i=0}^{m} \rho_i^m z^i \qquad \text{and} \qquad D_m(z) = N_m(-z). \tag{30}$$

We perform the Taylor expansion of $X(t_{n+1}) = X(t_n + \Delta t)$ and $X(t_n)$ around $t_n + \dfrac{\Delta t}{2}$ at order $2m$. For simplicity we note $X^{(k)} = X^{(k)}\left(t_n + \dfrac{\Delta t}{2}\right)$ the k-th derivative of X(t) with respect to t at $t_n + \dfrac{\Delta t}{2}$. Recalling that $C = \Delta t A$, we replace $N(C) = N_m(\Delta t A)$ and $D(C) = D_m(\Delta t A)$ by their expression in (28). After performing the Taylor expansion it gives

$$\begin{aligned} \phi = &\sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k X^{(k)} \\ &- \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k X^{(k)} + O(\Delta t^{2m+1}). \end{aligned} \tag{31}$$

To evaluate $X^{(k)}$ we differentiate $(k-1)$-times the following relation

$$\frac{dX(t)}{dt} - AX(t) = F(t),$$

to obtain:

$$X^{(k)} = \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + A^k X^{(0)}, \tag{32}$$

where $F^{(j)}$ is the j-th derivative of the function F at point $t_n + \dfrac{\Delta t}{2}$ and $F^{(0)} = F(t_n + \dfrac{\Delta t}{2})$. Using formula (32) in expression (31) gives:

$$\begin{aligned}\phi = & \sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} \\ & - \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + S_m + O(\Delta t^{2m+1}),\end{aligned} \tag{33}$$

where S_m is given by

$$S_m = \sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k (\Delta t\, A)^{k+i} X^{(0)} - \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q,\, q\in\mathbb{N}}^{2m} C_k (\Delta t\, A)^{k+i} X^{(0)}. \tag{34}$$

Numerically we have observed that $S_m = O(\Delta t^{2m+1}), \forall m \in \mathbb{N}$. If $S_m \neq O(\Delta t^{2m+1})$, it would mean that the numerical scheme (7) (with $F = 0$) would be of order lower than $2m$ which is in contradiction with the properties of Padé approximant of the exponential. Actually, this property can be proved giving by this way a demonstration of the order of the scheme.

Proposition 3.2 *For all $m \in \mathbb{N}$,*

$$S_m = \sum_{r=2p+1,\, p\in\mathbb{N}}^{2m-1} \zeta_r^m (\Delta t A)^r X^{(0)} + O(\Delta t^{2m+1}), \tag{35}$$

with

$$\zeta_r^m = \sum_{i=0}^{min(m,r)} (-1)^i \rho_i^m C_{r-i} \tag{36}$$

Proof Let r be an integer defined by $r = k + i$. The result comes by developing the sum (34) and sorting by powers of $\Delta t A$. □

Theorem 3.3 *Let ζ_r^m be defined in (36). Then we have*

$$\zeta_r^m = 0, \quad r = 2p-1,\ 1 \leq p \leq m, \tag{37}$$

which implies that

$$S_m = O(\Delta t^{2m+1}). \tag{38}$$

The proof of the Theorem involves the following lemma:

Lemma 3.4 *Let $\Upsilon_{m,r}$ be the polynomial defined by*

$$\Upsilon_{m,r}(x) = \frac{(-1)^r}{(2m)!\ 2^{r-1}} x^m (x+2)^m. \tag{39}$$

Then

$$\frac{d^{2m-r}\Upsilon_{m,r}}{dx}(-1) = \zeta_r^m, \quad r = 2p-1,\ 1 \leq p \leq m. \tag{40}$$

Proof We use Newton's binomial formula to develop (39) which yields

$$\Upsilon_{m,r}(x) = \frac{(-1)^r}{(2m)!\ 2^{r-1}} \sum_{i=0}^{m} \binom{m}{i} 2^i x^{2m-i}.$$

Then we differentiate this expression $(2m-r)$-times. It gives

$$\frac{d^{2m-r}\Upsilon_{m,r}}{dx^{2m-r}}(x) = \frac{(-1)^r}{(2m)!\ 2^{r-1}} \sum_{i=0}^{min(m,r)} \binom{m}{i} \frac{(2m-i)!}{(r-i)!} 2^i x^{r-i},$$

which we evaluate at $x=-1$ to have

$$\begin{aligned}\frac{d^{2m-r}\Upsilon_{m,r}}{dx^{2m-r}}(-1) &= \frac{(-1)^r}{(2m)!\ 2^{r-1}} \sum_{i=0}^{min(m,r)} \binom{m}{i} \frac{(2m-i)!}{(r-i)!} 2^i (-1)^{r-i} \\ &= \frac{(-1)^{2r}}{(2m)!\ 2^{r-1}} \sum_{i=0}^{min(m,r)} \binom{m}{i} \frac{(2m-i)!}{(r-i)!} 2^i (-1)^{-i} \\ &= \sum_{i=0}^{min(m,r)} (-1)^i \binom{m}{i} \frac{(2m-i)!}{(2m)!} \times \frac{1}{(r-i)!\ 2^{r-i-1}} = \zeta_r^m . \square\end{aligned}$$

Proof Theorem 3.3 comes from the fact that all odd derivatives of $\Upsilon_{m,r}(x)$ equal zero at $x=-1$. In fact, we note that

$$\Upsilon_{m,r}(x) = \Upsilon_{m,r}(-x + 2 \times (-1)),$$

which means $\Upsilon_{m,r}(x)$ has an axis of symmetry at $x=-1$. Therefore the odd derivatives of $\Upsilon_{m,r}$ are equal to 0 at $x=-1$. Since r is odd, we obtain that $\frac{d^{2m-r}\Upsilon_{m,r}}{dx^{2m-r}}(-1)$ is equal to zero which gives $\zeta_r^m = 0$ for $r = 2p-1$. $\square$

Applying Theorem 3.3 to (33) implies the following result.

Corollary 3.5 *The simplified expression of ϕ defined in (33) is given by*

$$\begin{aligned}\phi = &\sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} \\ &- \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + O(\Delta t^{2m+1}).\end{aligned} \tag{41}$$

To achieve the order of accuracy $p=2m$ we can take k from 0 to $2m-1-i$ only which finally gives

$$\begin{aligned}\phi = &\sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m-1-i} C_k \Delta t^{k+i} \sum_{j=1}^{k} A^{i+k-j} F^{(j-1)} \\ &- \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m-1-i} C_k \Delta t^{k+i} \sum_{j=1}^{k} A^{i+k-j} F^{(j-1)} + O(\Delta t^{2m+1}).\end{aligned} \tag{42}$$

We change indexes in the sum by introducing $r = i+k-j+1$. We then obtain the following expression:

$$\begin{aligned}\phi = &\sum_{r=1}^{2m-1} \Delta t^r A^{r-1} \sum_{j=1, j-r=2q, q\in\mathbb{Z}}^{2m-r+1} \Delta t^{j-1} F^{(j-1)} \\ &\left(\sum_{i=2p, p\in\mathbb{N}}^{min(m,r-1)} \rho_i^m C_{r+j-i-1} - \sum_{i=2p+1, p\in\mathbb{N}}^{min(m,r-1)} \rho_i^m C_{r+j-i-1} \right) + O(\Delta t^{2m+1}).\end{aligned} \tag{43}$$

In the sum in j, j has the same parity as r. It means, that if r is even, j will be equal to 2, 4, 6, ... If r is odd, j will be equal to 1, 3, 5, ... Let us introduce

$$\alpha_j^r = \begin{cases} \sum_{i=0}^{min(m,r-1)} (-1)^i \rho_i^m C_{r+j-i}, & \text{if } j-r \equiv 0[2], \\ 0 \quad \text{otherwise.} \end{cases} \tag{44}$$

When $r \geq m+1$, we have

$$\alpha_j^r = \zeta_{r+j}^m = 0.$$

This induces that the sum in r can be reduced to a sum from 1 to m. Finally, ϕ is written as:

$$\phi = \sum_{r=1}^{m} \Delta t^r A^{r-1} \sum_{j=1}^{2m-r+1} \alpha_{j-1}^r \Delta t^{j-1} F^{(j-1)} + O(\Delta t^{2m+1}). \tag{45}$$

To illustrate (45), we provide then the expression of ϕ for two particular values of m:

Fourth-order diagonal Padé scheme The source vector ϕ for the fourth-order diagonal Padé scheme ($m = 2$) reads

$$\phi = \Delta t \left(F + \frac{\Delta t^2}{24} F^{(2)} \right) - A \frac{\Delta t^3}{12} F^{(1)} + O(\Delta t^5). \tag{46}$$

Sixth-order diagonal Padé scheme For $m = 3$, it is given as

$$\begin{aligned} \phi = \Delta t \left(F + \frac{\Delta t^2}{24} F^{(2)} + \frac{\Delta t^4}{1920} F^{(4)} \right) - A \Delta t^2 \left(\frac{\Delta t}{12} F^{(1)} + \frac{\Delta t^3}{480} F^{(3)} \right) \\ + A^2 \Delta t^3 \left(\frac{1}{60} F + \frac{\Delta t^2}{480} F^{(2)} \right) + O(\Delta t^7). \end{aligned} \tag{47}$$

In the current expression of ϕ, we need to approximate different derivatives of the function F. Our purpose is now to provide accurate formulas to compute $F^{(j)}$, $j \geq 1$.

3.2.3 Numerical approximation of the RHS ϕ

We consider the following approximation:

$$\sum_{i=0}^{2m-r} \alpha_i^r \Delta t^i F^{(i)} \approx \sum_{i=0}^{n_w-1} \omega_i^r F\left(t_n + \Delta t\, c_i\right), \tag{48}$$

where c_i are given points chosen in $[0,1]$ and ω_i^r represent the weights. We choose Gauss-Legendre points for c_i and w_i^r have to be computed for each r.

Since we have

$$F\left(t_n + \Delta t\, c_i\right) \approx \sum_{j=0}^{2m-1} \frac{\left(c_i - \frac{1}{2}\right)^j}{j!} \Delta t^j F^{(j)},$$

(48) can be written as follows

$$\sum_{i=0}^{2m-r} \alpha_i^r \Delta t^i F^{(i)} \approx \sum_{j=0}^{2m-1} \underbrace{\sum_{i=0}^{n_w-1} \omega_i^r \frac{\left(c_i - \frac{1}{2}\right)^j}{j!}}_{=\alpha_j^r} \Delta t^j F^{(j)}. \tag{49}$$

We identify $\Delta t^i F^{(i)}$ in (49) and deduce

$$\sum_{j=0}^{n_w-1} \omega_j^r \frac{\left(c_j - \frac{1}{2}\right)^i}{i!} = b_i = \begin{cases} \alpha_i^r, & \text{if } i \leq 2m-r, \\ 0 & \text{otherwise.} \end{cases} \tag{50}$$

We define the Vandermonde matrix $\mathrm{VDM} \in M_{n_w}(\mathbb{R})$ such that

$$\mathrm{VDM}_{i,j} = \frac{\left(c_j - \frac{1}{2}\right)^i}{i!}, \; 0 \leq i, j \leq n_w - 1.$$

Knowing c_i we evaluate ω_i^r by solving the linear system

$$\text{VDM}\ \omega^r = b, \tag{51}$$

At a first glance, $n_w = 2m-1$ should be needed to approximate correctly ϕ. It turns out that when we choose $n_w = m$ Gauss-Legendre points, the obtained approximation ϕ_n has the correct order. As a result, we have

$$n_w = m.$$

Finally we replace c_i and ω_i^r in (48) to approximate ϕ in (45). ϕ_n is therefore given by

$$\phi_n = \sum_{r=1}^{m} A^{r-1} \Delta t^r \sum_{i=0}^{m-1} \omega_i^r \, F\left(t_n + \Delta t \, c_i\right).$$

Remark We have observed that Padé schemes with this approximation of ϕ_n based on Gauss-Legendre points are strictly equivalent to Gauss-Runge-Kutta schemes. Padé schemes can be seen as a different algorithm to compute X_{n+1} from X_n. The advantage of this algorithm is that only systems of size N have to be solved (complex and/or real) where N is the number of degrees of freedom (i.e. the size of the vector X_n), whereas Gauss-Runge-Kutta method requires the solution of a system of size $m \times N$.

Remark If Gauss-Runge-Kutta schemes are considered for linear ODEs, the intermediary unknowns $k_1, k_2, \cdots k_m$ can be eliminated formally to obtain a polynomial system to solve for $y_{n+1} - y_n$. By performing this elimination, we obtain the Padé schemes. However, we think that it is easier to implement the computation of coefficients ω_i^r with our procedure (through the solution of a Vandermonde matrix).

Remark The presence of the source does not imply any additional matrix-vector product with the matrix $C = \Delta t A$. Indeed, the computation of ϕ_n is mixed with the computation of $G = (N_m - D_m)(C)X^n$ such that only the evaluations of F represent an additional cost of the inhomogeneous case.

4 Linear-SDIRK methods $s+l$-stages of $(s+1)^{th}$ order

> To be A-Stable, and possibly useful for stiff systems, a Runge-Kutta formula must be implicit.
>
> *R. Alexander 1977* [12]

Historically, the first Runge-Kutta schemes developed were explicit and only of second order. The need of high order schemes and the apparition of stiff problems led to the developments of implicit Runge-Kutta schemes using first the Gauss quadrature formula to have order of $2s$ when a scheme of s stages is used. But this kind of scheme requires the solution of a large linear system of size $s \times N$ (s being the number of stages and N being the number of unknowns) at each time step which make this approach quite inefficient.

> There is a significant computational advantage in diagonally implicit formulae, whose coefficient matrix is lower triangular with all diagonal element equal.
>
> *R. Alexander 1977* [12]

To reduce the computational burden, the idea is to construct implicit Runge-Kutta scheme in which we will have to solve the same linear system of size N with different right hand sides at each time step. This kind of method is called a Singly Diagonally Implicit Runge-Kutta (SDIRK) method ([6], [12], [13], [20]).

In this section we will consider a rational polynomial function $R(z)$ which approximates the exponential function and minimizes the error. In order to have the same linear system to solve, the denominator of $R(z)$ is given by

$$D(z) = (1 - \gamma z)^{s+l},$$

where γ is a real positive number and $s+l$ is the number of stages. In this section, we propose to find the numerator $N(z)$ with the best constant γ satisfying the following requirements

- The method (3) is A-stable,
- The method (3) is of order $s+1$.

We denote $R_s^l(z)$ the obtained stability function. By construction, we will have:

$$\mathrm{e}^z - R_s^l(z) = \eta\, z^{s+2} + O(z^{s+3}).$$

The stability function R_s^l will be found by minimizing the coefficient $|\eta|$ under the constraints described above. The resulting schemes are called Linear-SDIRK schemes.

Remark The rational polynomial function $R_s^l(z)$ constructed by this approach will coincide with the stability function of Singly Diagonal Runge-Kutta (SDIRK) schemes for low orders (2, 3 and 4). However, this is no longer the case for higher order schemes. Indeed, from the stability function and by imposing the so-called order conditions [20], one can try to reconstruct Runge-Kutta coefficients. For higher orders, there are too many order conditions (because of non-linear order conditions) to be satisfied, such that we are not able to find Runge-Kutta coefficients. This means that the developed schemes in this section work only for linear ODEs and that is why we called them Linear-SDIRK schemes.

The stability function of the $(s+1)^{th}$-order Linear-SDIRK schemes is given by

$$R_s^l(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^s}{s!} + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_{s+l} z^{2s+l+1}}{(1-\gamma z)^{s+l}}.$$

In this form, R_s^l has the correct order by construction. It is then sufficient to satisfy the A-stability condition. In the following subsections, we describe the obtained schemes for $l=0$, $l=1$, $l=2$ and $l=3$. In practice (to implement the numerical scheme), we use the following expression of R_s^l:

$$R_s^l(z) = \frac{N_s^l(z)}{(1-\gamma z)^{s+l}},$$

where the expression of N_s^l is given in equations (55), (57), (58) and (59) for respectively $l=0$, $l=1$, $l=2$ and $l=3$. In the following, we use D_s^l as

$$D_s^l(z) = (1-\gamma z)^{s+l}.$$

4.1 Linear-SDIRK methods s-stages of order $s+1$

In this section we choose $l=0$ and we present the constructions of Linear-SDIRK scheme of order $s+1$ with a minimal number of stages s, $s \in \mathrm{N}$.

Order 2 The Linear-SDIRK of order 2 is found for $s=1$. Its stability function $R_1^0(z)$ is sought as

$$R_1^0(z) = 1 + z + \frac{z^2}{2} + \frac{\alpha_0 z^3}{(1-\gamma z)}.$$

Obviously, the associated scheme is of order 2. We have

$$R_1^0(z) = \frac{1 + (1-\gamma)z + \left(\frac{1}{2} - \gamma\right) z^2 + \left(\alpha_0 - \frac{\gamma}{2}\right) z^3}{(1-\gamma z)}.$$

In order to have a A-stable scheme, we need to satisfy at least $\gamma = \frac{1}{2}$ and $\alpha_0 = \frac{\gamma}{2}$. As a result, we obtain

$$R_1^0(z) = \frac{1 + \frac{z}{2}}{1 - \frac{z}{2}},$$

which is the stability function of the Crank-Nicolson scheme.

Order 3 To construct a third order Linear-SDIRK scheme with $s = 2$ and $l = 0$, we must find α_1, α_2 and γ such that:

$$R_2^0(z) = 1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \frac{\alpha_1 z^4 + \alpha_2 z^5}{(1-\gamma z)^2}.$$

As detailed for $s = 1$, we reduce to the least common denominator to find conditions on α_1, α_2 and γ for the third order approximation. We get

$$\alpha_2 + \frac{\gamma^2}{6} = 0 \quad \text{and} \quad \alpha_1 - \frac{\gamma}{3} + \frac{\gamma^2}{2} = 0, \tag{52}$$

$$\frac{1}{6} - \gamma + \gamma^2 = 0. \tag{53}$$

We compute γ as a solution to (53), α_1 and α_2 are deduced from relations in (52). The obtained stability function is then given by

$$R_2^0(z) = \frac{1 + (1-2\gamma)z + \left(\frac{1}{2} - 2\gamma + \gamma^2\right) z^2}{(1-\gamma z)^2}.$$

The two possible choices for γ are $\frac{1}{2} - \frac{1}{2\sqrt{3}}$ and $\frac{1}{2} + \frac{1}{2\sqrt{3}}$ (roots of (53)). The one that leads to A-stable scheme is $\gamma = \frac{1}{2} + \frac{1}{2\sqrt{3}}$. In fact with this choice of γ the modulus of the asymptote of $R_2^0(z)$ when z tends to $+\infty$ satisfies

$$\left| \frac{\frac{1}{2} - 2\gamma + \gamma^2}{\gamma^2} \right| < 1,$$

which is a necessary condition to have an A-stable scheme. The other root does not satisfy this condition. The associated method has the same stability function as the SDIRK of order 3 obtained by Crouzeix (see [12]).

Order 4 To construct a fourth order Linear-SDIRK scheme with $s = 3$ and $l = 0$, we must find α_1, α_2, α_3 and γ such that:

$$R_3^0(z) = 1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \frac{z^4}{4!} + \frac{\alpha_1 z^5 + \alpha_2 z^6 + \alpha_3 z^7}{(1-\gamma z)^3}.$$

As previously, we obtain α_1, α_2, α_3 from γ. The parameter γ is solution to

$$\gamma^3 - \frac{3}{2}\gamma^2 + \frac{\gamma}{2} - \frac{1}{24} = 0,$$

which is necessary for the A-stability condition. Only one root of this equation leads to an A-stable scheme. It is

$$\gamma = \frac{1}{\sqrt{3}} \cos\left(\frac{\pi}{18}\right) + \frac{1}{2}.$$

We note the polynomial

$$P(z) = (1-\gamma z)^3 \left(1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \frac{z^4}{4!}\right) = a_0(\gamma) + a_1(\gamma) z + \cdots + a_7(\gamma) z^7.$$

The numerator $N_3^0(z)$ is then obtained by truncating this polynomial (since the coefficients α_1, α_2 and α_3 are set to cancel the higher order terms):

$$N_3^0(z) = a_0(\gamma) + a_1(\gamma) z + a_2(\gamma) z^2 + a_3(\gamma) z^3.$$

The stability function of the Linear-SDIRK of order 4 with $s = 3$ and $l = 0$ is then

$$R_3^0(z) = \frac{N_3^0(z)}{(1-\gamma z)^3}.$$

This scheme has the same stability function as the SDIRK scheme of order 4 obtained by Crouzeix (see [12]).

General case Now we present a general method to construct a s-stages Linear-SDIRK scheme of order $s+1$. Let R_s^0 be the stability function. We search for R_s^0 of the form:

$$R_s^0(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_s z^{2s+1}}{(1-\gamma z)^s}.$$

We note the polynomial P that appears while reducing to the common denominator:

$$P(z) = (1-\gamma z)^s \left(1 + z + \frac{z^2}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!}\right) = a_0(\gamma) + a_1(\gamma) z + \cdots + a_{2s+1}(\gamma) z^{2s+1}.$$

The constants α_i are chosen to balance higher-order terms of P, i.e.

$$\alpha_i = -a_{s+1+i}(\gamma), \quad i = 1 \dots s.$$

A necessary condition to obtain A-stable property is that the term in z^{s+1} vanishes, that is to say

$$a_{s+1}(\gamma) = \sum_{i=0}^{s} \frac{(-\gamma)^i \binom{s}{i}}{(s+1-i)!} = 0. \tag{54}$$

Finally, the numerator of $R_s^0(z)$ is given by

$$N_s^0(z) = a_0(\gamma) + a_1(\gamma) z + a_2(\gamma) z^2 + \cdots + a_s(\gamma) z^s. \tag{55}$$

We have

$$R_s^0(z) = \frac{N_s^0(z)}{(1-\gamma z)^s}.$$

To ensure the A-stability condition we choose γ as follows: for each γ root of (54), we compute the asymptote of $R_s^0(z)$ when z tends to infinity. If the asymptote is lower or equal to 1, we look for the roots of the following polynomial equation:

$$|N(i\sqrt{z})|^2 = |D(i\sqrt{z})|^2. \tag{56}$$

In fact, since $\gamma > 0$, R_s^0 is holomorphic in the negative half plane ($\mathbb{C}^-$) and based on the maximum principle, R_s^0 will reach its maximal value on the imaginary axis or when $|z|$ tends to infinity. If the polynomial equation (56) has no real roots except zero, then the scheme is A-stable, otherwise the scheme is not A-stable. We present in Table 1, the A-stable schemes we have obtained.

Table 1: Minimal stage Linear-SDIRK of order **s+1** and associated value of γ.

s	value of γ	comment
1	0.5	-
2	0.788675134594813	-
3	1.068579021301629	-
4	x	No A-stable schemes
5	0.473268391258295	-
$r \geq 6$	x	No A-stable schemes

As presented in Table 1, 6 is the maximal order that we can achieve. In fact, for $s \geq 6$, there is no root γ that leads to a A-stable scheme. Without extra stage ($l = 0$), we have retrieved the Linear-SDIRK schemes presented by Burrage [17]. To get higher order schemes, we need to increase the number of stages. This will be addressed in the next subsections.

4.2 Linear-SDIRK methods $(s+1)$-stages of order $s+1$

Here we add one extra stage which corresponds to $l = 1$. The stability function is then equal to

$$R_s^1(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_{s+1} z^{2s+2}}{(1-\gamma z)^{s+1}}.$$

In this case, γ is a free parameter. We note P the polynomial that is involved while reducing to the common denominator:

$$P(z) = (1-\gamma z)^{s+1}\left(1 + z + \frac{z^2}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!}\right) = a_0(\gamma) + a_1(\gamma)z + \cdots + a_{2s+2}(\gamma)z^{2s+2}.$$

The constants α_i are chosen to balance higher-order terms of P, i.e.

$$\alpha_i = -a_{s+1+i}(\gamma), \quad i = 1 \ldots s+1.$$

Finally, the numerator of $R_s^1(z)$ is given by

$$N_s^1(z) = a_0(\gamma) + a_1(\gamma)z + a_2(\gamma)z^2 + \cdots + a_{s+1}(\gamma)z^{s+1} \tag{57}$$

and we thus have

$$R_s^1(z) = \frac{N_s^1(z)}{(1-\gamma z)^{s+1}}.$$

The optimization is done by minimizing $|\eta| = |\alpha_1 - \frac{1}{(s+2)!}|$ under the A-stability constraint. Since γ is the only free parameter, this constraint imposes that γ belongs to an interval or a set of

Table 2: Linear-SDIRK of order **s+1** with one additional stage

s	interval for γ	γ_{opt}
1	$[\frac{1}{4}, \infty[$	leads to third order
2	$[\frac{1}{3}, 1.06866]$	leads to fourth order
3	$[0.39434, 1.28057]$	0.394337567297407
4	$[0.24651, 0.3618] \cup [0.42079, 0.47326]$	leads to sixth order
5	$[0.28407, 0.5409]$	0.284064638011799
6	x	No A-stable schemes
7	$[0.21705, 0.26471]$	0.217049743094304
$r \geq 8$	x	No A-stable schemes

intervals. The admissible intervals for γ, for which the schemes are A-stable, are represented in Table 2. The optimal value of γ that minimizes the error term $|\alpha_1|$ is also provided for each s. We have obtained the same admissible intervals as Burrage (see [17]).

We see here that we are able to obtain an A-stable scheme of order eight contrary to the previous section. We have also found a fifth-order A-stable scheme which after optimization leads to a sixth order scheme.

Remark The optimization has been performed manually, by finding first the stable region and then zooming in to find the optimal value of γ. The values of γ we found are rather small. As a result, the matrix $(I - \gamma \Delta t A)$ will be well-conditioned.

4.3 Linear-SDIRK methods $(s+2)$-stages of order $s+1$

To obtain higher order Linear-SDIRK scheme we increase the number of stages. Here we take $l = 2$ (instead of $l = 1$ in the previous subsection), and it leads to Linear-SDIRK schemes with two additional stages:

$$R_s^2(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_{s+2} z^{2s+3}}{(1-\gamma z)^{s+2}}.$$

In this case we have two free parameters γ and α_1. We let P be the polynomial that appears while reducing to the common denominator:

$$P(z) = (1-\gamma z)^{s+2}\left(1 + z + \frac{z^2}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!}\right) = a_0(\gamma) + a_1(\gamma)z + \cdots + a_{2s+3}(\gamma)z^{2s+3}.$$

The constants α_i are chosen to cancel out higher-order terms of P, i.e.

$$\alpha_i = -a_{s+1+i}(\gamma), \quad i = 2 \dots s+2.$$

Finally, the numerator of $R_s^2(z)$ is given by

$$N_s^2(z) = a_0(\gamma) + a_1(\gamma)z + a_2(\gamma)z^2 + \cdots + a_{s+1}(\gamma)z^{s+1} + (a_{s+2}(\gamma) + \alpha_1)z^{s+2}. \tag{58}$$

We have

$$R_s^2(z) = \frac{N_s^2(z)}{(1-\gamma z)^{s+2}}.$$

The optimization is done by minimizing $|\eta| = |\alpha_1 - \frac{1}{(s+2)!}|$ under the A-stability constraint. Since we have two parameters, this constraint will impose that γ and α_1 belong to 2-D regions.

Table 3: Linear-SDIRK of order **s+1** with two additional stages

s	γ_{opt}	$\alpha_{1_{opt}}$	comment
$1 \leq s \leq 4$	-	-	No uniqueness
5	0.204071	$1.9839430662 \cdot 10^{-4}$	-
6	-	-	leads to eighth order
7	0.16689	$2.9259251764 \cdot 10^{-6}$	-
8	x	x	No A-stable schemes
9	0.141940	$2.2982637210 \cdot 10^{-8}$	-
$r \geq 10$	x	x	No A-stable schemes

The optimal values of γ and α_1 are presented in Table 3. We see here that we are able to obtain an A-stable tenth-order scheme (versus 8 in the previous subsection). For $1 \leq s \leq 4$, we did not find a unique optimal choice for the two free parameters.

4.4 Linear-SDIRK methods $(s+3)$-stages of order $s+1$

Following the results obtained in previous subsections, we increase again the number of additional stages up to $l = 3$. The stability function is then written as follows:

$$R_s^3(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_{s+3} z^{2s+4}}{(1-\gamma z)^{s+3}}.$$

We have three free parameters γ, α_1 and α_2. Like in previous subsections, we note P the polynomial that appears while reducing to the common denominator:

$$P(z) = (1-\gamma z)^{s+3}\left(1 + z + \frac{z^2}{2!} + \cdots + \frac{z^{s+1}}{(s+1)!}\right) = a_0(\gamma) + a_1(\gamma)z + \cdots + a_{2s+4}(\gamma)z^{2s+4}.$$

The constants α_i, $i = 3 \dots s+3$ are chosen to compensate higher-order terms of P, i.e.

$$\alpha_i = -a_{s+1+i}(\gamma), \quad i = 3 \dots s+3.$$

Finally, the numerator of $R_s^3(z)$ is given by

$$N_s^3(z) = a_0(\gamma) + a_1(\gamma)z + a_2(\gamma)z^2 + \cdots + a_{s+1}(\gamma)z^{s+1} + (a_{s+2}(\gamma) + \alpha_1)z^{s+2} + (a_{s+3}(\gamma) + \alpha_2)z^{s+3}. \tag{59}$$

The stability function reads:

$$R_s^3(z) = \frac{N_s^3(z)}{(1-\gamma z)^{s+3}}.$$

The optimization is done by minimizing $|\eta| = |\alpha_1 - \frac{1}{(s+2)!}|$ under the A-stability constraint. The optimal values of γ, α_1 and α_2 are presented in Table 4. We see here that we are able to obtain an A-stable twelve order scheme (versus 10 in the previous section). For $1 \leq s \leq 6$, we did not find a unique optimal choice for the three free parameters.

Table 4: Linear-SDIRK of order **s+1** with three additional stages

s	γ_{opt}	$\alpha_{1_{opt}}$	$\alpha_{2_{opt}}$
$1 \leq s \leq 6$	-	-	No uniqueness
7	0.136339	$2.767416226 \cdot 10^{-06}$	$-3.464398093 \cdot 10^{-06}$
8	-	-	-
9	0.151706	$2.459114959 \cdot 10^{-08}$	$-4.3140917546 \cdot 10^{-08}$
10	x	x	x
11	0.132572	$1.644515143 \cdot 10^{-10}$	$-2.89891484131 \cdot 10^{-10}$
$r \geq 12$	x	x	x

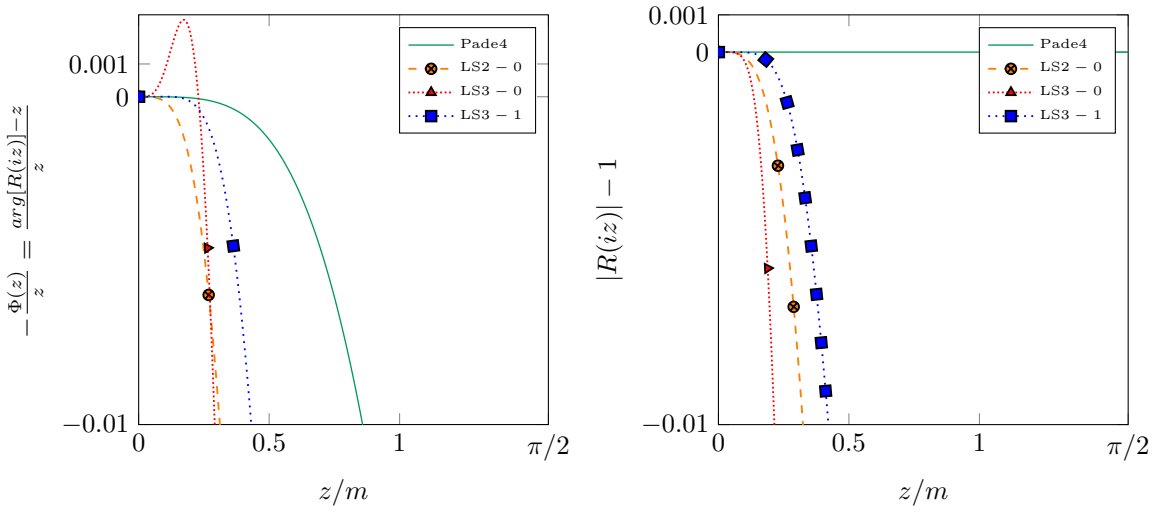
Remark There are no A-stable schemes of order 5, 7, 9 and 11 for l equal respectively to 0, 1, 2 and 3. As far as we know, there won't be any A-stable schemes of order 13, 15, ... for l respectively equal to 4, 5, ... It can be conjectured that the maximal order for an A-stable Linear-SDIRK scheme is $2l + 6$.

Remark Except the third order scheme obtained for $l = 0$, a scheme of odd order p leads to a scheme of order $p + 1$ after optimization. That is why only even orders are represented in Tables 3 and 4.

4.5 Numerical stability, dissipation and dispersion

By construction, all the Linear-SDIRK schemes presented in this paper are A-stable. Furthermore, the second order Linear-SDIRK has the same stability function as the second order diagonal Padé scheme. Both are not dissipative when applied to the test equation (11) and have the same dispersion error. In Figures 2, 3, 4 and 5, we present the relative dispersion error (on the left) and the relative dissipation error (on the right) of diagonal Padé schemes compared to the Linear-SDIRK schemes. In the x-axis, we have chosen to represent $\frac{z}{m}$. m represents the computational complexity of the scheme (see subsection 3.2 for diagonal Padé schemes). For the Linear-SDIRK schemes of order $s + 1$, $m = s + l$ is the number of linear systems to be solved to compute the numerical solution after one step.

Figure 2: Dispersion and dissipation curves of diagonal Padé schemes of order 4 compared with that of the Linear-SDIRK, when applied to the test equation (11). LS$2 - l$ and LS$3 - l$ represents the $s = 2$ and $s = 3$ plus l additional stages Linear-SDIRK of order 3 and 4.



Remark In the Figures 2, 3, and 4, the diagonal Padé schemes, which are not dissipative, are less dispersive than any Linear-SDIRK schemes of the same order. Among the Linear-SDIRK schemes with the same order, the less dispersive and the less dissipative scheme corresponds to the one with maximal additional stages.

Figure 3: Dispersion and dissipation curves of diagonal Padé schemes of order 6 compared with that of the Linear-SDIRK, when applied to the test equation (11). LS5 $-$ l represents the $s = 5$ plus l additional stages Linear-SDIRK of order 6.

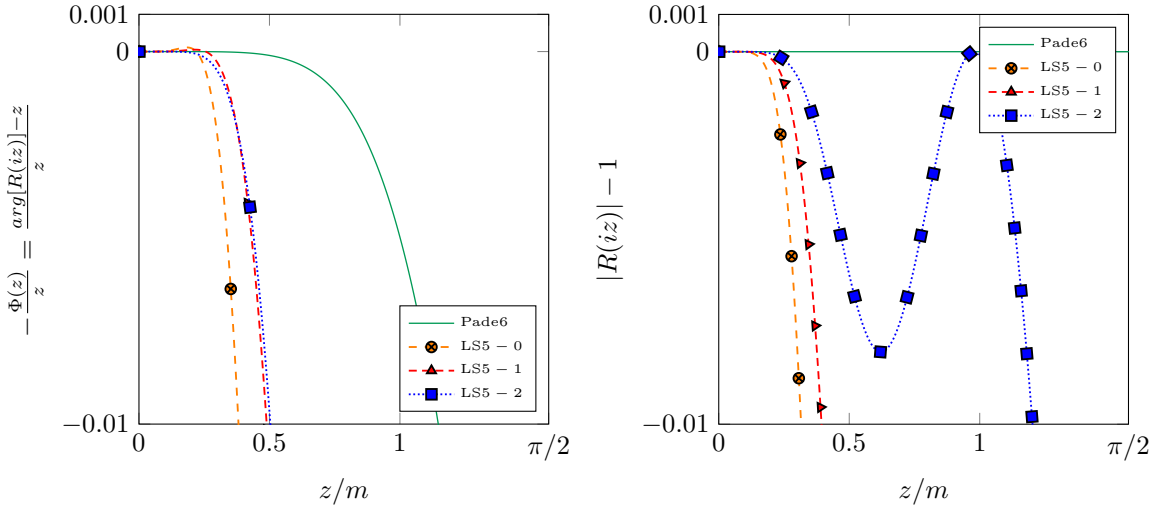
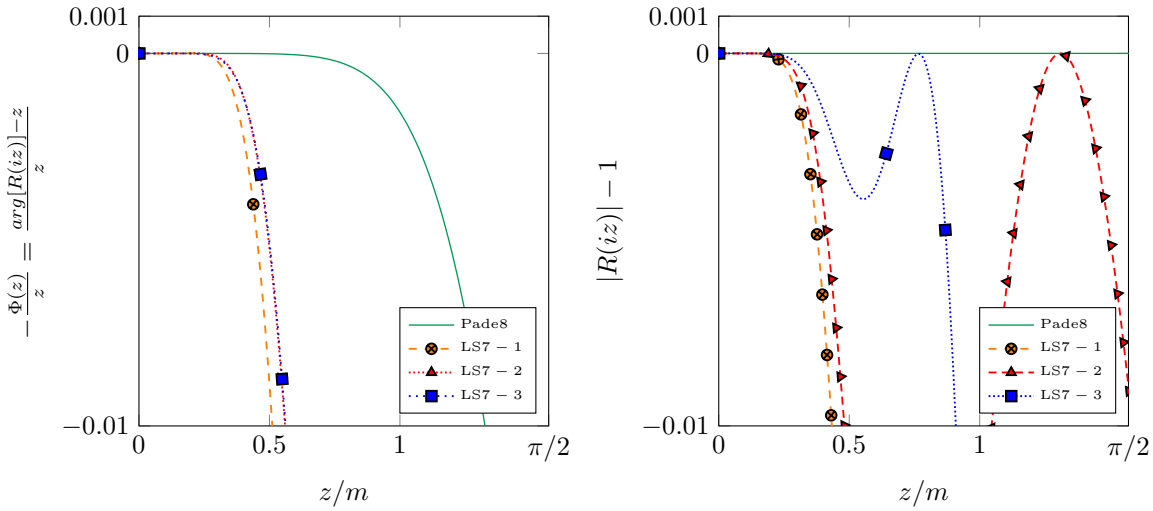


Figure 4: Dispersion and dissipation curves of diagonal Padé schemes of order 8 compared with that of the Linear-SDIRK, when applied to the test equation (11). LS7 $-$ l represents the $s = 7$ plus l additional stages Linear-SDIRK of order 8.



4.6 Handling a right hand side term

The stability function of Linear-SDIRK schemes of order $s \geq 2$ can be written in the form

$$R_s^l(z) = \frac{N_s^l(z)}{D_s^l(z)}.$$

with $N_s^l(z)$ and $D_s^l(z)$ defined in the previous subsections. Like in Padé schemes we introduce

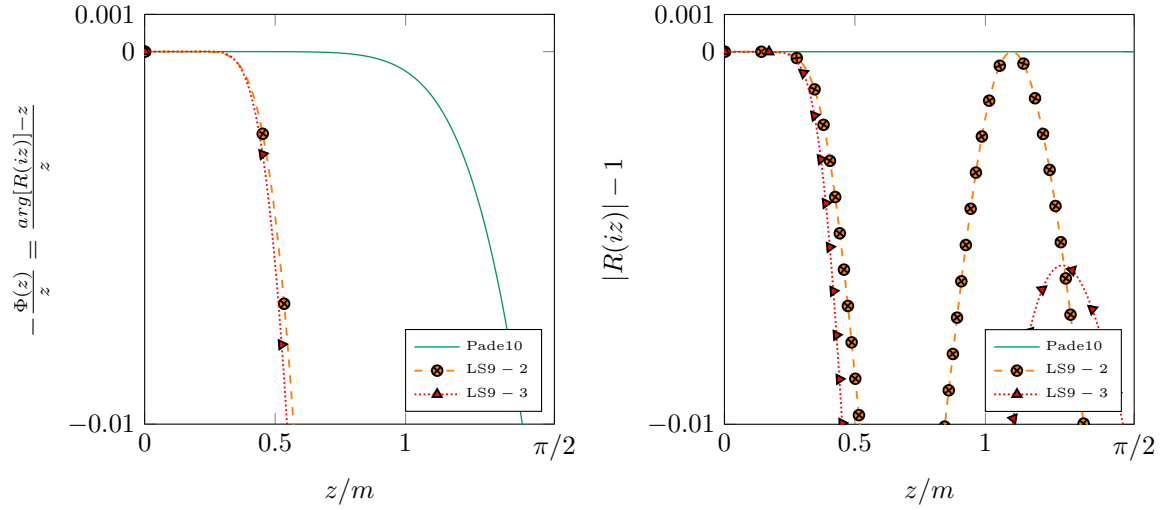
$$\phi = D_s^l(C)X(t_{n+1}) - N_s^l(C)X(t_n) + O(\Delta t^{s+2}).$$

Polynomials D_s^l and N_s^l are represented as follows:

$$D_s^l(z) = \sum_{i=0}^{s+l} D_i z^i, \quad N_s^l(z) = \sum_{i=0}^{s+l} N_i z^i.$$

We perform the Taylor expansion of $X(t_{n+1}) = X(t_n + \Delta t)$ and $X(t_n)$ around $t_n + \dfrac{\Delta t}{2}$ at order $s+1$. For simplicity we note $X^{(k)} = X^{(k)}\left(t_n + \dfrac{\Delta t}{2}\right)$ the k-th derivative of X(t) with respect to t at

Figure 5: Dispersion and dissipation curves of diagonal Padé schemes of order 10 compared with that of the Linear-SDIRK, when applied to the test equation (11). LS9 $-$ l represents the $s = 9$ plus l additional stages Linear-SDIRK of order 10.



$t_n + \dfrac{\Delta t}{2}$. We obtain:

$$\phi = \sum_{i=0}^{s+l} \sum_{k=0}^{s+1} \left(\frac{\Delta t}{2}\right)^k \frac{C^i}{k!} \left(D_i - (-1)^k N_i\right) X^{(k)} + O(\Delta t^{s+2}).$$

Then, we use the relation

$$X_n^{(k)} = \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + A^k X_n,$$

to obtain the following expression

$$\begin{aligned}\phi = & \left[\sum_{i=0}^{s+l} \sum_{k=0}^{s+1} \frac{C^{i+k}}{2^k\, k!} \left(D_i - (-1)^k N_i\right)\right] X^{(0)} \\ & + \Delta t \left[\sum_{i=0}^{s+l} \sum_{k=0}^{s+1} \sum_{j=1}^{k} \frac{C^{i+k-j}}{2^k\, k!} \left(D_i - (-1)^k N_i\right) \Delta t^{j-1} F^{j-1}\right] + O(\Delta t^{s+2}).\end{aligned}$$

The first term is in $O(\Delta t^{s+2})$ because the homogeneous scheme is of order $s + 1$. Regarding the second term, we introduce $r = i + k - j + 1$ to find

$$\phi = \Delta t \sum_{r=1}^{s+1} (\Delta t A)^{r-1} \sum_{j=1}^{s+2-r} \alpha_{j-1}^{r,l} \Delta t^{j-1} F^{(j-1)} + O(\Delta t^{s+2})$$

where

$$\alpha_{j-1}^{r,l} = \sum_{i=0}^{\min(r-1,s+l)} \frac{1}{2^{r+j-i-1}(r+j-i-1)!} \left(D_i - (-1)^{r+j-i-1} N_i\right). \tag{60}$$

We can notice that

$$\alpha_0^{s+1,0} = 0$$

because γ solves Equation (54). It means that the sum over r can be reduced to $r \in [1, s]$ when $l = 0$. Finally we end up with the approximation

$$\sum_{i=0}^{s+1} \alpha_i^{r,l} \Delta t^i F^{(i)} \approx \sum_{i=0}^{n_w-1} \omega_i^{r,l} F\left(t_n + \Delta t\, c_i\right).$$

We have chosen a sum from 0 to $s + 1$ instead of $s + 1 - r$ because the obtained schemes are more accurate with this choice. It can be noted that the coefficients $(\alpha_i^{r,l})_{i>s+1-r}$ can be chosen freely,

they do not affect the order of accuracy. In the same way, the points c_i can be chosen freely. We have made the choice to take the nominal values for $(\alpha_i^{r,l})_{i>s+1-r}$ (as defined by equation (60)) and to take $s+l+1$ Gauss-Legendre points for c_i in the interval $[0,1]$ ($n_w = s+1$). The weights $\omega_i^{r,l}$ are found by solving a Vandermonde system as detailed in the Paragraph 3.2.3. ϕ_n is therefore computed by using the formula

$$\phi_n = \sum_{r=1}^{s+l} A^{r-1} \Delta t^r \sum_{i=0}^{s} \omega_i^{r,l} F\left(t_n + \Delta t\, c_i\right).$$

4.7 Stable algorithm

For Linear-SDIRK schemes with $s+l \geq 8$, we have observed an instability because of the very large eigenvalues of the matrix C due to a local refinement for instance. For $s+l=7$, the scheme is stable but polluted by round-off errors such that it can be less efficient than fourth-order Linear-SDIRK schemes. Indeed, this instability occurs because of round-off errors, it does not occur when we are using quadruple precision arithmetic for example. This instability is due to Hörner's algorithm. We have detailed in the Algorithms 1 and 2 the stable algorithms in the case where N_s^l has real and complex conjugate roots. These roots are grouped together such that only second-degree polynomials of A are involved.

When no source is present, a stable algorithm consists in factorizing the stability function $R_s^l(z)$ as follows

$$R_s^l(z) = \left(\prod_{k=1}^{n_r} \frac{1-\dfrac{z}{\lambda_k}}{1-\gamma z}\right) \quad \left(\prod_{k=1}^{n_c} \frac{(1-b_k z + a_k z^2)}{(1-\gamma z)^2}\right),$$

and of using this factorization to get a stable algorithm in Algorithm 1 when there is no source term.

Algorithm 1 Stable algorithm without source term

$y = X_n$
for $k = 1, \ldots, n_c$ **do**
  $b = y - b_k \Delta t A y + a_k (\Delta t A)^2 y$
  Solve $(I - \gamma \Delta t A)^2 y = b$
end for
for $k = 1, \ldots, n_r$ **do**
  $b = y - \dfrac{\Delta t A}{\lambda_k} y$
  Solve $(I - \gamma \Delta t A) y = b$
end for
$X_{n+1} = y$

When the source term is added, we rewrite ϕ_n in the following form:

$$\phi_n = \Delta t \sum_{r=1}^{s+l} Q_{r-1}(\Delta t A) \sum_{i=0}^{s} \widetilde{\omega}_i^{r,l} F\left(t_n + \Delta t\, c_i\right), \tag{61}$$

where the polynomials Q_{r-1} are based on the factorization of the numerator $N_s^l(z)$

$$
\begin{aligned}
Q_0(z) &= \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_2z+a_2z^2\right)z,\\
Q_1(z) &= \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_2z+a_2z^2\right),\\
Q_2(z) &= \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_3z+a_3z^2\right)(1-\gamma z)^2z,\\
Q_3(z) &= \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_3z+a_3z^2\right)(1-\gamma z)^2,\\
&\cdots\\
Q_{s+l-2}(z) &= \left(1-\frac{z}{\lambda_{n_r}}\right)(1-\gamma z)^{s+l-1},\\
Q_{s+l-1}(z) &= (1-\gamma z)^{s+l}.
\end{aligned}
$$

Let us introduce:

$$G_r = \sum_{i=0}^{s} \widetilde{\omega}_i^{r,l}\, F\left(t_n + \Delta t\, c_i\right).$$

A stable algorithm taking into account the presence of the source term is given in Algorithm 2.

Algorithm 2 Stable algorithm with source term

$y = X_n$
for $k = 1, \ldots, n_c$ **do**
  $b = (\Delta t A)\left(a_k \Delta t A y + G_{2k-1} + b_k y\right) + G_{2k}$
  Solve $(I - \gamma \Delta t A)^2 y = b$
end for
for $k = 1, \ldots, n_r$ **do**
  $b = y - \dfrac{\Delta t A}{\lambda_k} y + G_{2n_c+k}$
  Solve $(I - \gamma \Delta t A) y = b$
end for
$X_{n+1} = y$

5 Numerical results for 1D and 2D wave equations

We are interested in solving the acoustic wave equation formulated as a first order system. 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:

$$
\begin{cases}
\rho\, \partial_t u - \operatorname{div} v = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\
\mu^{-1} \partial_t v - \nabla u = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\
u(x,0) = \partial_t u(x,0) = 0, \quad \forall x \in \Omega \quad \text{(null initial conditions)} \\
u = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\
\mu \partial_n u = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\
\mu \partial_n u + \sqrt{\rho\mu}\, \partial_t u = f_A, \quad x \in \Gamma_A \quad \text{(Absorbing condition)}
\end{cases}
\tag{62}
$$

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 outgoing normal 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.

5.1 Finite element discretization

5.1.1 Finite element spaces and semi-discrete problem

The computational domain Ω is meshed with regular intervals in 1-D and quadrilaterals in 2-D. Each element is denoted by K_i:

$$\Omega = \bigcup K_i$$

The equation (62) is solved with mixed spectral elements (see [1]) with the following finite element spaces

$$u(t) \in U_h = \left\{ u \in H^1(\Omega) \text{ such that } u|_{K_i} \circ F_i \in \mathbb{Q}_r \right\},$$

$$v(t) \in V_h = \left\{ v \in (L^2(\Omega))^d \text{ such that } v|_{K_i} \circ F_i \in (\mathbb{Q}_r)^d \right\},$$

where d is the dimension, r is the order of approximation, and $\mathbb{Q}_r$ is the space of polynomials of degree lower or equal to r in each space variable. In 2-D, F_i is the map from the unit square $\hat{K}$ to the element K_i (see figure 6). Gauss-Lobatto points are used both for interpolation and quadrature

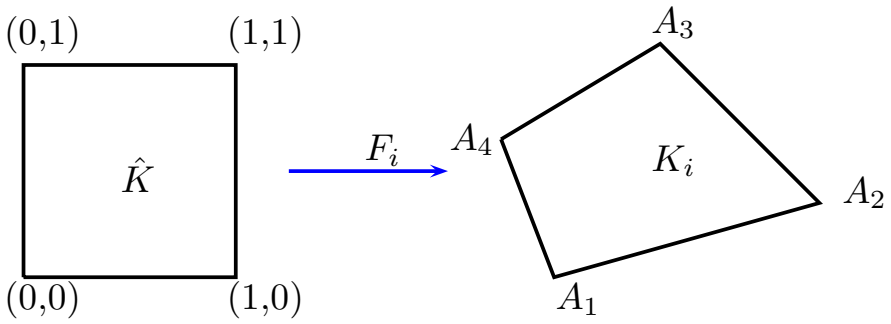


Figure 6: Transformation F_i for a quadrilateral.

formulas (see [1]), leading to a diagonal mass matrix. Let us denote by φ_i the basis functions for u and ψ_i the basis functions for v. The obtained semi-discrete system reads:

$$\begin{cases} D_h \dfrac{dU}{dt} + S_h U + R_h V = F_u(t) \\ B_h \dfrac{dV}{dt} - R_h^T U = F_v(t) \end{cases} \tag{63}$$

where h is the mesh size and T denotes the transpose matrix. We have the following finite element matrices:

$$(D_h)_{i,j} = \int_\Omega \varphi_i \varphi_j \, dx,$$

$$(S_h)_{i,j} = \int_{\Gamma_A} \sqrt{\rho\mu}\varphi_i \varphi_j \, dx,$$

$$(R_h)_{i,j} = \int_\Omega \nabla\varphi_i \cdot \psi_j \, dx,$$

$$(B_h)_{i,j} = \int_\Omega \psi_i \cdot \psi_j \, dx.$$

The matrices D_h, B_h and S_h are diagonal. The source term is given by:

$$F_U(t) = \int_{\Gamma_N} f_N \varphi \, dx + \int_{\Gamma_A} f_A \varphi \, dx.$$

The source term F_v comes from the inhomogeneous Dirichlet condition (if $f_D \neq 0$). Degrees of freedom associated with Dirichlet condition are not included in the vector $U(t)$, the associated values $f_D(x_i)$ provide a source vector F_v. The evolution system (63) falls in the class of ODEs (1) presented in the general setting.

5.1.2 Solution of linear systems

When using an implicit time-scheme, we need to solve systems of the form:

$$\begin{cases} \beta\, D_h U + S_h U + R_h V = F_1 \\ \beta\, B_h V - R_h^T U = F_2 \end{cases} \tag{64}$$

where β is a coefficient depending on the time scheme used. The unknown V is eliminated such as to obtain a symmetric linear system to solve for U:

$$\left(\beta\, D_h + S_h + \beta^{-1}\, R_h B_h^{-1} R_h^T\right) U = F_1 + \beta^{-1}\, R_h B_h^{-1} F_2. \tag{65}$$

The stiffness matrix K_h is given by

$$K_h = R_h B_h^{-1} R_h^T,$$

with entries

$$(K_h)_{i,j} = \int_\Omega \mu \nabla\varphi_i \cdot \nabla\varphi_j \, dx.$$

The system (65) is the same kind of linear system that appears when using θ-schemes for the second-order formulation of the wave equation. This system is symmetric positive definite if β is real positive. Moreover, the internal degrees of freedom are removed (process known as static condensation) to reduce the size of the final linear system. This final linear system is solved by using a tridiagonal solver in 1-D and MUMPS (see [25]) in 2-D.

5.1.3 PML layers (2-D)

In order to truncate the 2-D domain, instead of absorbing boundary conditions, we can use Perfectly Matched Layers (PML). In this paragraph, the efficient implementation of PML is detailed in order to obtain a reduced linear system to solve. More precisely, we consider the following split formulation

$$\begin{cases} \rho\, \partial_t u_1 \,+\, \sigma_x\, \rho\, u_1 - \partial_x v_x = 0 \\ \rho\, \partial_t u_2 \,+\, \sigma_y\, \rho\, u_2 - \partial_y v_y = 0 \\ \mu^{-1} \partial_t v \,+\, \sigma \mu^{-1} v - \nabla(u_1 + u_2) = 0 \\ \text{+ homogeneous Dirichlet condition on PML boundaries} \end{cases}$$

where $u = u_1 + u_2$ is the physical solution, u_1, u_2, v are intermediary unknowns. σ_x, σ_y are damping functions, non-null inside the PML, with a parabolic profile (see [26]):

$$\sigma_x = \frac{3\,\log 1000}{2a^3}(x - x_0)^2\, \sigma\, v_{\max} \quad \text{and} \quad \sigma_y = \frac{3\,\log 1000}{2a^3}(y - y_0)^2\, \sigma\, v_{\max},$$

where σ is a damping coefficient chosen *a priori*, $v_{\max}$ is the maximal wave velocity, x_0, y_0 are equal to $x_{\min}, y_{\min}, x_{\max},$ or $y_{\max}$ depending on the layer you are looking at. In order to have directly the physical field u as unknown, we write the problem with the two unknowns:

$$u = u_1 + u_2 \quad \text{and} \quad u^* = u_1 - u_2.$$

As a result, u, u^*, v are solutions to the following system

$$\begin{cases} \rho\, \partial_t u + \rho \left(\dfrac{\sigma_x + \sigma_y}{2}\right) u + \rho \left(\dfrac{\sigma_x - \sigma_y}{2}\right) u^* - \operatorname{div} v = 0 \\ \rho\, \partial_t u^* + \rho \left(\dfrac{\sigma_x + \sigma_y}{2}\right) u^* + \rho \left(\dfrac{\sigma_x - \sigma_y}{2}\right) u - (\partial_x v_x - \partial_y v_y) = 0 \\ \mu^{-1} \partial_t v + \mu^{-1} \sigma\, v - \nabla u = 0 \end{cases}$$

The unknown u^* is discretized only inside the PML and is equal to 0 on the external boundary and the interface between PML and physical domain. The finite element space for u^* is the same as u (except that it is reduced to PML region). We usually take $\sigma = 2$, in order to have a reflection

coefficient around 10^{-6}. The linear system to be solved in time is given by (we write here the equations without detailing the associated discrete formulation):

$$\rho\,\beta\,u + \rho\left(\frac{\sigma_x+\sigma_y}{2}\right)u + \rho\left(\frac{\sigma_x-\sigma_y}{2}\right)u^* - \operatorname{div} v = f, \tag{66a}$$

$$\rho\,\beta\,u^* + \rho\left(\frac{\sigma_x+\sigma_y}{2}\right)u^* + \rho\left(\frac{\sigma_x-\sigma_y}{2}\right)u - (\partial_x v_x - \partial_y v_y) = f^*, \tag{66b}$$

$$\mu^{-1}\beta\, v + \mu^{-1}\sigma\, v - \nabla u = f_v, \tag{66c}$$

u_1 and u_2 can be reconstructed:

$$\begin{cases} \rho\,\beta u_1 + \rho\,\sigma_x\,\rho\,u_1 - \partial_x v_x = \dfrac{f+f^*}{2} \\ \rho\,\beta u_2 + \rho\,\sigma_y\,u_2 - \partial_y v_y = \dfrac{f-f^*}{2} \\ \mu^{-1}\beta v + \sigma\mu^{-1} v - \nabla(u_1+u_2) = f_v \end{cases}$$

The first equation is multiplied by $\beta+\sigma_y$, the second one by $\beta+\sigma_x$ and the two equations are combined to obtain:

$$\rho\,(\beta+\sigma_x)\,(\beta+\sigma_y)u - \operatorname{div}\left(\begin{pmatrix} \beta+\sigma_y & 0 \\ 0 & \beta+\sigma_x \end{pmatrix} v\right) = (\beta+\sigma_y)\frac{f+f^*}{2} + (\beta+\sigma_x)\frac{f-f^*}{2}.$$

Finally v is eliminated and we obtain a single equation in u:

$$\rho\,(\beta+\sigma_x)\,(\beta+\sigma_y)u - \operatorname{div}\left[\begin{pmatrix} \mu\frac{\beta+\sigma_y}{\beta+\sigma_x} & 0 \\ 0 & \mu\frac{\beta+\sigma_x}{\beta+\sigma_y} \end{pmatrix}\nabla u\right] = \left[\beta + \frac{\sigma_x+\sigma_y}{2}\right] f + \frac{\sigma_y-\sigma_x}{2} f^*.$$

As a result a symmetric positive definite system needs to be solved for U:

$$(\widetilde{D}_h + \widetilde{K}_h)\,U = F_h,$$

where

$$(\widetilde{D}_h)_{i,j} = \int_\Omega \rho\,(\beta+\sigma_x)\,(\beta+\sigma_y)\varphi_i\varphi_j\,dx,$$

$$(\widetilde{K}_h)_{i,j} = \int_\Omega \begin{pmatrix} \mu\frac{\beta+\sigma_y}{\beta+\sigma_x} & 0 \\ 0 & \mu\frac{\beta+\sigma_x}{\beta+\sigma_y} \end{pmatrix}\nabla\varphi_i\cdot\nabla\varphi_j\,dx.$$

As a result, the presence of PML does not increase a lot the computational burden by performing this procedure, since the linear system to be solved does not involve the intermediary unknowns u^* or v. The intermediary unknowns u^* and v are reconstructed thanks to equations (66b) and (66c)

5.2 Convergence curves and numerical results in 1-D

The wave equation (62) is solved in 1-D in a homogeneous medium $\rho=\mu=1$ in the computational domain $\Omega=[0,500]$. An inhomogeneous Dirichlet condition is set on the left extremity

$$u(x=0,t) = e^{-i\omega t}\,\exp\left(-\frac{1}{2}\left(\frac{t-T}{\tau}\right)^2\right),$$

where

$$\omega = 2\pi, \quad \tau = \frac{20}{2\sqrt{2\log 2}}, \quad T = 100,$$

and a homogeneous Neumann condition is set on the right extremity. In space, mixed spectral elements of order 16 are used. The computational domain Ω is subdivided into 500 regular subintervals. As a result, we have 16 points per wavelength, which is rather high, but with this choice, the space discretization error is about 10^{-12}. We choose $[0,1000]$ for the time interval. For

this case, we can compare the numerical results with the following analytical solution (before reflection)

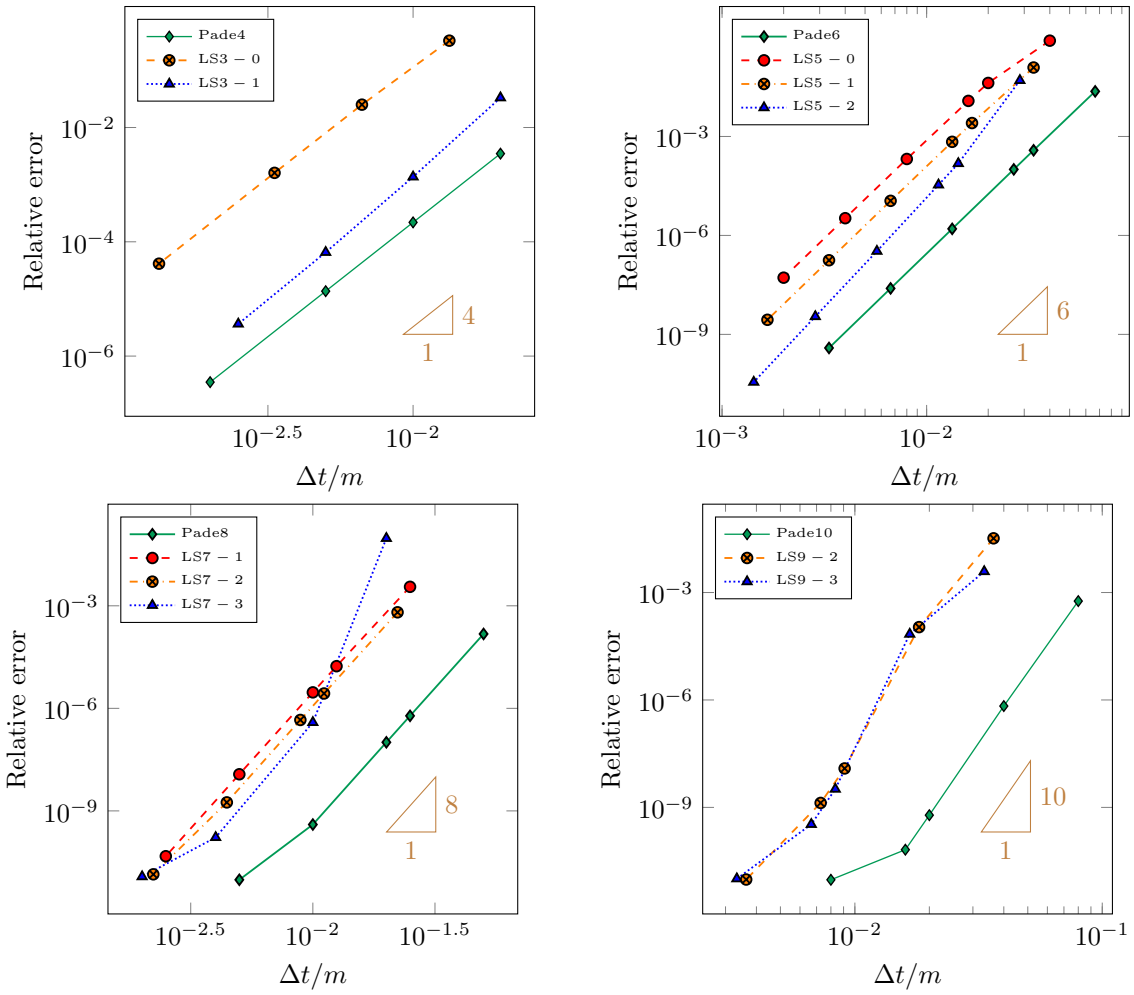
$$u^{\text{exact}}(x,t) = e^{i\omega(x-t)} \exp\left(-\frac{1}{2}\left(\frac{t-T-x}{\tau}\right)^2\right).$$

After the first reflection, the solution u will be conjugated.

In Figure 7, we present the relative L^2 error between the exact solution and the numerical solution (obtained with diagonal Padé schemes and Linear-SDIRK schemes of order 4, 6 and 8) at $t = 200$. In the $x-$coordinate we have chosen to represent $\frac{\Delta t}{m}$ where Δt is the time step and m is the number of linear systems we need to solve at each time step for each scheme (see Section 3 for Padé schemes and 4 for the Linear-SDIRK schemes). The advantage of this choice is that for a given $\frac{\Delta t}{m}$, the complexity of the different time schemes is the same.

The obtained convergence curves (Figure 7) show that the diagonal Padé schemes are more efficient than the Linear-SDIRK of the same order. These curves confirm the results we have obtained for the dispersion and dissipation curves as shown in Figures 2, 3 and 4. For the Linear-SDIRK of the same order, we can also see that the best ones are the LS$3-1$, LS$5-2$ and LS$7-3$, which was also noticeable in the dispersion and dissipation curves.

Figure 7: Relative L^2 error between numerical solution and exact solution for $t = 200$ versus the time step. Comparison of diagonal Padé and Linear-SDIRK of order 4, 6 and 8. LS$s-l$ represent the s plus l additional stages Linear-SDIRK of order s+1. The space discretization error is about 10^{-12}.



To evaluate the efficiency of the schemes regarding the computational times, we have chosen the best Linear-SDIRK schemes of order 4, 6 and 8. We aimed to one percent (1%) of relative L^2

error which is computed at $t = 1000$ between the numerical solution and the analytical solution. We present the computational times needed for the Linear-SDIRK schemes and the diagonal Padé in Table 5 to reach this error. The results we obtained confirm that the diagonal Padé schemes are more efficient than the Linear-SDIRK schemes in 1-D.

Table 5: Computational time after imposing 1% of relative errors (1-D case)

	Pade4	LS$3-0$	LS$3-1$	Pade6	LS$5-0$	LS$5-1$	LS$5-2$
Number of time steps	33333	110000	25960	8360	18835	11540	7355
Computational time	1mn36	5mn23	1mn48s	37s	1mn31	1mn09	53s
	Pade8	LS$7-1$	LS$7-2$	LS$7-3$	Pade10	LS$9-2$	LS$9-3$
Number of time steps	3875	5830	4600	3700	2326	3106	2845
Computational time	24s	46s	43s	38s	17s	34s	34s

5.3 Numerical results in 2D

5.3.1 Results with an absorbing boundary condition

In this paragraph, we consider the scattering of a resonant cavity. The computational domain Ω is meshed with quadrilaterals (see Figure 8). The external boundary is a circle of radius 1.5. The

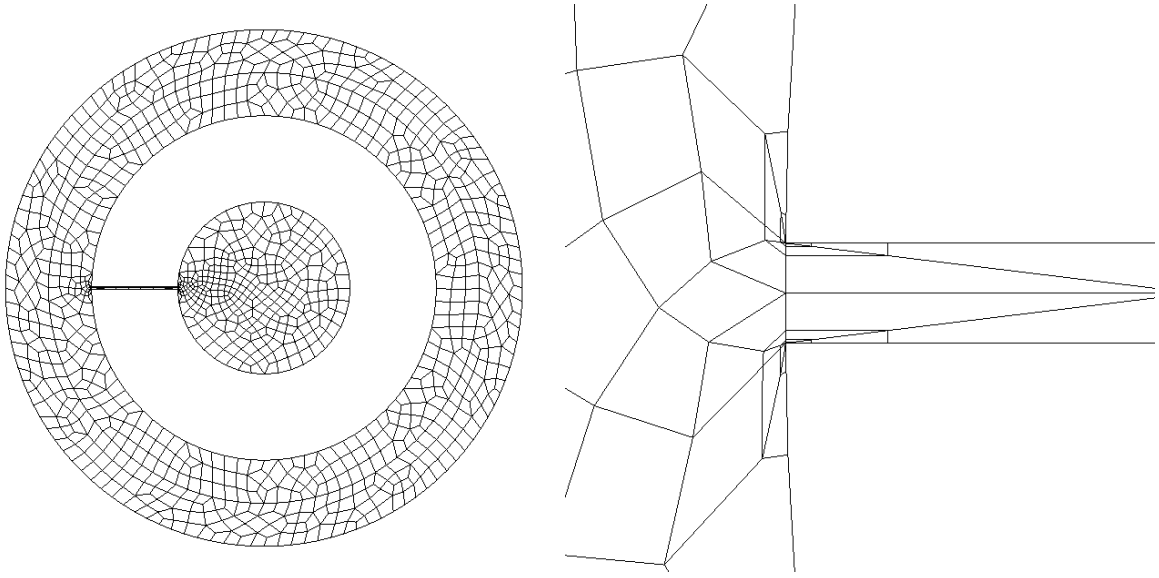


Figure 8: Mesh used for the resonant cavity. At right, detail of the mesh close to the entry of the cavity.

internal boundary is a circle of radius 1, the cavity is a circle of radius 0.5. The circular cavity is linked with the exterior domain by a rectangle of thickness $\sin\left(\frac{\pi}{180}\right)$. We have chosen the following physical parameters

$$\rho = \begin{cases} 0.8 \text{ if } \sqrt{x^2+y^2} \leq 1.25 \\ 1.0 \text{ otherwise} \end{cases}, \quad \mu = \begin{cases} 1.2 \text{ if } \sqrt{x^2+y^2} \leq 1.25 \\ 1.0 \text{ otherwise} \end{cases}.$$

A homogeneous Neumann boundary condition is set on all the boundaries except at the external circle. On this circle of radius 1.5, an inhomogeneous absorbing boundary condition is set (corresponding to the scattering by a plane wave):

$$\mu \partial_n u + \sqrt{\rho\mu}\, \partial_t u = \mu \partial_n u^{\text{inc}} + \sqrt{\rho\mu}\, \partial_t u^{\text{inc}}, \quad \text{on } \Gamma_A,$$

where the incident field is given by

$$u^{\text{inc}} = h(t - 1.5 - x),$$

where

$$h(t) = \sin(\omega t)e^{-b(t-T)^2} \quad \text{with} \quad \omega = 16\pi,\ b = 4,\ T = \sqrt{\log(10^6)}.$$

The solution is computed with real numbers (contrary to 1-D results), for a time interval $[0, 10]$. The solution is displayed in Figure 9 for $t = 2$ and $t = 10$. The mesh is locally refined close to the

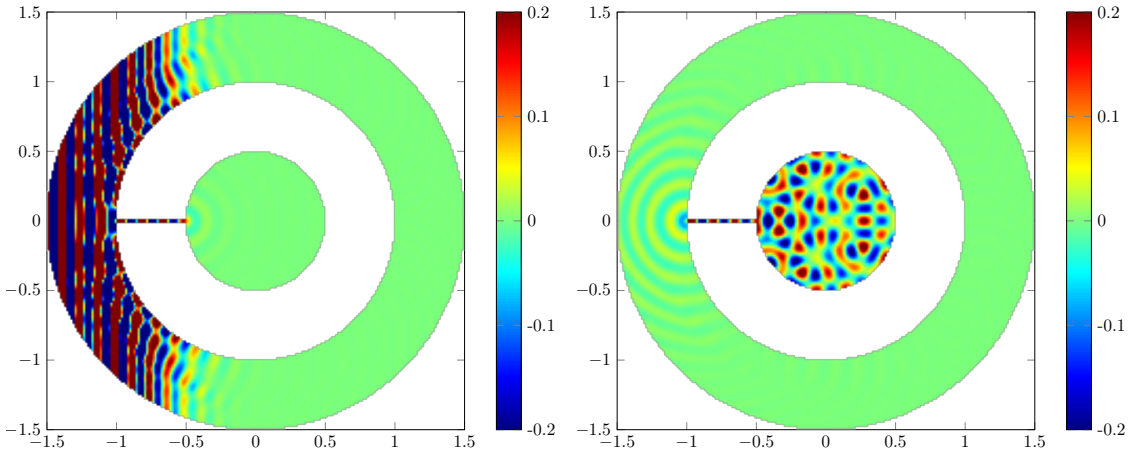


Figure 9: Solution obtained for the scattering of a resonant cavity for $t = 2$ and $t = 10$.

points where the solution possesses a singularity (see Figure 8) with five levels of refinement and a ratio 4. We are using $\mathbb{Q}_{10}$ finite elements (as detailed previously) such that the error due to the space discretization is below 10^{-6}. The reference solution is computed on this mesh with $\Delta t = 0.01$ and eighth-order Padé scheme. By modifying only the time step Δt (the mesh is always the mesh of Figure 8), we aimed at reaching a relative L^2 error (compared to the reference solution) below 0.1% for the final time $t = 10$. In Table 6, the number of time iterations and the computational time needed to reach this error are given.

Table 6: Computational time after imposing 0.1% of L^2 relative error for the scattering of a resonant cavity.

	Pade4	LS3−0	LS3−1	Pade6	LS5−0	LS5−1	LS5−2
Number of time steps	2370	7745	1955	623	1393	856	823
Computational time	5mn25	21mn17	7mn27	2mn16	6mn27	4mn49	5mn26
	Pade8	LS7−1	LS7−2	LS7−3	Pade10	LS9−2	LS9−3
Number of time steps	297	449	370	911	181	478	481
Computational time	1mn33	3mn23	3mn13	8mn40	1mn19	5mn05	5mn37

Padé schemes are clearly more efficient for this case. We have observed that LS7 − 1 is more efficient than LS7 − 3. This is due to the fact that the source term is not treated in an optimal fashion. We think that by choosing appropriately the coefficients $\alpha_i^{r,l}$ and the points c_i (free parameters introduced in the Section 4.6), it might be possible to recover a good behavior. To confirm this observation, we have set a homogeneous absorbing boundary condition and the following initial condition (instead of 0):

$$u = \exp\left(-7\frac{(x^2 + y^2)}{0.15^2}\right).$$

For the initial condition we find that LS7 − 3 is more efficient than LS7 − 1 as expected. The comparison results of Linear-SDIRK schemes and Padé schemes are represented in Table 7. We see that Linear-SDIRK schemes perform well, but they are less efficient than Padé schemes.

Table 7: Computational time after imposing 0.1% of L^2 relative error with an initial condition.

	Pade4	LS3−1	Pade6	LS5−0	LS5−1	LS5−2
Number of time steps	2500	2059	738	1642	1007	679
Computational time	5mn31	7mn30	2mn37	7mn38	5mn41	4mn28

	Pade8	LS7−1	LS7−2	LS7−3	Pade10	LS9−2	LS9−3
Number of time steps	381	560	442	365	248	316	298
Computational time	1mn53	4mn18	3mn49	3mn29	1mn36s	3mn10	3mn17

5.3.2 Results with PML

In this paragraph, we show some results when the mesh includes PML layers. The physical domain is the rectangle $[-2.8, 2.8] \times [-3.5, 3.5]$ which is supplemented by PML layers of thickness 0.3 (for $x > 2.8$, $x < -2.8$ and $y < -3.5$). An array of circular inclusions (disks of diameter 0.04, represented in green in the Figure 10) is considered. Two consecutive inclusions are separated by

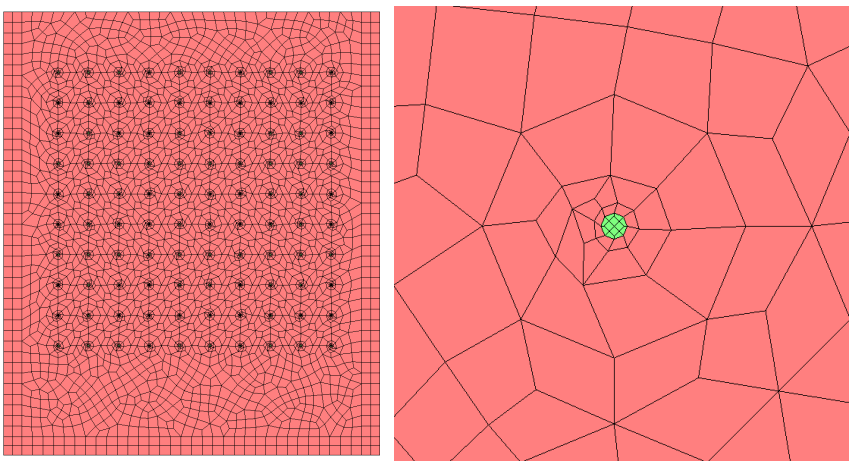


Figure 10: Mesh used for the array of inclusions. At right, detail of the mesh close to an inclusion (in green).

a distance of 0.5. We take the following physical parameters:

$$\rho = \begin{cases} 4.0 & \text{if inside an inclusion} \\ 1.0 & \text{otherwise} \end{cases}, \quad \mu = \begin{cases} 0.8 & \text{if inside an inclusion} \\ 1.0 & \text{otherwise} \end{cases}.$$

On the top boundary, an inhomogeneous Neumann condition is set

$$\partial_n u = \sqrt{\frac{\alpha}{\pi}}\, e^{-\alpha x^2}\, \sin(\omega t)\, e^{-b(t-T)^2}, \quad \text{for } y = 3.5,$$

where

$$\alpha = \frac{\log(10^6)}{1.8^2}, \quad \omega = 10\pi, \quad b = 2, \quad T = \sqrt{2\log(10^6)}.$$

The solution for $t = 3$ and $t = 16$ is plotted in the Figure 11. We are using $\mathbb{Q}_{10}$ finite elements (as detailed previously) on the mesh of the Figure 10 such that the error due to the space discretization

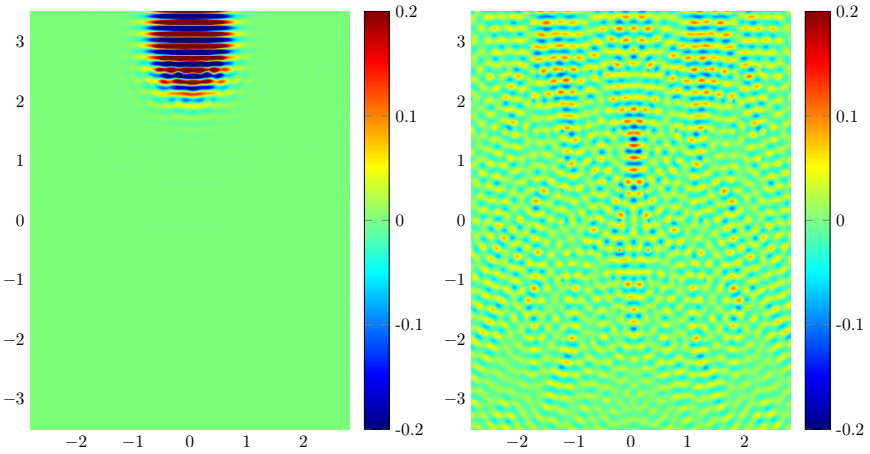


Figure 11: Solution obtained for the scattering of circular inclusions for $t = 3$ and $t = 16$.

is below 10^{-6}. The reference solution is computed on this mesh with $\Delta t = 0.01$ and the eighth-order Padé scheme. We aimed at reaching a relative L^2 error (compared to this reference solution) below 1% for the final time $t = 10$. In the Table 8, the number of time iterations and the computational time needed to reach this error are given.

Table 8: Computational time after imposing 1% of relative L^2 error for the scattering of inclusions

	Pade4	LS$3-1$	Pade6	LS$5-0$	LS$5-1$	LS$5-2$
Number of time steps	1410	1215	438	968	594	524
Computational time	16mn32	28mn51	10mn47	30mn53	20mn30	20mn57

	Pade8	LS$7-1$	LS$7-2$	LS$7-3$	Pade10	LS$9-2$	LS$9-3$
Number of time steps	226	336	280	447	145	295	251
Computational time	7mn18	15mn20	14mn38	25mn24	4mn59	15mn21	14mn34

The tenth-order Padé scheme is the most efficient for this case. We observe also that LS$7-3$ is not efficient. When we look at the numerical error, we have observed that it involves high-frequency modes whereas the numerical error for Padé schemes involves the "usual" modes.

6 Conclusion

In this paper, we have investigated two different classes of one-step schemes satisfying the A-stability property which are Padé schemes and Linear-SDIRK schemes. For both types of schemes, we have provided a description of the construction at any order and more importantly how to handle the source term. For Linear-SDIRK schemes, we have computed the coefficients γ and α_i until order 12. An implementation in Python of these schemes is proposed in the provided files quadrature.py and linear_scheme.py. These files can be downloaded at https://www.math.u-bordeaux.fr/~durufle/codes.php. The implementation includes stable algorithms that are robust with respect to round-off errors. This property is important for high-order schemes when the linear discrete operator has large eigenvalues (which is usually the case when implicit methods are used). Dispersion and dissipation analyses show that Padé schemes are more efficient than Linear-SDIRK schemes. Moreover, they are more robust when the system includes a source term. Concerning Linear-SDIRK, extra-stages are usually beneficial. For a homogeneous linear differential equation (without source term), LS$3-1$, LS$5-2$, LS$7-3$ and LS$9-2$ are more efficient. We think that for higher orders (such as tenth order schemes), the optimization should

take into account the set of coefficients α_i and not only α_1 to recover the advantage of extra-stages. Numerical experiments show that LS$3-0$ is much less accurate than LS$3-1$, it explains why several works have proposed low dispersive third and fourth order DIRK schemes (e.g. [13], [14], [27]). For the inhomogeneous case (with a source term), it turns out that LS$3-1$, LS$\,5-1$, LS$\,7-2$ and LS$\,9-2$ are more efficient than other Linear-SDIRK schemes.

We think that Linear-SDIRK schemes may be improved in the inhomogeneous case by choosing appropriate interpolation points c_i or coefficients $\alpha_i^{r,l}$. Another prospect is to perform the optimization of Linear-SDIRK schemes with another objective than minimizing $|\eta|$. We can think about finding the best approximation of the exponential on a given interval of the imaginary axis, or minimizing the dispersion error for example. Our next objective is to construct locally implicit schemes by using the implicit schemes developed here.

Acknowledgements

This work has been supported by the Inria-TOTAL strategic action DIP (dip.inria.fr) and the Conseil Départemental des Pyrénées Atlantiques and has received funding from the European Union's Horizon 2020, research and innovation programme under the Marie Sklodowska-Curie grant agreement N 644202. It takes part to the research program aknowledged by the competitiveness Pole Avenia http://www.pole-avenia.com/eng/. Experiments presented in this work were carried out using the PlaFRIM experimental platform.

References

[1] G. Cohen and S. Fauqueux, "Mixed finite elements with mass-lumping for the transient wave equation", *Journal of Computational Acoustics*, vol 8, pp 171–188, 2000

[2] G. Cohen and S. Pernet, "Finite element and discontinuous Galerkin methods for transient wave equations", *Springer*, 2017

[3] T. Warburton and T. Hagstrom, "Taming the CFL Number for Discontinuous Galerkin Methods on Structured Meshes", *SIAM J. Numer. Anal.*, vol 46(6), pp 3151–3180, 2008

[4] M. Mehlin, T. Mitkova and M. Grote, "Runge-Kutta-based explicit local time-stepping methods for wave propagation", *SIAM J. on Scientific Computing*, vol 37(2), 2015

[5] S. Piperno, "Symplectic Local Time-Stepping in non-dissipative DGTD Methods Applied to Wave Propagation Problems", *ESAIM: Mathematical Modelling and Numerical Analysis*, vol 40(5), pp 815–841, 2006

[6] E. Hairer and G. Wanner, "Solving ordinary differential equations II Stiff and differential-algebraic problem", *Springer*, 2010

[7] A. Kanevsky, M.H. Carpenter, D. Gottlieb and J.S. Hesthaven, "Application of implicit-explicit high order Runge-Kutta methods to discontinuous Galerkin schemes", *J. Comput. Phys.*, vol 225, pp 1753–1781, 2007

[8] S. Descombes, S. Lanteri and L. Moya, "Locally implicit time integration strategies in a discontinuous Galerkin method for Maxwell's equations", Research Report INRIA 7983, 2012

[9] J. Chabassier and S. Imperiale, "Fourth-order energy-preserving locally implicit time discretization for linear wave equations", *Int. J. Numer. Meth. Engng*, vol 106, pp 593–622, 2016

[10] G. Wanner, "Dahlquist's classical papers on stability theory", *BIT Numerical Mathematics*, 2006

[11] J. R. Cash, "On the exponential fitting of composite, multiderivative linear multistep methods", *SIAM J. Numer. Anal*, 1981

[12] R. Alexander, "Diagonally implicit Runge-Kutta methods for stiff O.D.E.'s", *SIAM J. Numer Anal.*, vol 14, 1977

[13] L.M. Skvortsov, "Diagonally Implicit Runge-Kutta Methods for Stiff Problems", *Computational Mathematics and Mathematical Physics*, vol 46(12), pp 2110–2123, 2006

[14] P. J. Van Der Houwen And B. P. Sommeijer, "Phase-lag analysis of implicit Runge-Kutta methods", *SIAM J. Numer. Anal.*, vol 26, pp 214–229, 1989

[15] B. L. Ehle, "A-Stable methods and Padé approximations to the exponential", *SIAM J. Numer. Anal.*, 1973

[16] O. Axelsson and A. B. Kucherov, "Real valued iterative methods for solving complex linear systems", *Department of Mathematics, University of Nijmegen*, 1999

[17] K. Burrage, "A special family of Runge-Kutta methods for solving stiff differential equations", *BIT*, pp 22–41, 1977

[18] M. Duruflé, "Intégration numérique et éléments finis d'ordre élevé appliqués aux équations de Maxwell en régime harmonique", *Université Paris Dauphine*, 2006

Read More: http://montjoie.gforge.inria.fr/

[19] C. K. W. Tam and J. C. Webb, "Dispersion-Relation-Preserving Finite Difference Schemes for Computational Acoustics", *Computational Physics*, vol 107, pp 262–281, 1993

[20] J. C. Butcher, "Numerical Methods for Ordinary Differential Equations Second Edition", *John Willey & Sons Ltd*, 2008

[21] J. Wimp, "On the zeros of a convluent hypergeometric function", *American Mathematical Society*, 1965

[22] E. B. Saff and R. S. Varga, "On the zeros and poles of Padé approximants to e^z. III", *Numer. Math.*, vol 30, pp 241–266, 1978

[23] B. L. Ehle, "On Padé approximations to the exponential function and A-stable methods for the Numerical solution of initial value problems", *University of Waterloo*, 1969

[24] E. Gallopoulos and Y. Saad, “On the parallel solution of parabolic equations”, *Research Institute for Advanced Computer Science NASA Ames Research Center*, RIACS Technical Report 89.19, 1989

[25] P.R. Amestoy and I.S. Duff and J. Koster and J.-Y. L'Excellent, “A fully asynchronous multifrontal solver using distributed dynamic scheduling”, *SIAM Journal on Matrix Analysis and Applications*, vol 23(1), pp 15–41, 2001

[26] Francis Collino and Chrysoula Tsogka, “Application of the Perfectly Matched Absorbing Layer Model to the Linear Elastodynamic Problem in Anisotropic Heterogeneous Media” *Geophysics*, vol 66, pp 294–307, 2001

[27] A. Najafi-Yazdi and L. Mongeau, “A Low-dispersion and Low-dissipation implicit Runge-Kutta scheme”, *J. Comput. Phys.*, vol 233, pp 315-323, 2013