# On the study and development of high-order time integration schemes for odes applied to acoustic and electromagnetic wave propagation problems

Mamadou N'Diaye

ACADÉMIE DE BORDEAUX |--|--|--|--|--|--|--|--|--|--|

January 01, 2015 - December 31, 2017

# Ph.D. Thesis

Presented by


## Mamadou N'DIAYE

UNIVERSITÉ DE PAU ET DES PAYS DE L'ADOUR
LABORATOIRE DE MATHÉMATIQUES ET DE LEURS APPLICATIONS UMR CNRS 5142
ÉQUIPE-PROJET MAGIQUE-3D INRIA-UPPA-CNRS

ÉCOLE DOCTORALE DES SCIENCES ET LEURS APPLICATIONS - ED 211


# ON THE STUDY AND DEVELOPMENT OF HIGH-ORDER TIME INTEGRATION SCHEMES FOR ODES APPLIED TO ACOUSTIC AND ELECTROMAGNETIC WAVE PROPAGATION PROBLEMS

December 8, 2017, 10:30 am

## Defense committee

| | | |
|---|---|---|
| Hélène BARUCQ | Senior Researcher INRIA Bordeaux Sud-Ouest | Advisor |
| Henri CALANDRA | Expert Numerical Methods and HPC, TOTAL SA | Examiner |
| Stéphane DESCOMBES | Professor Université Nice Sophia Antipolis | Reviewer |
| Julien DIAZ | Junior Researcher INRIA Bordeaux Sud-Ouest | Examiner |
| Marc DURUFLÉ | Assistant Professor ENSEIRB-MATMECA | Co-advisor |
| Jéronìmo RODRIGUEZ | Professor Universidad de Santiago de Compostela | Reviewer |
| François-Xavier ROUX | Professor Université Paris 6, UPMC | President |

I y'a mɛn Jɔn Kunna di fo Sababu Ɲuman.

À la famille, amis et collègues.

Avant tout je voudrais remercier mes directeurs de thèse, Hélène Barucq et Marc Duruflé, pour l'encadrement dont j'ai bénéficié ainsi que pour leur disponibilité. Je suis très heureux d'avoir pu travailler avec eux et profiter de leur expertise scientifique et de leurs qualités humaines.

Je souhaite ensuite remercier l'ensemble des membres de mon jury de thèse d'avoir accepté d'assister à ma soutenance. Je remercie tout particulièrement mes rapporteurs de thèse, Stéphane Descombes et Jéronìmo Rodriguez, pour leur appréciation et leurs commentaires pertinents sur ce manuscrit. Je remercie également François-Xavier Roux d'avoir présidé le jury ainsi qu'Henri Calandra et Julien Diaz pour leur soutiens et encouragements tout au long de cette thèse.

Je remercie aussi Inria et CG64 [1] pour le financement de cette thèse.

Je souhaite exprimer mes sincères remerciements à l'ensemble des personnes qui, de près ou de loin, ont contribué à la réussite de mes travaux.
Toi que je connais, toi que j'ai croisé, toi avec qui j'ai échangé un regard, toi avec qui j'ai partagé un sourire, toi avec qui j'ai partagé un repas, toi avec qui j'ai échangé un mot, je te remercie. Je te remercie toi qui lis ces mots, toi qui as travaillé si dur pour que ces jours soient.

A vous tous (amis, collègues, famille ...) merci, c'est chacun d'entre vous qui fait ce que je suis.

[1] This work has been supported by the Inria-TOTAL strategic action DIP (dip.inria.fr) and the Conseil Départemental des Pyrénées Atlantiques. It takes part to the research program acknowledged by the competitiveness Pole Avenia http://www.pole-avenia.com/eng/.

# Contents

# List of Figures

# List of Tables

Chapter 1

# Introduction

The solution of wave propagation problems in electromagnetism, 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. In the context of this thesis we are mainly concerned with the solution of acoustic and electromagnetic wave equations even though its content is fully applicable in other areas.

High-order finite element methods (FEM) have now demonstrated their strong capability for solving wave equations (for example [23, 24, 43, 57, 66]). 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) [38] or Perfectly Matched Layers (PML [7, 24]). 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 [23]). After space discretization, the obtained ODE can be discretized either with explicit ([12, 30, 40, 45, 46, 64] ...) or implicit time schemes ([50, 60, 62] ...). Explicit time schemes ([41]) 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 [41]) so small that the computational cost becomes prohibitive.

A nice work has been done in [77] 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 local time-stepping techniques ([30, 55, 67, 68]) in order to have globally explicit schemes and a reduced computational cost with only few elements having a small time-step. In fact, using a tiny time-step in the all domain may increase the dispersion errors where the domain contain bigger elements [68] and the global computational time increases as well. Another approach consists in applying globally implicit time-stepping techniques with unconditionally stability (Dahlquist's A-stable property [42]). In that case, there is no time-step restriction regarding the stability. 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 ([16, 28, 47]) 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, part of this work is mainly focused on the construction of high-order one-step methods that are A-stable following the definition introduced by Dahlquist [42] that we recall in the Chapter 2. Linear multi-step schemes have not been considered because A-stable linear multi-step schemes are at most second-order schemes. This fact is known as the second Dahlquist's barrier (see [75]). Nevertheless, we have found some interesting investigations on multi-derivative multi-step schemes in [15] in which the author proposes schemes up to order 5.

There are two main classes of implicit Runge-Kutta which are A-stable: Gauss Runge-Kutta (Gauss-RK) schemes ([42]) that can be written at any order and some Singly Diagonally Implicit Runge Kutta (SDIRK) schemes that have been constructed up to order 5 ([1, 42, 44, 71]). Both schemes Gauss-RK and SDIRK can be used to solve linear and non-linear system. 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 Differ-

ential Equations (ODE) of the form

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

where $A$ is a linear operator. The analytical solution to this ODE involves an exponential matrix that can be approximated [56] to construct different numerical schemes. This kind of ODE is usually obtained after discretizing partial differential equations that model wave propagation problems. Space discretization methods are presented in the Chapter 3. We consider one-step schemes adapted to this class of ODE. In the Chapter 4 we study both explicit and implicit schemes of the familly of Runge-Kutta schemes. We point out the main drawback of these classical Runge-Kutta schemes when they are used to solve wave propagation problems. From these analyses, we investigate in the Chapter 5 the construction of schemes based on the diagonal Padé approximant of exponential (see [34]). In [4], these schemes are detailed for fourth-order. In this work, we have written them for an arbitrary 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 $m \times N$ as done for 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 presented in the Chapter 4, we have developed in the Chapter 6 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 [13]). By adding extra-stages, we construct Linear-SDIRK up to order 12. It is possible to construct higher-order schemes by adding more extra-stages. These schemes seem attractive when the memory is a critical issue because they are less memory consuming compared to Padé schemes when a direct solver is used (see the chapters 4, 5 and 6). However, using implicit schemes only to solve 3D realistic problems is still a big challenge since they require a large resource memory compared to explicit schemes.

A good compromise can be found when we can take advantage of the good stability properties of implicit schemes and the less memory consuming process of explicit schemes. That is why it might be interesting to consider locally implicit time schemes in which implicit schemes are used only in a small part of the computational domain (e.g. refined elements or elements where we use high order) and explicit schemes are used in the remaining part of the domain.

With the aim of developing locally implicit schemes, we have studied and developed in Chapter 7 explicit schemes that are obtained by maximizing the CFL number. Those schemes are also developed exclusively for linear ODEs. Then in Chapter 8 we present a method inspired from the work in [55] that combine the explicit and implicit schemes developed in Chapters 5, 6 and 7 to

form a locally implicit time scheme for linear ODEs. It is worth noting that for each scheme that has been developed in this work, we have performed an analysis of dispersion and dissipation with the aim of providing comparisons of the different approaches.

All the schemes presented in this thesis have been implemented in the high-order finite elements C++ code Montjoie [31]. Plus the validation test in the different and numerical results provided for 1D and 2D test cases for each scheme, the last Chapter provides numerical results when solving 3D acoustic and electromagnetic wave equations.

Beginning with an overview to situate the work, the remainder of this thesis is divided into three major divisions. In the first part we recall definitions and properties for the time integration scheme, we present the wave equations considered and the space discretization methods used. The second part focusses on the development of time schemes for linear ODEs and the introduction of locally implicit time schemes. In the last part, we present numerical results obtained when solving 3D acoustic and electromagnetic wave propagation problems.

# Introduction Générale

La résolution des problèmes de propagation d'ondes électromagnétiques, acoustiques et élastiques a de plus en plus d'applications dans de nombreux domaines de l'ingénierie et de la science tels que la géophysique (imagerie sismique), la médecine (imagerie médicale), l'aérospatiale (radar) et les télécommunications. Cette large gamme d'applications a conduit au développement de nombreuses techniques de calcul pour résoudre les équations aux dérivées partielles (EDP) régissant les problèmes de propagation d'ondes. Dans le cadre de cette thèse, nous nous intéressons principalement à la résolution d'équations d'ondes acoustiques et électromagnétiques, même si le contenu est pleinement applicable dans d'autres domaines.

Les méthodes d'éléments finis (FEM) d'ordre élevé sont connus maintenant pour être bien adaptées pour prendre en compte des géométries complexes et des milieux hétérogènes lorsque l'on résout les problèmes d'ondes ([23], [24]). La mise en œuvre de ces méthodes nécessite d'introduire des frontières artificielles. Ces frontières peuvent être modélisées par une condition de bord absorbante (ABC) [38] ou des couches absorbantes parfaitement adaptées (connu sous son acronyme anglais PML [7],[24]). En pratique, les conditions de bord ABC ou les PML sont plus faciles à gérer lorsque l'équation d'onde est formulée comme un système de premier ordre (en espace et en temps) comme nous le considérons dans ce manuscrit (comme dans [23]). Après la discrétisation en espace du problème d'onde par une méthode FEM, l'Équation Différentielle Ordinaire (EDO) obtenue peut être discrétisée avec des schémas explicites ([12, 30, 40, 45, 46, 64]) ou des schémas implicites en temps ([50, 60, 62]). Les schémas explicites ([41]) sont très populaires, car ils génèrent des algorithmes plus faciles à mettre en œuvre et utilisent très peu de ressources mémoire. Cependant, pour des raisons de stabilité, le pas de temps utilisé pour la discrétisation temporelle est limité par la taille du plus petit élément du maillage et par le degré des polynômes utilisés lors de la discrétisation spatiale par une méthode FEM. En conséquence, la présence de quelques petits éléments dans un maillage peuvent rendre la valeur maximale du pas de temps (connue sous le nom de la condition de Courant-Friedrichs-Lewy ou CFL [41]) si petite que le coût de calcul devient

prohibitif.

Pour augmenter la valeur du nombre CFL, [77] propose une technique pour des méthodes de discrétisations spatiales d'ordre élevé. Cette technique consiste à appliquer une discrétisation spécifique en espace de telle sorte que les valeurs propres de l'opérateur discret de l'EDO obtenu sont modifiées et conduit directement à un nombre CFL qui est maximal. D'autres travaux proposent la technique du pas de temps local ([30, 55, 67, 68]). Cela consiste à utiliser un pas de temps plus petit sur les parties du maillage composées d'éléments très petits et un pas de temps plus grand sur les autres parties. Avec cette technique, le nombre CFL est local pour chaque partie et le temps de calcul global est réduit. Pour éviter d'avoir une contrainte sur le pas de temps vis à vis de la stabilité, une autre approche consiste à utiliser uniquement des schémas implicites qui sont inconditionellement stables (la propriété A-stable de Dahlquist [42]). Cette approche est particulièrement intéressante pour des simulations en 1D et 2D. Par contre sur des gros cas 3D dans un milieu hétérogène, il paraît très difficile d'utiliser cette approche car les systèmes à résoudre pour chaque pas de temps du schéma implicite sont trop grands. C'est pourquoi dans certains travaux récents [47, 28, 16] une nouvelle technique, consistant à appliquer un schéma implicite sur une partie du domaine de calcul et un schéma explicite sur le reste est adoptée pour former un schéma localement implicite. Pour obtenir une solution robuste et bien précise avec une telle approche, il est important d'utiliser des schémas explicites et implicites qui sont robustes. Dans le cadre des problèmes d'ondes la robustesse d'un schéma d'intégration temporel est aussi caractérisée par des effets de dispersion et dissipation liés à ce schéma. Dans ce travail, nous étudions et construisons des schémas implicites qui sont inconditionnellement stables et qui ont de faibles erreurs de dispersion et de dissipation. Ainsi, nous développons des méthodes numériques d'ordre élevé en temps pertinentes pour des applications pratiques impliquant des équations d'ondes. Actuellement ces types de schémas implicites temporels ne sont pas communs au-delà de l'ordre quatre dans la littérature.

Suivant la définition des schémas A-stables introduite par Dahlquist [42], que nous rappellerons dans le Chapitre 2, une grande partie de ce travail est consacrée au développement de schémas implicites A-stables à un pas. Nous considérons en particulier des schémas Implicites Runge-Kutta (IRK), ils sont à un pas et donc ne nécessitent pas de schéma d'initialisation comme c'est le cas pour les schémas à plusieurs pas. De plus, il faut noter que les schémas à pas multiple qui sont A-stables, sont généralement au plus d'ordre deux à cause de la barrière de Dahlquist [75]. Cependant, il est possible d'avoir des schémas A-stables à pas multiple d'ordre supérieur à deux en utilisant la technique combinant l'opérateur discret et ses dérivées tel que présenté dans [15]. Dans [15], ces schémas sont développés jusqu'à l'ordre cinq, mais seuls les schémas d'ordre deux à quatre sont A-stables. Pour ces raisons nous nous limiterons aux schémas à un pas.

Pour revenir aux schémas implicites de type Runge-Kutta qui sont A-stables, on distingue principalement deux classes : les schémas obtenus à partir de la méthode de quadrature de Gauss Legendre appelés schémas de Gauss Runge-Kutta (Gauss RK) et les schémas Runge-Kutta de type SDIRK (Singly Diagonally Implicit Runge-Kutta) [42]. Les schémas A-stables Gauss RK peuvent être obtenus pour n'importe quel ordre de précision. Les schémas A-stables SDIRK sont eux disponible uniquement jusqu'à l'ordre cinq [42, 1, 71, 44]. Les deux classes de schémas sont utilisables pour EDO non-linéaire. Cependant, à cause de la taille des systèmes à résoudre (qui augmente avec l'ordre), les schémas Gauss RK sont très coûteux en terme de ressource mémoire ce qui les rend inefficaces. Les schémas SDIRK nécessitent eux la résolution d'un petit système plusieurs fois, mais ils n'en existent pas après l'ordre cinq, qui sont A-stables. C'est pour ces raisons qu'une grande partie de cette thèse est consacrée au développement des schémas implicites d'ordre élevé qui sont A-stables.

Dans ce travail, nous considérons L'EDO suivante

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

où $A$ est un opérateur linéaire. Ce type d'EDO est généralement obtenu lorsque l'on discrétise en espace les équations aux dérivées partielles (EDP) comme c'est le cas pour les équations d'ondes que nous présentons au Chapitre 3 de ce document. La solution analytique de cette EDO fait intervenir une fonction exponentielle de matrice qui peut être approchée, comme dans [56], pour construire un schéma numérique d'intégration temporel.

Dans le Chapitre 4, nous étudions les schémas RK explicites et implicites. On y présente le domaine de stabilité, les erreurs de dissipation et dispersion des schémas RK qui sont couramment utilisés dans la littérature. De cette analyse, on propose dans le Chapitre 5 une méthode pour construire des schémas d'intégration en temps basés sur l'approximant de Padé d'une fonction exponentielle [34]. Dans [4], ce type de schéma est développé pour l'approximant de Padé d'ordre 4. Dans cette thèse, on les propose jusqu'à l'ordre $2m$, $m \in \mathbb{N}^*$. En plus de donner une formule générale permettant de traiter la fonction source pour des EDO avec terme source, on propose un algorithme permettant de réduire le nombre de système à résoudre. On remarquera que ces schémas, que nous appellerons schémas de Padé, sont équivalents aux schémas Gauss RK pour les EDO linéaire. L'avantage d'utiliser les schémas de Padé réside dans le fait qu'avec ces schémas on résout $m$ systèmes linéaire de taille $N$, à chaque pas de temps, au lieu d'un grand système de taille $m \times N$ avec Gauss RK ($m$ représente le nombre d'étapes et $N$ le nombre d'inconnus).

A défaut d'avoir des schémas SDIRK A-stables d'ordre supérieur à cinq, nous proposons une méthode permettant de les construire à un ordre arbitraire pour les EDO linéaires dans le Chapitre 6. On les appellera schémas Linear-SDIRK puisque par construction, ils sont utilisables uniquement pour des EDO linéaires. Ils seront construites en approchant la fonction exponentielle par une

fraction de polynôme ayant un seul pôle (comme initialement étudié dans [13]). En ajoutant des étapes additionnelles, on propose les schémas Linear-SDIRK jusqu'à l'ordre 12. Ces schémas implicites A-stables ont un intérêt pratique, car ils utilisent peu de ressources mémoires comparés aux schémas de Padé comme on le verra dans les Chapitres 4, 5 et 6. Cependant, pour des problèmes de grande taille (en 3D par exemple), l'utilisation des schémas implicites uniquement est très difficile car elle nécessite beaucoup de ressources mémoires ce qui n'est pas le cas avec les schémas explicites.

Nous avons donc des schémas implicites qui ont de bonnes propriétés de stabilités et des schémas explicites qui consomment eux moins de ressource mémoire. L'idée est maintenant de combiner les deux types de schémas pour avoir un schéma qui sera moins restrictif en terme de stabilité et de ressource mémoire. C'est pour cela que nous nous sommes intéressés aux schémas localement implicites [47, 28, 16]. Ce type de schéma est obtenu en utilisant un schéma implicite sur une petite partie du domaine de calcul (exemple : les parties raffinées du maillage ou les éléments où on utilise de l'ordre élevé) et un schéma explicite sur le reste du domaine.
Pour cela, nous avons développé dans le Chapitre 7, des schémas explicites en maximisant le nombre CFL pour un profil type de spectre. Ces schémas ont aussi été développés uniquement pour des EDO linéaires. Puis dans le Chapitre 8 nous présentons une méthode, inspirée de [55], combinant les schémas explicites et implicites développés dans les Chapitres 5, 6 et 7, pour proposer des schémas localement implicites.

Tous les schémas que nous avons construits durant cette thèse, ont été implémentés dans le code éléments finis d'ordre élevé Montjoie [31] développé en C++. Une implémentation python de ces schémas peut être téléchargé sur le site https://www.math.u-bordeaux.fr/ durufle/codes.php. En plus des tests de validation en 1D et 2D que nous avons fait dans chaque chapitre, nous présentons dans le dernier chapitre des résultats numériques dans la résolution des équations d'ondes acoustiques et électromagnétiques en 3D.

On notera qu'une représentation des erreurs de dispersion et de dissipation numérique des schémas explicites et implicites, qui ont été présentés dans ce manuscrit, est proposée dans les différents chapitres pour comparer les différentes approches.

La reste de ce manuscrit est divisé en trois parties. Dans la première partie, nous introduisons les définitions et propriétés utilisées pour construire les schémas en temps dans le cadre de cette thèse. Puis nous présentons les équations d'ondes et les méthodes de discrétisations en espace que nous utilisons. Cette première partie se termine par une revue des schémas d'intégrations en temps qui sont couramment utilisés dans la littérature. La deuxième partie, qui constitue l'apport principale de cette thèse, est concentrée sur la construction des schémas d'intégrations en temps pour les EDO linéaires. On y présente deux classes de schémas A-stables implicites en temps, une famille de schémas ex-

plicite construite en maximisant le nombre CFL et enfin une méthodologie pour construire des schémas localement implicites. Pour conclure, nous présentons dans la troisième partie des résultats numériques lorsqu'on résout des équations d'ondes acoustiques et électromagnétiques en 3D.

# Part I

# Definitions, space discretization and RK schemes

Chapter 2

# Preliminary definitions and properties

This chapter presents some properties and definitions related to the stability of the solution of time evolution problems. We introduce the definition of the dissipation and dispersion errors due to a given time integration scheme. The stiffness of time evolution problems is also presented.

## Contents

## 2.1 Linear system of Ordinary Differential Equations (ODE)

Let us consider the following system:

$$\frac{dy(t)}{dt} = A(t)y(t) + B(t),\ t \in I \subset \mathbb{R} \tag{2.1}$$

where $y(t) \in \mathbb{C}^m$ is the unknown. $A = (a_{i,j}(t))_{1\leq i,j\leq m}$, $B(t) \in \mathbb{C}^m$ are continuous functions defined in $I$. The function $f$ which associates $y$ to $A(t)y(t) + B(t)$ is Lipschitz and continuous on $\mathbb{C}^m$ with respect to $y$ for each value of $t \in I$. Hence, the Cauchy problem (2.1) with the initial condition $y(t_0) = y_0$ has a unique solution see [27]. In this work, we only address the case where $A$ has constant coefficients.

### *2.1.1 Matrix exponential*

The exponential matrix function appears in general to construct the analytical solution of the ODEs. Before introducing approximation methods to compute the exponential of a matrix [56], we define the matrix exponential when $A$ is diagonalizable. Then, there exists an invertible matrix $P$ such that $D = P^{-1}AP$ is a diagonal matrix. The diagonal matrix $D$ is defined by $D_{ij} = \lambda_i$ for $i = j$ and $D_{ij} = 0$ for $i \neq j$. The coefficients $\lambda_i$, $i = 1, \cdots, m$ are the eigenvalues of the matrix $A$. Then we have

$$\mathrm{e}^A = \mathrm{e}^{PDP^{-1}} = P\mathrm{e}^D P^{-1} = P \begin{pmatrix} \mathrm{e}^{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \mathrm{e}^{\lambda_m} \end{pmatrix} P^{-1}. \tag{2.2}$$

The equation (2.2) comes from the fact that $A^n = PD^nP^{-1}$. A more general case when $A$ is not diagonalizable is detailed in [27].

### *2.1.2 Solution of a linear differential system*

Let us consider that there is no source term, $B(t) = 0$. Then the solution of (2.1) satisfying the initial conditions $y(t_0) = y_0$ is given by

$$y(t) = \mathrm{e}^{(t-t_0)A}\, y_0. \tag{2.3}$$

In the general case where $B(t) \neq 0$ in (2.1), we can use the variation of parameters method to solve the inhomogeneous linear ODE. This is done as follows: first we look for a particular solution in the form

$$y(t) = \mathrm{e}^{(t-t_0)A} v(t)$$

where $v$ is differentiable. Then by differentiating $y(t)$ we get

$$y'(t) = A\,y(t) + \mathrm{e}^{(t-t_0)A} v'(t).$$

It suffices to chose $v$ such that $\mathrm{e}^{(t-t_0)A}\,v'(t) = B(t)$, we obtain

$$v(t) - v(t_0) = \int_{t_0}^{t} \mathrm{e}^{-(u-t_0)A}\,B(u)\,du, \quad t_0 \in I.$$

Knowing $v$, the particular solution of (2.1) that satisfies $y(t_0) = 0$ is given by

$$y(t) = \mathrm{e}^{(t-t_0)A} \int_{t_0}^{t} \mathrm{e}^{-(u-t_0)A}\,B(u)\,du. \tag{2.4}$$

Finally the general solution to the problem (2.1) such that $y(t_0) = y_0$ is then given by

$$y(t) = \mathrm{e}^{(t-t_0)\,A} y_0 + \int_{t_0}^{t} \mathrm{e}^{(t-u)A}\,B(u)\,du \tag{2.5}$$

## 2.2 Stability for the solution of time evolution problems

In this section, we investigate the stability behaviour of the solution to an ordinary differential equation (ODE) for all initial values in a neighbourhood of a certain equilibrium point. We consider linear ODEs and show that the stability of the solutions depends on the sign of the real part of the eigenvalues of the matrix associated to the linear equation.

### *2.2.1 Stable solution*

We recall the definition of a stable solution borrowed from the book [27]. We first give the definition of a maximal solution to an ODE in a given interval.

**Definition 2.2.1**

- *Let $y : I \to \mathbb{R}^m$, $\tilde{y} : \tilde{I} \to \mathbb{R}^m$ be solutions to (2.1). $\tilde{y}$ is an extension of $y$ to $\tilde{I}$ if $I \subset \tilde{I}$ and $\tilde{y}|_I = y$.*
- *The solution $y : I \to \mathbb{R}^m$ is said to be maximal if there is no extension $\tilde{y} : \tilde{I} \to \mathbb{R}^m$ of $y$ for $I \subset \tilde{I}$.*

Now we can present the definition related to a stable solution.

**Definition 2.2.2** *We consider the following ODE*

$$y'(t) = f(y(t)), \quad t \in [t_0, +\infty),\ t_0 \in \mathbb{R}^+ \tag{2.6}$$

*with the initial condition $y(t_0) = z_0$. We note $y(t, z_0)$ the solution of (2.6). Let $y(t, z)$ be the maximal solution of (2.6) such that $y(t_0, z) = z$. Then the solution $y(t, z_0)$ is stable if there is a closed set $\bar{B}(z_0, r) = \{z,$ such that $\|z - z_0\| \leq r,\ r \in \mathbb{R}^+\}$ and a constant $C \geq 0$ such that:*

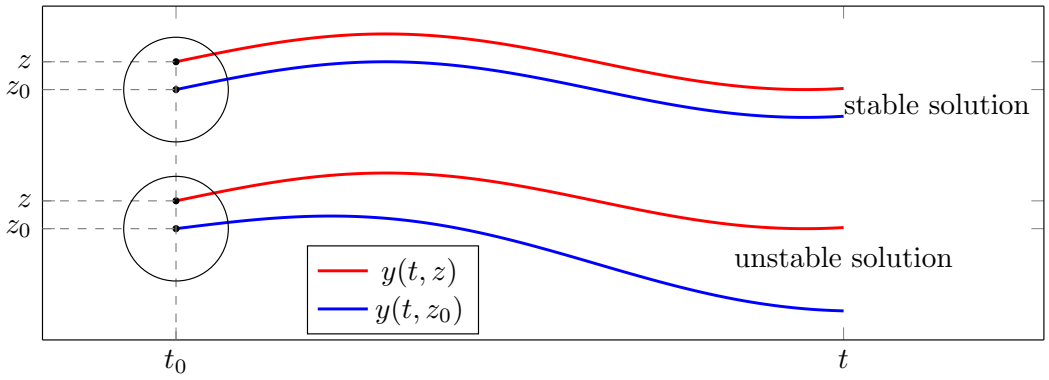



Figure 2.1: Illustration of a stable and unstable solution

- *For all* $z \in \bar{B}(z_0, r)$, $t \to y(t, z)$ *is well defined in* $[t_0, +\infty)$;
- *For all* $z \in \bar{B}(z_0, r)$ *and* $t \geq t_0$ *we have*

$$\|y(t, z) - y(t, z_0)\| \leq C\|z - z_0\|.$$

In the Figure 2.1, we illustrate how the numerical solution behaves when the solution is unstable. In the same figure we observe the variation when the solution is stable as well.

### 2.2.2 An example of linear system

Consider the case where $f(\mathbf{y}(t)) = A\mathbf{y}(t)$ in (2.6), $A$ being a matrix with constant coefficients in $\mathbb{C}$. We consider the following linear system with initial condition (Cauchy problem)

$$\begin{cases} \mathbf{y}'(t) = A\mathbf{y} \\ \mathbf{y}(t_0) = \mathbf{z} \end{cases}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}, \quad A = \begin{pmatrix} a_{11} & \dots & a_{1m} \\ \vdots & \vdots & \vdots \\ a_{m1} & \dots & a_{mm} \end{pmatrix}. \tag{2.7}$$

The solution to the Cauchy problem (2.7) reads

$$\mathbf{y}(t, \mathbf{z}) = \mathrm{e}^{(t-t_0)A}\mathbf{z} \tag{2.8}$$

and

$$\mathbf{y}(t, \mathbf{z}) - \mathbf{y}(t, \mathbf{z}_0) = \mathrm{e}^{(t-t_0)A}(\mathbf{z} - \mathbf{z}_0). \tag{2.9}$$

To ensure stability, the norm $\|\mathrm{e}^{(t-t_0)A}\|$ must be bounded at infinity, when $t \to \infty$. When the matrix $A$ is diagonalizable, according to the definition of the matrix exponential the study reduces to:

- $m = 1$; the equation (2.7) is scalar and

$$y(t, z) = \mathrm{e}^{(t-t_0)\lambda} z$$

Considering the stability of the solution amounts ensuring $|\mathrm{e}^{(t-t_0)\lambda}| = \mathrm{e}^{(t-t_0)Re(\lambda)}$ to be bounded at infinity ($t >> t_0$). We thus have a stable solution if and only if $Re(\lambda) \leq 0$. Moreover, if $Re(\lambda) < 0$ it is called asymptotically stable.

- $m \geq 2$ and $A$ is similar to a diagonal matrix $D$ defined by $D_{ij} = \lambda_i$ for $i = j$ and $D_{ij} = 0$ for $i \neq j$. The coefficients $\lambda_i$, $i = 1, \cdots, m$ are the eigenvalues of the matrix $A$. Then the equation (2.7) consists of $m$ linear independent equations $y_j' = \lambda_j y_j$, $j = 1, \cdots, m$ which give rise to a solution $y := (y_j)_{j=1,\cdots,m}$ of the form

$$y_j(t, \mathbf{z}) = z_j \mathrm{e}^{(t-t_0)\lambda_j}, \quad j = 1, \cdots, m.$$

We thus have a stable solution if and only if $Re(\lambda_j) \leq 0$ for $1 \leq j \leq m$ and it is **asymptotically stable** if $Re(\lambda_j) < 0$ for all $j$.

In [27], the author provides more details for the different cases recalled here and show that the solution is stable only for $Re(\lambda_j) < 0$, $\lambda_j \in sp(A)$ when $A$ is not diagonalisable.

To recapitulate, assuming that $A$ is similar to diagonal matrix, we have

**Theorem 2.2.3** *The linear system of equations (2.7) is stable if and only if all roots $\lambda_i$, $i = 1, \cdots, m$ of the characteristic equation*

$$det(\lambda I - A) = 0, \tag{2.10}$$

*satisfy $Re(\lambda_i) \leq 0$.*

## 2.3 Stability of numerical one-step time integration schemes

This section deals with the stability results of numerical time integration schemes. First of all, we introduce the idea of stability function, then the stability region and we end up with defining the A-stability.

### *2.3.1 Stability function of a time integration scheme*

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 $\Delta t$ is the time step. For simplicity we consider the example of the linear system (2.7) whose analytical solution is given by

$$\mathbf{y}(t_{n+1}) = \mathrm{e}^{\Delta t A}\mathbf{y}(t_n). \tag{2.11}$$

Let $\mathbf{y}_n$ be an approximation of the analytical solution $\mathbf{y}(t_n)$. A given one-step time scheme will compute $y_{n+1}$ from $y_n$ only:

$$\mathbf{y}_{n+1} = R(\Delta t A)\mathbf{y}_n \tag{2.12}$$

The function $R$ can be seen as an approximation of the exponential function of (2.11):

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

**Definition 2.3.1** *The function $R$ in (2.12) is called the stability function of the corresponding numerical scheme.*

Usually, the function $R$ is a polynomial function or a rational function whose numerator and denominator are both polynomial functions (see the Chapters 4, 5 and 6). When $R$ is a polynomial function, computing the solution $\mathbf{y}_{n+1}$ of (2.12) involves only evaluations of the function $f$ (i.e. matrix-vector products for a linear ODE). The scheme is then called explicit scheme. Otherwise, when $R$ is a rational function, computing $\mathbf{y}_{n+1}$ involves also the inversion of a polynomial of matrices. This scheme then provides an implicit representation of the solution and is thus called implicit.

As an example, let us consider the forward and backward Euler methods. Both are constructed by using the following finite difference approximation

$$y'(t) \approx \frac{y(t+\Delta t) - y(t)}{\Delta t},$$

which leads to

$$y(t+\Delta t) \approx y(t) + \Delta t y'(t). \tag{2.13}$$

The forward Euler scheme corresponds to $y'(t_n) = f(t_n, y_n)$ and approximates $y(t_n + \Delta t)$ by $y_{n+1}$. In the case of the linear system (2.8) we obtain

$$\mathbf{y}_{n+1} = (I + \Delta t A)\mathbf{y}_n = R(\Delta t A)\mathbf{y}_n.$$

The forward Euler method is then an explicit scheme and its stability function reads

$$R(z) = 1 + z,\ \forall z \in \mathbb{C}.$$

The backward Euler method, corresponds to $y'(t_n) = f(t_{n+1}, y_{n+1})$ in (2.13) and the numerical solution satisfies

$$(I - \Delta t A)\mathbf{y}_{n+1} = \mathbf{y}_n.$$

The backward Euler method is then an implicit scheme and its stability function is given by

$$R(z) = \frac{1}{1-z},\ \forall z \neq 1.$$

### 2.3.2 *A-Stability of a numerical time scheme*

> I didn't like all these "strong", "perfect", "absolute", "generalized", "super", "hyper", "complete" and so on in mathematical definitions, I wanted something neutral; and having been impressed by David Young's "property A", I chose the term "$A$-stable".
>
> (*G. Dahlquist, in 1979* [42])

**Definition 2.3.2** *Let $R$ be the stability function as previously defined. Then, the stability region of the corresponding numerical scheme is defined as*

$$\mathcal{S} = \{z \in \mathbb{C} \textit{ such that } |R(z)| \leq 1\}. \tag{2.14}$$

The analytical solution is stable if and only if

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

for a given time step $\Delta t$. $sp(A)$ represents the spectrum of the matrix $A$. The same condition must be satisfied by $R(z)$, $z = \Delta t\lambda$ for the numerical solution to be stable. This is the element that underlies the Definition 2.3.2.

If $R$ is polynomial, as it is for explicit scheme, we have $\lim_{|z|\to\infty} |R(z)| = \infty$, $z \in \mathbb{C}^-$. The stability region is then restricted to a bounded domain. This implies that the time step $\Delta t$ must be chosen small enough such that $z = \Delta t\lambda \in \mathcal{S}$, $\forall \lambda \in sp(A)$.

Implicit schemes have rational stability function. For some of them, the stability function may satisfy $|R(z)| \leq 1$, $\forall z \in \mathbb{C}^-$. This may occur when the degree of the denominator $D$ of $R$ is greater or equal than that of its numerator $N$. For this reason, the stability region $\mathcal{S}$ of implicit schemes may cover the entire negative half plane. In that case $sp(A) \subset \mathcal{S}$ since the spectrum of $A$ is usually included in the negative half plane ($sp(A) \subset \mathbb{C}^-$). This means that there is no constraint on the choice of the time step $\Delta t$ to obtain a stable solution using such an implicit scheme. We recall the following definition introduced by Dahlquist [42]:

**Definition 2.3.3** *A numerical scheme, whose stability region satisfies*

$$\mathcal{S} \supset \mathbb{C}^- = \{z \in \mathbb{C},\ Re(z) \leq 0\} \tag{2.15}$$

*is called A-stable.*

It is clear that A-stable schemes are unconditionally stable i.e. there is no constraint regarding the stability. It is also clear that explicit schemes cannot be A-stable. Only implicit schemes may offer the opportunity of verifying the A-stability property because they have a rational stability function. A simple way to verify the A-stability property of implicit schemes is to use the following result:

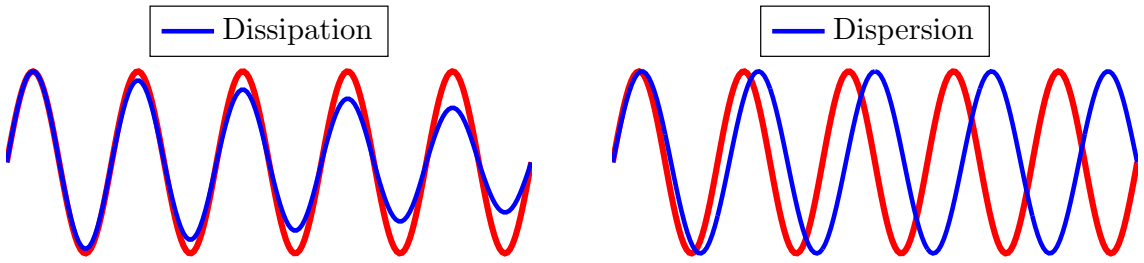



Figure 2.2: Dissipation and dispersion effects. On the left the numerical solution (blue line) has amplitude error and on the right it has phase error.

**Proposition 2.3.4** *A time integration method whose stability function $R$ is a rational function with polynomial numerator $N$ and polynomial denominator $D$ is A-stable if and only if*

$$|R(iy)| \leq 1 \quad \textit{for all } y \in \mathbb{R} \tag{2.16}$$

*and*

$$R(z) \quad \textit{is analytic for } Re(z) < 0,\ z \in \mathbb{C}. \tag{2.17}$$

This result comes from the maximum principle applied to $\mathbb{C}^-$ (see [42]).

## 2.4 Dissipation and dispersion of numerical time integration schemes

> 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* [74]

In the previous section, we have introduced the stability of a given numerical scheme. It is important to figure out that the stability of a numerical schemes doesn't guarantee its accuracy. For wave propagation problems the accuracy of a given numerical solution depends on the numerical dissipation and the numerical dispersion.

A numerical scheme is dispersive if the numerical and exact solutions have a different phase speed while it is dissipative if they have a different amplitude. In the Figure 2.2 we illustrate how a dissipative scheme may affect the amplitude of the numerical solution compared to the physical solution. We also represent the effect of a dispersive scheme on the numerical solution to show the phase shift between the numerical and the physical solution.

Following the analysis in [44] we propose to write explicitely the relation between the stability function and the errors of amplitude and phase.

To illustrate the dissipation and dispersion effects, we choose to consider the following linear test equation

$$y' = i\lambda y, \quad y(t_0) = y_0 \text{ and } \lambda \in \mathbb{R}, \tag{2.18}$$

At each time step, the numerical solution reads then

$$y_{n+1} = R(iz)y_n, \tag{2.19}$$

where $z = \lambda\,\Delta t$. The exact solution to the test equation (2.18) is given by

$$y_{n+1} = \mathrm{e}^{iz}y_n \tag{2.20}$$

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

Therefore, we define the dissipation and dispersion errors, as follows:

**Definition 2.4.1** *The dissipation error of a numerical scheme applied to* (2.18) *is measured by the function*

$$\psi(z) = |R(iz)| - 1, \tag{2.21}$$

*and the dispersion error of a numerical scheme applied to* (2.18) *is measured by the function*

$$\Phi(z) := \arg\left[\frac{\mathrm{e}^{iz}}{R(iz)}\right] = z - arg[R(iz)]. \tag{2.22}$$

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

Knowing the stability function of a numerical scheme, we can represent the relative dispersion and the relative dissipation errors using the Definition 2.4.1. For that purpose, we only have to choose a set of values for $z$ and evaluate the functions $\psi(z)$ for the dissipation and $\Phi(z)$ for the dispersion on that set.

## 2.5 Stiffness of an ODE

When solving a time evolution problem, we can use either explicit or implicit schemes as mentioned in the Section 2.3. To the best of our knowledge, there is no clear definition of the stiffness of an ODE. Roughly speaking, a time evolution problem could be stiff if time explicit schemes demonstrate severe issues of stability (see [42]).

The stiffness of a time dependent problem is generally related to the discretization used in space. To briefly illustrate this fact, let us consider the following ODE:

$$\begin{cases} M_h\dfrac{dX(t)}{dt} + K_hX(t) = F(t) \quad t \in (0,T] \\ X(0) = X_0 \end{cases} \tag{2.23}$$

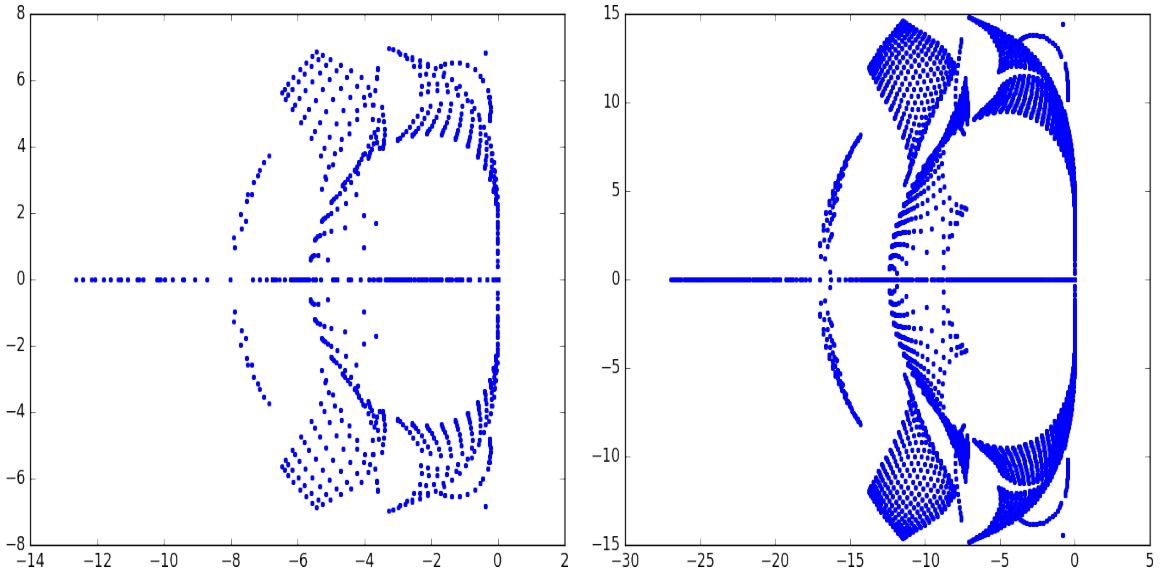
obtained after spatial discretization of a PDE, where $M_h$ is the mass matrix, $K_h$ is the stiffness matrix and $h$ denotes the mesh size. $F(t)$ is a source term obtained after discretizing the continuous source term in space and $X_0$ is the initial condition. In the Chapter 3, it is detailed how this ODE is obtained in the case of the solution of wave equations. The ODE (2.23) becomes stiff when the eigenvalues $\lambda_i$ of $A = M_h^{-1}K_h$ are large. This will be the case:

- if there are small elements in the computational mesh,
- or if we use high-order space discretization.

The spectrum of $A = M_h^{-1}K_h$ is depicted in the Figures 2.3 and 2.4. In the Figure 2.3 we have used a finite element method of order 2 ($\mathbb{Q}_2$) with HDG (Hybridizable Discontinous Galerkin) formulation on a regular grid covering $[-4, 4]^2$ to discretize the acoustic wave equation with Neumann boundary condition (see the Chapter 3). In the Figure 2.3(a) we have chosen only $10$ points along the $x$-axis. Then in the Figure 2.3(b) we have refined the mesh by taking $20$ points along the $x$-axis instead of $10$. We can observe that the spectrum is almost twice bigger in the Figure 2.3(b) than in the Figure 2.3(a). In the Figure 2.4 we have used HDG method on a regular grid covering $[-4, 4]^2$ with $10$ points along the $x$-axis to discretize the acoustic wave equation with Neumann boundary condition. In the Figure 2.4(a) we have the spectrum when we use the $\mathbb{Q}_2$ elements. In the Figure 2.4(b) we have used the $\mathbb{Q}_4$-elements (order 4) and we can observe that the spectrum is about three times bigger than that of the Figure 2.4(a). It shows that increasing the order of approximation ($p$-refinement) is a stiffer procedure than increasing the number of points in the mesh ($h$-refinement).

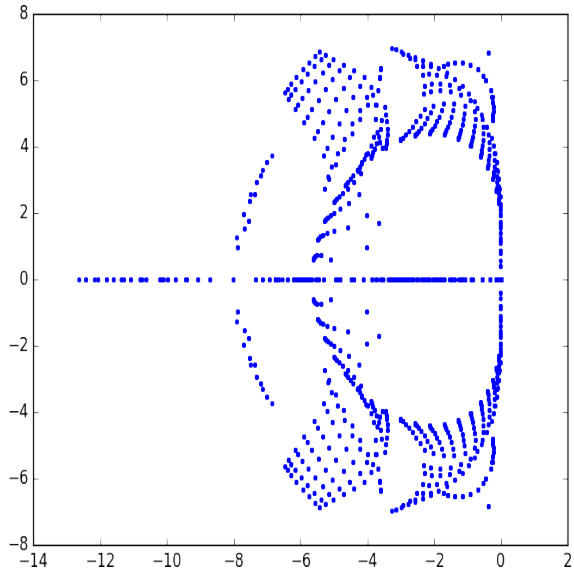
We can conclude that when we refine the mesh or increase the space discretization order, the resulting ODE becomes stiffer. As a consequence explicit methods may become too expensive because the time step $\Delta t$ must be chosen very small such that $\Delta t\lambda_i \in \mathcal{S}$ in order to get a stable solution.

Implicit methods generally have good stability properties which may allow to take large time-steps $\Delta t$. But computing a numerical solution with an implicit method requires to solve at each time-step one or several linear systems involving the matrix $A = M_h^{-1}K_h$. For this reason implicit methods can be expensive in terms of memory usage. In fact by increasing the order of the polynomial used in the space discretization method or by refining the mesh, the size of the matrix $A$ becomes larger. When solving realistic problems, in 3D for example, the memory space needed to solve the linear systems may not be available which makes most of the implicit schemes impracticable for this class of applications.

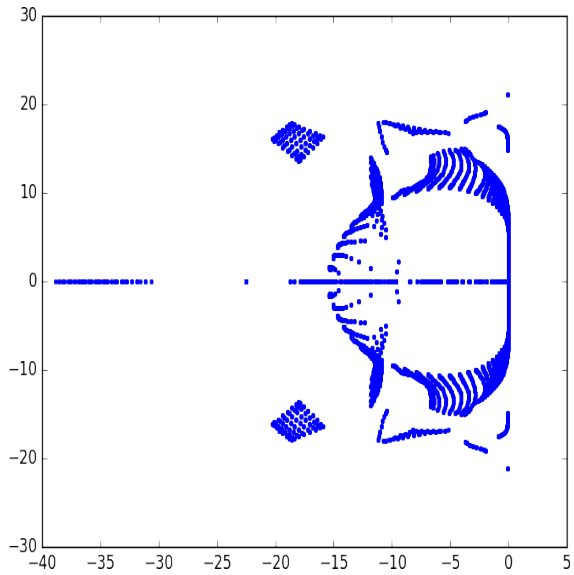



(a) 10 points along x (b) 20 points along x

Figure 2.3: Spectrum of $M_h^{-1}K_h$ for the second order ($\mathbb{Q}_2$) HDG method on the regular grid $[-4,4]^2$. Acoustic wave equation with Neumann boundary condition




(a) Order 2 ($\mathbb{Q}_2$)




(b) Order 4 ($\mathbb{Q}_4$)

Figure 2.4: Spectrum of $M_h^{-1}K_h$ for HDG method on the regular grid $[-4,4]^2$ with $10$ points along the $x$ axis. Acoustic wave equation with Neumann boundary condition.

Chapter 3

# Space discretizations of the wave equations

This chapter provides the details on the methods used to discretize the space domain in 1D, 2D and 3D following the thesis [31]. The content corresponds to the implementation proposed in the high-order finite element code Montjoie. The numerical methods have been developed in this software. We present the first order formulation of the acoustic wave equation and the Maxwell's equations. For each type of equations we provide the variational formulation and the corresponding semi-discrete ODE when using either the continuous finite elements or hybridizable discontinuous Galerkin (HDG) methods. In addition, the formulation with classical boundary conditions and the formulation integrating the PML are also given. The technique of mass lumping is used to obtain diagonal and block-diagonal matrices. When using implicit schemes, this technique is combined with the static condensation technique in order to obtain a smaller linear system to be solved.

## Contents

## 3.1 Introduction

In this chapter we state the different space discretization methods we use for solving the wave equations. For each method we present the associated linear system when we use implicit time integration schemes. In coming chapters, we will provide more details on the time integration schemes we have developed.

Even though our work can be used for any partial differential equation involving the solution of an ODE in time, herein we are interested in solving the time domain acoustic wave equation formulated as a first order system and Maxwell's equations. We recall the acoustic wave equation and the Maxwell's equations respectively in the next two sections of this document. For each wave problem we recall the finite elements method (FEM - [57, 23, 31, 65, 8]) and the hybridizable discontinuous Galerkin (HDG - [51, 11, 50, 20, 17]) method used for the space discretization to construct the corresponding semi-discrete problems.

## 3.2 The acoustic wave equation

In this section we introduce the first order formulation of the acoustic wave equation. We denote by $u$ a scalar field and $\mathbf{v}$ a vectorial field. Both depend on the space variable $\mathbf{x}$ and the time $t$. They are solutions to the following boundary value problem:

$$\begin{cases} \rho\,\partial_t u - \operatorname{div}\mathbf{v} = 0, \quad \forall(\mathbf{x},t)\in\Omega\times\mathbb{R}^+ \\ \mu^{-1}\partial_t\mathbf{v} - \nabla u = 0, \quad \forall(\mathbf{x},t)\in\Omega\times\mathbb{R}^+ \\ u(\mathbf{x},0) = \partial_t u(\mathbf{x},0) = 0, \quad \forall\mathbf{x}\in\Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad \mathbf{x}\in\Gamma_D \quad \text{(Dirichlet condition)} \\ \mu\partial_n u = f_N, \quad \mathbf{x}\in\Gamma_N \quad \text{(Neumann condition)} \\ \mu\partial_n u + \sqrt{\rho\mu}\,\partial_t u = f_A, \quad \mathbf{x}\in\Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{3.1}$$

where $\Omega$ is the computational domain. $\Gamma_D$, $\Gamma_N$ and $\Gamma_A$ are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. $\mathbf{n}$ is the unit outgoing normal vector of the considered boundary, $\rho$ and $\mu$ are physical parameters, which are piecewise constant. $f_D$, $f_N$ and $f_A$ are given source functions. To model the acoustic wave propagation problems, we consider

$$\rho = \frac{1}{\rho_f c^2}, \quad \mu = \frac{1}{\rho_f}, \tag{3.2}$$

where $\rho_f$ represents the fluid density and $c$ is the acoustic wave propagation speed (which can be given by the speed of sound) in meter per second (m/s). In that case the scalar field $u$ represents the pressure field in Pascal (Pa) and the vector field $\mathbf{v}$ stands for the displacement of the wave field.

### 3.2.1 Discretization with continuous finite elements

In this section we present mixed spectral finite elements [23, 25] used to discretize the acoustic wave equation (3.1). In [31] this method is detailed for the Helmholtz equation. The technique of mass lumping is used to obtain a diagonal or block-diagonal mass matrix.

At a first glance, the first-order formulation seems to produce large linear systems (with two unknowns $u$ and v) while the second-order formulation would involve only one unknown $u$. In this section, it will be explained how the unknown v can be removed explicitly in order to obtain a linear system in $u$ only. Moreover, we will apply a static condensation - which is a well-known method [6, 72] in order to remove internal degrees of freedom of $u$. As a result, the linear system to solve will be quite small. The whole procedure reducing the size of the linear system will be called static condensation. It will be detailed both for a case with absorbing boundary conditions and for a case with PML ([38, 7, 24]).

Let $\Omega \subset \mathbb{R}^d$, $d \in \mathbb{N}$ be an open set. We denote $L^2(\Omega)$ the space of square-integrable functions of $\Omega$ with the associated norm $\|\cdot\|$ and the inner-product $<\cdot,\cdot>$. We denote by $H^1(\Omega)$ the set of functions $\phi \in L^2(\Omega)$ having their gradient in $(L^2(\Omega))^d$.

#### Case without PML

The computational domain $\Omega$ is meshed with regular intervals in 1D, quadrilaterals in 2D and tetrahedra, prisms, pyramids and hexahedra in 3D. Each element is denoted by $K_i$:

$$\Omega = \bigcup K_i$$

The equation (3.1) is solved with mixed spectral elements (see [23]) using the following finite element spaces for quadrilateral/hexahedral elements

$$\begin{aligned} u(t) \in U_h &= \left\{ u \in H^1(\Omega) \text{ such that } u|_{K_i} \circ F_i \in \mathbb{Q}_r \right\} \\ v(t) \in V_h &= \left\{ v \in (L^2(\Omega))^d \text{ such that } v|_{K_i} \circ F_i \in (\mathbb{Q}_r)^d \right\} \end{aligned} \tag{3.3}$$

where $d$ is the dimension, $r$ is the order of approximation, $\mathbb{Q}_r$ is the space of polynomials of degree lower or equal to $r$ in each space variable, and $F_i$ is the transformation map from the reference element $\hat{K}$ to the mesh element $K_i$ [24]. For instance in 2D, $F_i$ is the map from the unit square $\hat{K}$ to the element $K_i$ (see the Figure 3.1). We note $DF_i$ the Jacobian matrix of $F_i$ and $J_i$ its determinant. $J_i$ is assumed to be always strictly positive, the local numbering is modified to ensure the positiveness of $J_i$. We take $\varphi_i$ the basis function associated with the space $U_h$ and $\psi_i$ the basis function associated with $V_h$. We use Gauss-Lobatto quadrature points for both interpolation and quadrature formulas as in [23]. We have the following relations:

$$\hat{\varphi}_i = \varphi_i \circ F_i, \quad \hat{\psi}_i = \psi_i \circ F_i$$

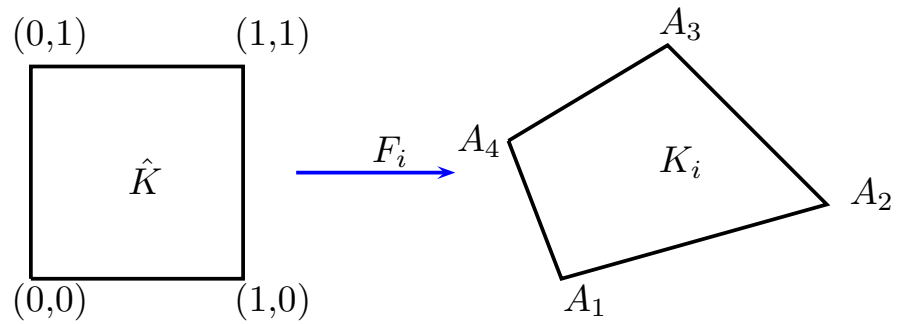



Figure 3.1: Transformation $F_i$ for a quadrilateral.

with

$$\hat{\varphi}_i(\hat{x},\hat{y},\hat{z}) = \varphi_{i_1}^{1D}(\hat{x})\,\varphi_{i_2}^{1D}(\hat{y})\,\varphi_{i_3}^{1D}(\hat{z}), \quad \varphi_i^{1D}(\hat{x}) = \frac{\prod_{j\neq i}(\hat{x}-\hat{\xi}_j^{1D})}{\prod_{j\neq i}(\hat{\xi}_i^{1D}-\hat{\xi}_j^{1D})}$$

for hexahedra where $\hat{\xi}_i^{1D}$ are Gauss-Lobatto points on the interval $[0,1]$. For other elements, the basis functions and finite element spaces $U_h$ and $V_h$ are detailed in [8]. The quadrature points on the unit cube are obtained by tensorization

$$\hat{\xi}_k = (\hat{\xi}_{k_1}^{1D}, \hat{\xi}_{k_2}^{1D}, \hat{\xi}_{k_3}^{1D}).$$

By construction the basis functions on the cube satisfy

$$\hat{\varphi}_i(\hat{\xi}_j) = \delta_{i,j}.$$

The choice of basis functions $\psi_i$ are based on the quadrature points $\hat{\xi}_k$ (even for other elements than hexahedra):

$$\hat{\psi}_i^1(\hat{\xi}_k) = \delta_{i,k}\,e_x, \quad \hat{\psi}_i^2(\hat{\xi}_k) = \delta_{i,k}\,e_y, \quad \hat{\psi}_i^3(\hat{\xi}_k) = \delta_{i,k}\,e_z.$$

After multiplying the equation (3.1) by the basis functions $\varphi_i$ and $\psi_i$, the variational formulation is obtained by integrating by parts the first equation and reads:

$$\begin{cases} \dfrac{d}{dt}\displaystyle\int_\Omega \rho\,u\,\varphi_i\,dx + \int_\Omega v\cdot\nabla\varphi_i\,dx + \int_{\Gamma_A}\sqrt{\rho\mu}\,u\,\varphi_i\,dx = \int_\Gamma f\varphi_i\,dx \\ \dfrac{d}{dt}\displaystyle\int_\Omega \mu^{-1}v\cdot\psi_i\,dx - \int_\Omega \nabla u\cdot\psi_i\,dx = 0 \end{cases} \tag{3.4}$$

The obtained semi-discrete system is then given by:

$$\begin{cases} D_h\dfrac{dU}{dt} + R_hV + S_hU = F_u(t) \\ B_h\dfrac{dV}{dt} - R_h^TU = 0 \end{cases} \tag{3.5}$$

where $h$ is the mesh size, $T$ denotes the transpose matrix and the finite element matrices are given by:

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

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

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

$$(B_h)_{i,j} = \int_\Omega \mu^{-1}\psi_i \cdot \psi_j\, dx$$

Since the basis functions $\psi_i$ are associated with quadrature points, the matrix $B_h$ is diagonal for the orthotropic case ($\mu$ orthotropic) because:

$$\int_{K_e} \mu^{-1}\psi_j^\ell \cdot \psi_i^m = (\mu^{-1})_{\ell,m}\, \omega_i\, J_e(\hat{\xi}_i)\, \delta_{i,j}$$

where $\omega_i$ is the weight associated with the point $\hat{\xi}_i$. In the anisotropic case $B_h$ will be block diagonal each block being a $d \times d$ matrix with $d$ the dimension of the space. The elementary stiffness matrix is given by:

$$\int_{K_e} \psi_j^\ell \cdot \nabla\varphi_i\, dx = \omega_j\, J_e(\hat{\xi}_j) DF_e^{-T}(\hat{\xi}_j) \hat{\nabla}\hat{\varphi}_i(\hat{\xi}_j) \cdot e_\ell.$$

The source term is defined as:

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

A source term $F_v$ may come from the inhomogeneous Dirichlet condition (if $f_D \neq 0$). In fact, the 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 $F_v$. Let $X = (U, V)^t$ and $F = (F_u, F_v)^t$, the evolution system (3.5) falls in the class of ODEs of the type

$$\begin{cases} M_h \dfrac{dX(t)}{dt} + K_h X(t) = F(t) \quad t \in (0, T] \\ X(0) = X_0 \end{cases} \tag{3.6}$$

where $M_h$ the mass matrix and $K_h$ the stiffness matrix are given by

$$M_h = \begin{pmatrix} D_h & 0 \\ 0 & B_h \end{pmatrix}, \qquad K_h = \begin{pmatrix} S_h & R_h \\ -R_h^T & 0 \end{pmatrix}.$$

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. The

stiffness matrix is not stored, the matrix-vector product $K_h X$ is performed on the fly by using matrices $DF_e^{-T}(\hat{\xi}_j)$ computed on quadrature points. The mass matrix $D_h$ is diagonal for quadrilateral/hexahedral elements since:

$$\int_{K_e} \varphi_j \varphi_i dx = \omega_i J_e(\hat{\xi}_i)\delta_{i,j}.$$

This property is called mass lumping and is advantageous for explicit time schemes since no linear system needs to be solved. $D_h$ is no longer diagonal for other elements (pyramids, prisms and tetrahedra) with the basis functions chosen (as detailed in [8] and [24]). Some attempts have been made to obtain mass lumping when using tetrahedra and triangular elements [37, 58, 19, 24]. The efficiency of such elements is rather poor. Moreover, since we are mainly interested in implicit time schemes, we preferred to stick to classical elements with the drawback of dealing with a non-diagonal mass matrix when the mesh includes tetrahedra, prisms or pyramids.

**Static condensation:** When using an implicit time-scheme, we need to solve a linear system (or several) of the form:

$$\begin{cases} \beta\, D_h U + S_h U + R_h V = F_U \\ \beta\, B_h V - R_h^T U = F_V \end{cases} \tag{3.7}$$

where $\beta$ is a coefficient depending on the time scheme used. The second equation of (3.7) is used to substitute the unknown $V$ in the first equation. By this procedure, we obtain a symmetric linear system to solve for $U$:

$$\left(\beta^2\, D_h + \beta S_h + R_h B_h^{-1} R_h^T\right) U = \beta F_U - R_h B_h^{-1} F_V \tag{3.8}$$

Once we know $U$, $V$ is easily computed from (3.7). We note $K_h$ the stiffness matrix $K_h$ given by

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

and the source term

$$F_h = \beta F_U - R_h B_h^{-1} F_V.$$

The global matrix $K_h$ is constructed using the following elementary matrices

$$\sum_k \omega_k J_e(\hat{\xi}_k)\mu(\hat{\xi}_k) \left[DF_e^{-T}(\hat{\xi}_k)\hat{\nabla}\hat{\varphi}_i(\hat{\xi}_k)\right] \cdot \left[DF_e^{-T}(\hat{\xi}_k)\hat{\nabla}\hat{\varphi}_j(\hat{\xi}_k)\right]$$

We finally deduce that:

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

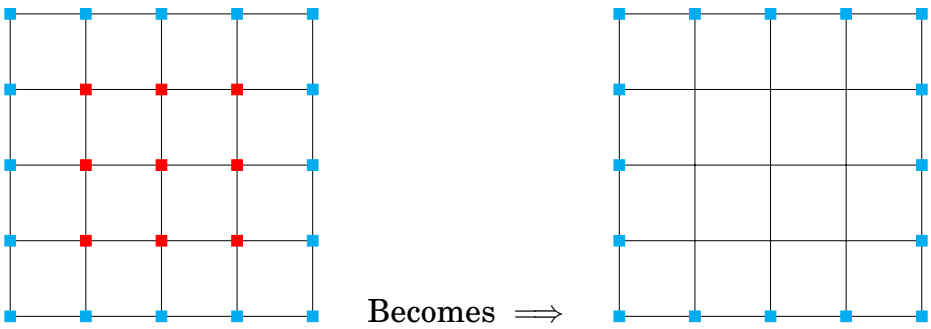



Figure 3.2: Illustration of the static condensation used to remove internal degrees of freedom of a mesh element.

which is true for any element $K_i$ since we have chosen basis functions $\psi_i$ associated with the quadrature points. The system (3.8) is the same kind of linear system that appears when using $\theta$-schemes for the second-order formulation of the wave equation. This system is symmetric positive definite if $\beta$ 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. In fact, the global stiffness matrix $K_h$ is partitioned into block matrices coming from each element. Using Gaussian elimination on an element level allows to remove unknowns related to internal degrees of freedom. As a result the final linear system has a smaller size and by this way the required computational effort. To illustrate the elimination process, we denote by $U_e$ and $U_i$ the unknown vectors related to the external (cyan square) and internal (red square) degrees of freedom in the Figure 3.2. We assume that the matrix of the element under consideration are partitioned into the form

$$\overbrace{\begin{pmatrix} K_{ee} & K_{ei} \\ K_{ie} & K_{ii} \end{pmatrix}}^{K\text{elem}} \begin{pmatrix} U_e \\ U_i \end{pmatrix} = \begin{pmatrix} F_e \\ F_i \end{pmatrix} \tag{3.9}$$

Using the second matrix equation in (3.9) leads to

$$U_i = K_{ii}^{-1}(F_i - K_{ie}U_e). \tag{3.10}$$

The relation (3.10) is then used to substitute for $U_i$ into the first matrix equation in (3.9) to obtain the condensed equation in $U_e$ (only on the external degrees of freedom)

$$\underbrace{(K_{ee} - K_{ei}K_{ii}^{-1}K_{ie})}_{\textit{Schur complement}} U_e = F_e - K_{ei}K_{ii}^{-1}F_i. \tag{3.11}$$

Since the computation of the Schur complement and the right hand side in (3.11) as well as the condensation step in (3.8) are local for all elements, the process is suitable for parallelization.

**Case with PML in 2D**

In order to truncate the 2D domain, we use here Perfectly Matched Layers (PML). In this paragraph, the efficient implementation of PML is detailed. We split the physical unknown $u$ of the equation (3.1) as follows

$$u = u_x + u_y.$$

Then we consider the following split formulation

$$\begin{cases} \rho\,\partial_t u_x \;+\; \sigma_x\,\rho\,u_x - \partial_x v_x = 0 \\ \rho\,\partial_t u_y \;+\; \sigma_y\,\rho\,u_y - \partial_y v_y = 0 \\ \mu^{-1}\partial_t v \;+\; \mu^{-1} T_{x,y} v - \nabla(u_x + u_y) = 0 \\ \text{+ homogeneous Neumann condition on PML boundaries} \end{cases}$$

where $u_x, u_y, v$ are intermediary unknowns, $T_{x,y}$ is damping tensor given by

$$T_{x,y} = \begin{pmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{pmatrix}$$

and $\sigma_x, \sigma_y$ are damping functions, non-null inside the PML, with a parabolic profile (see [26]):

$$\sigma_x \;=\; \frac{3\,\log 1000}{2a^3}(x - x_0)^2\,\sigma\;v\text{max}$$

$$\sigma_y \;=\; \frac{3\,\log 1000}{2a^3}(y - y_0)^2\,\sigma\;v\text{max}$$

with $\sigma$ a damping coefficient chosen *a priori*, $v\text{max}$ is the maximal wave velocity, $x_0, y_0$ are equal to $x\text{min}, y\text{min}, x\text{max},$ or $y\text{max}$ depending on the layer considered. In order to have directly the physical field $u$ as unknown, we write the problem with the two unknowns:

$$u \;=\; u_x + u_y$$

$$u^* \;=\; u_x - u_y$$

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

$$\begin{cases} \rho\,\partial_t u \;+ \rho\left(\dfrac{\sigma_x + \sigma_y}{2}\right) u \;+\; \rho\left(\dfrac{\sigma_x - \sigma_y}{2}\right) u^* \;-\; \text{div}\, v = 0 \\ \rho\,\partial_t u^* \;+\; \rho\left(\dfrac{\sigma_x + \sigma_y}{2}\right) u^* \;+\; \rho\left(\dfrac{\sigma_x - \sigma_y}{2}\right) u \;-\; (\partial_x v_x - \partial_y v_y) = 0 \\ \mu^{-1}\partial_t v \;+\; \mu^{-1} T_{x,y}\, v - \nabla u = 0 \end{cases} . \tag{3.12}$$

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 semi-discrete system is given by

$$\begin{cases} D_h \dfrac{dU}{dt} + S_h^{xx} U + S_h^{xy} U^* + R_h V = 0 \\ D_h^* \dfrac{dU^*}{dt} + S_h^{yy} U^* + S_h^{yx} U + R_h^* V = 0 \\ B_h \dfrac{dV}{dt} + T_h V - R_h^T U = 0 \end{cases}$$

where the second equation holds in the PML region only. The matrix $R_h^*$ is obtained after multiplying and integrating by parts the second equation of (3.12):

$$(R_h^*)_{i,j} = \int_\Omega (\psi_j)_x \partial_x \varphi_i - (\psi_j)_y \partial_y \varphi_i.$$

The semi-discrete system falls also into the class of ODE (3.6) by taking:

$$X = \begin{pmatrix} U \\ U^* \\ V \end{pmatrix}, \qquad M_h = \begin{pmatrix} D_h & 0 & 0 \\ 0 & D_h^* & 0 \\ 0 & 0 & B_h \end{pmatrix}, \qquad K_h = \begin{pmatrix} S_h^{xx} & S_h^{xy} & R_h \\ S_h^{yx} & S_h^{yy} & R_h^* \\ -R_h^T & 0 & T_h \end{pmatrix}.$$

**Static condensation:** The linear system to be solved in time is given by:

$$\rho\,\beta\,u + \rho\left(\frac{\sigma_x + \sigma_y}{2}\right) u + \rho\left(\frac{\sigma_x - \sigma_y}{2}\right) u^* - \operatorname{div} v = f \tag{3.13a}$$

$$\rho\,\beta\,u^* + \rho\left(\frac{\sigma_x + \sigma_y}{2}\right) u^* + \rho\left(\frac{\sigma_x - \sigma_y}{2}\right) u - (\partial_x v_x - \partial_y v_y) = f^* \tag{3.13b}$$

$$\mu^{-1}\beta\, v + \mu^{-1} T_{x,y}\, v - \nabla u = f_v \tag{3.13c}$$

where $\beta$ is a coefficient depending on the time scheme used. The unknown $u_x$ and $u_y$ can be reconstructed by adding and subtracting (3.13a) and (3.13b). As a result, we obtain:

$$\begin{cases} \rho\,\beta u_x + \rho\,\sigma_x\, u_x - \partial_x v_x = \dfrac{f + f^*}{2} \\ \rho\,\beta u_y + \rho\,\sigma_y\, u_y - \partial_y v_y = \dfrac{f - f^*}{2} \\ \mu^{-1}\beta v + \mu^{-1} T_{x,y} v - \nabla(u_x + u_y) = f_v \end{cases}$$

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

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

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

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

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

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

where

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

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

and

$$(F_h)_i=\int_\Omega\left[\beta+\frac{\sigma_x+\sigma_y}{2}\right]f\varphi_i+\frac{\sigma_y-\sigma_x}{2}f^*\varphi_i-\begin{pmatrix}\mu\frac{\beta+\sigma_y}{\beta+\sigma_x} & 0\\ 0 & \mu\frac{\beta+\sigma_x}{\beta+\sigma_y}\end{pmatrix}f_v\cdot\nabla\varphi_i\,dx$$

This result is true when the matrices $D_h$, $D_h^*$, $S_h^{xx}$, $S_h^{xy}$, $S_h^{yx}$, $S_h^{yy}$ are diagonal. It induces that the computational mesh must comprise only quadrilateral elements in the PML areas. It will be the case in our numerical computations. 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 (3.13b) and (3.13c).

### Case with PML in 3D

In the 3D case the unknowns are split as follows:

$$u=u_x+u_y+u_z$$

Then the split system takes the form:

$$\begin{cases} \rho\,\partial_t u_x \;+\; \sigma_x\,\rho\,u_x - \partial_x v_x = 0 \\ \rho\,\partial_t u_y \;+\; \sigma_y\,\rho\,u_y - \partial_y v_y = 0 \\ \rho\,\partial_t u_z \;+\; \sigma_z\,\rho\,u_z - \partial_z v_z = 0 \\ \mu^{-1}\partial_t v \;+\; \mu^{-1} T_{x,y,z} v - \nabla(u_x + u_y + u_z) = 0 \\ \text{+ Neumann condition on PML boundaries} \end{cases}$$

where $T_{x,y,z}$ is the damping tensor given by

$$T_{x,y,z} = \begin{pmatrix} \sigma_x & 0 & 0 \\ 0 & \sigma_y & 0 \\ 0 & 0 & \sigma_z \end{pmatrix},$$

$\sigma_x, \sigma_y$ are the same as for the 2D case and for $\sigma_z$ we take:

$$\sigma_z \;=\; \frac{3\,\log 1000}{2a^3}(z - z_0)^2\,\sigma\;v\mathbf{max}$$

Like in 2D, to directly have the physical field $u$, we introduce the following unknowns:

$$u \;=\; u_x + u_y + u_z$$

$$u^* \;=\; u_x - u_y$$

$$u^\diamond \;=\; u_x - u_z$$

We then obtain the system for $u, u^*, u^\diamond, v$ in the form:

$$\begin{cases} \rho\,\partial_t u + \rho\left(\dfrac{\sigma_x + \sigma_y + \sigma_z}{3}\right) u + \rho\left(\dfrac{\sigma_x - 2\sigma_y + \sigma_z}{3}\right) u^* \\ \qquad\qquad + \rho\left(\dfrac{\sigma_x + \sigma_y - 2\sigma_z}{3}\right) u^\diamond - \mathbf{div}\, v = 0 \\ \rho\,\partial_t u^* + \rho\left(\dfrac{\sigma_x + 2\sigma_y}{3}\right) u^* + \rho\left(\dfrac{\sigma_x - \sigma_y}{3}\right)(u + u^\diamond) - (\partial_x v_x - \partial_y v_y) = 0 \\ \rho\,\partial_t u^\diamond + \rho\left(\dfrac{\sigma_x + 2\sigma_z}{3}\right) u^\diamond + \rho\left(\dfrac{\sigma_x - \sigma_z}{3}\right)(u + u^*) - (\partial_x v_x - \partial_z v_z) = 0 \\ \mu^{-1}\partial_t v + \mu^{-1} T_{x,y,z} v - \nabla u = 0 \end{cases} \tag{3.14}$$

$u^*$ and $u^\diamond$ are only discretized in the PML region. Then after multiplying (3.14) by the basis functions and integrating by parts the second and third equations

we obtain the semi-discrete system which reads

$$\begin{cases} D_h \dfrac{dU}{dt} + S_h^{xxx} U + S_h^{xyx} U^* + S_h^{xyz} U^\diamond + R_h V = 0 \\ D_h^* \dfrac{dU^*}{dt} + S_h^{yyy} U^* + S_h^{yxy} U + S_h^{yxz} U^\diamond + R_h^* V = 0 \\ D_h^\diamond \dfrac{dU^\diamond}{dt} + S_h^{zzz} U^\diamond + S_h^{zyz} U^* + S_h^{zyx} U + R_h^\diamond V = 0 \\ B_h \dfrac{dV}{dt} + T_h V - R_h^T U = 0 \end{cases}$$

where the second and third equations hold in the PML region only. The matrix $R_h^\diamond$ is given by

$$(R_h^\diamond)_{i,j} = \int_\Omega (\psi_j)_x \partial_x \varphi_i - (\psi_j)_z \partial_z \varphi_i.$$

As for the 2D case, the semi-discrete system falls also into the class of ODE (3.6) by taking:

$$X = \begin{pmatrix} U \\ U^* \\ U^\diamond \\ V \end{pmatrix}, \qquad M_h = \begin{pmatrix} D_h & 0 & 0 & 0 \\ 0 & D_h^* & 0 & 0 \\ 0 & 0 & D_h^\diamond & 0 \\ 0 & 0 & 0 & B_h \end{pmatrix}, \qquad K_h = \begin{pmatrix} S_h^{xxx} & S_h^{xyx} & S_h^{xyz} & R_h \\ S_h^{yxy} & S_h^{yyy} & S_h^{yxz} & R_h^* \\ S_h^{zyx} & S_h^{zyz} & S_h^{zzz} & R_h^\diamond \\ -R_h^T & 0 & 0 & T_h \end{pmatrix}.$$

**Static condensation:** The linear system to be solved when using implicit time schemes is then given by:

$$\begin{aligned} &\rho\beta u + \rho\left(\frac{\sigma_x+\sigma_y+\sigma_z}{3}\right) u + \rho\left(\frac{\sigma_x-2\sigma_y+\sigma_z}{3}\right) u^* \\ &\qquad + \rho\left(\frac{\sigma_x+\sigma_y-2\sigma_z}{3}\right) u^\diamond - \mathbf{div}\, v = f \end{aligned} \tag{3.15a}$$

$$\rho\beta u^* + \rho\left(\frac{\sigma_x+2\sigma_y}{3}\right) u^* + \rho\left(\frac{\sigma_x-\sigma_y}{3}\right)(u+u^\diamond) - (\partial_x v_x - \partial_y v_y) = f^* \tag{3.15b}$$

$$\rho\beta u^\diamond + \rho\left(\frac{\sigma_x+2\sigma_z}{3}\right) u^\diamond + \rho\left(\frac{\sigma_x-\sigma_z}{3}\right)(u+u^*) - (\partial_x v_x - \partial_z v_z) = f^\diamond \tag{3.15c}$$

$$\mu^{-1}\beta v + \mu^{-1} T_{x,y,z} v - \nabla u = f_v \tag{3.15d}$$

The unknowns $u_x$, $u_y$ and $u_z$ can then be reconstructed by combining the first three equations:

$$\begin{cases} \rho\,\beta u_x + \rho\,\sigma_x\,u_x - \partial_x v_x = \dfrac{f+f^*+f^\diamond}{3} \\ \rho\,\beta u_y + \rho\,\sigma_y\,u_y - \partial_y v_y = \dfrac{f-2f^*+f^\diamond}{3} \\ \rho\,\beta u_z + \rho\,\sigma_y\,u_z - \partial_z v_z = \dfrac{f+f^*-2f^\diamond}{3} \\ \mu^{-1}\beta v + \mu^{-1} T_{x,y,z} v - \nabla(u_x+u_y) = f_v \end{cases}$$

The first equation is multiplied by $(\beta+\sigma_y)(\beta+\sigma_z)$, the second equation by $(\beta+\sigma_x)(\beta+\sigma_z)$ and the third one by $(\beta+\sigma_x)(\beta+\sigma_y)$. The three results are combined to obtain

$$\begin{aligned}
&\rho(\beta+\sigma_x)(\beta+\sigma_y)(\beta+\sigma_z)u \\
&\quad -\operatorname{div}\left[\begin{pmatrix}(\beta+\sigma_y)(\beta+\sigma_z) & 0 & 0\\ 0 & (\beta+\sigma_x)(\beta+\sigma_z) & 0\\ 0 & 0 & (\beta+\sigma_x)(\beta+\sigma_y)\end{pmatrix} v\right] \\
&=(\beta+\sigma_y)(\beta+\sigma_z)\frac{f+f^*+f^\diamond}{3}+(\beta+\sigma_x)(\beta+\sigma_z)\frac{f-2f^*+f^\diamond}{3} \\
&\quad +(\beta+\sigma_x)(\beta+\sigma_y)\frac{f+f^*-2f^\diamond}{3}
\end{aligned}$$

Next we eliminate $v$ to obtain a single equation in terms of $u$:

$$\begin{aligned}
&\rho(\beta+\sigma_x)(\beta+\sigma_y)(\beta+\sigma_z)u \\
&\quad -\operatorname{div}\left[\begin{pmatrix}\mu\frac{(\beta+\sigma_y)(\beta+\sigma_z)}{\beta+\sigma_x} & 0 & 0\\ 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_z)}{\beta+\sigma_y} & 0\\ 0 & 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_y)}{\beta+\sigma_z}\end{pmatrix}\nabla u\right] \\
&=(\beta+\sigma_y)(\beta+\sigma_z)\frac{f+f^*+f^\diamond}{3}+(\beta+\sigma_x)(\beta+\sigma_z)\frac{f-2f^*+f^\diamond}{3} \\
&\quad +(\beta+\sigma_x)(\beta+\sigma_y)\frac{f+f^*-2f^\diamond}{3} \\
&\quad +\operatorname{div}\left[\begin{pmatrix}\mu\frac{(\beta+\sigma_y)(\beta+\sigma_z)}{\beta+\sigma_x} & 0 & 0\\ 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_z)}{\beta+\sigma_y} & 0\\ 0 & 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_y)}{\beta+\sigma_z}\end{pmatrix} f_v\right]
\end{aligned}$$

As a result we obtain a symmetric positive definite system to solve for $U$, like in 2D, which is

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

with

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

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

and

$$
\begin{aligned}
(F_h)_i = \int_\Omega & (\beta+\sigma_y)(\beta+\sigma_z)\frac{(f+f^*+f^\diamond)}{3}\,\varphi_i + (\beta+\sigma_x)(\beta+\sigma_z)\frac{(f-2f^*+f^\diamond)}{3}\,\varphi_i \\
& + (\beta+\sigma_x)(\beta+\sigma_y)\frac{(f+f^*-2f^\diamond)}{3}\,\varphi_i \\
& - \begin{pmatrix} \mu\frac{(\beta+\sigma_y)(\beta+\sigma_z)}{\beta+\sigma_x} & 0 & 0 \\ 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_z)}{\beta+\sigma_y} & 0 \\ 0 & 0 & \mu\frac{(\beta+\sigma_x)(\beta+\sigma_y)}{\beta+\sigma_z} \end{pmatrix} f_v \cdot \nabla\varphi_i \, dx
\end{aligned}
$$

This method makes the treatment of PML region less expensive since the intermediary unknowns $u^*, u^\diamond$ and $v$ are not involved in the linear system to be solved. As in 2D, this result holds when the matrices $D_h$, $D_h^*$, $D_h^\diamond$, $S_h^{xxx}$, $S_h^{xyx}$, $S_h^{xyz}$, $S_h^{yxy}$, $S_h^{yyy}$, $S_h^{yxz}$, $S_h^{zxy}$, $S_h^{zyz}$, $S_h^{zzz}$ are diagonal. It induces that the computational mesh must comprise only hexahedral elements inside the PML areas. It will be the case in our numerical computations. After solving the linear system, $u^*$, $u^\diamond$ and $v$ are then reconstructed using the equations (3.15b), (3.15c) and (3.15d).

### *3.2.2 Hybridizable discontinuous Galerkin (HDG) methods*

As we have seen, the mass matrix $M_h$ is not always diagonal, when the mesh comprises pyramids, prisms or tetrahedra. Moreover, continuous elements based on Gauss-Lobatto points do not allow the use of a variable order approximation (i.e. an order for each element). Hierarchical basis functions (as detailed in [8]) allow variable orders with continuous finite elements, but there is no longer mass lumping. As a consequence, we have considered the use of discontinuous elements, more precisely the Hybridizable discontinuous Galerkin method (denoted HDG see [22, 39, 60, 61]). This method requires slightly more degrees of freedom and computational time, but the mass matrix $M_h$ is block-diagonal (with small blocks) for any kind of mesh and for a variable order of approximation. Moreover the implementation of locally implicit schemes seems to be easier with this method.

In this section we present the variational formulation of the acoustic wave equation using the hybridizable discontinuous Galerkin (HDG) method. For more insight we refer the reader to [11] in which the author presents the HDG method for the simulation of elastodynamic wave equation. In [50] the HDG method is also detailed for the time-domain acoustic wave equation using both explicit and implicit time integration schemes.

The unknowns $u$ and $v$ are discretized using the following finite element spaces

$$
\begin{aligned}
u(t) \in U_h &= \left\{ u \in L^2(\Omega) \text{ such that } u|_{K_i} \circ F_i \in \mathbb{Q}_r \right\} \\
v(t) \in V_h &= \left\{ v \in (L^2(\Omega))^d \text{ such that } v|_{K_i} \circ F_i \in (\mathbb{Q}_r)^d \right\}
\end{aligned}
\tag{3.16}
$$

The basis functions on the reference element $\hat{K}$ are the same as the basis functions considered for continuous elements. They are based on Gauss-Lobatto points for quadrilateral/hexahedral elements. To obtain the variational formulation we multiply the first two equations of (3.1) by the test functions $\varphi_i$ for $u$ and $\psi_i$ for $v$. After integration by parts, we obtain

$$\begin{cases} \dfrac{d}{dt}\displaystyle\int_K \rho\, u\, \varphi_i\, dx + \int_K v \cdot \nabla \varphi_i\, dx - \int_{\partial K} \hat{v} \cdot n\, \varphi_i\, dx = 0 \\ \dfrac{d}{dt}\displaystyle\int_K \mu^{-1} v \cdot \psi_i\, dx + \int_K u\ \mathrm{div}\, \psi_i - \int_{\partial K} \lambda\, \psi_i \cdot n\, dx = 0 \end{cases} \tag{3.17}$$

To make a link between the pressure field $u$ in different edges in 2D (faces in 3D) we introduce an auxiliary unknown $\lambda$. This unknown belongs to the following finite element space:

$$\Lambda_h = \{u \in L^2(\Sigma_h) \text{ such that } u \circ F_i|_{E_i} \in \mathbb{Q}_r\},$$

where $\Sigma_h$ is the set of the mesh edges (faces in 3D) and $E_i$ is an element of $\Sigma_h$. If $E_i$ is a triangular face (in 3-D), $u \circ F_i|_{E_i} \in \mathbb{P}_r$ where $\mathbb{P}_r$ is the space of polynomials of total degree lower or equal to $r$. The basis functions for $\lambda$ are chosen as the trace of the basis functions chosen for $u$ on the boundary. Then, we choose the expression of $\hat{v}$ as follows

$$\hat{v} = v - \tau(u - \lambda)n,$$

with $\tau$ taking the value

$$\tau = \sqrt{\rho\mu}.$$

We notice that $\tau$ has a different value for each edge (or face) when considering a heterogeneous media. A second integration by parts of the second equation of (3.17) and replacing $\hat{v}$ by its expression leads to

$$\begin{cases} \dfrac{d}{dt}\displaystyle\int_K \rho\, u\, \varphi_i\, dx + \int_K v \cdot \nabla \varphi_i\, dx - \int_{\partial K} v \cdot n\, \varphi_i\, dx + \int_{\partial K} \tau(u - \lambda)\, \varphi_i\, dx = 0 \\ \dfrac{d}{dt}\displaystyle\int_K \mu^{-1} v \cdot \psi_i\, dx - \int_K \nabla u \cdot \psi_i + \int_{\partial K} (u - \lambda)\psi_i \cdot n\, dx = 0 \end{cases} \tag{3.18}$$

This second integration by parts is made to symmetrize the final system in $\lambda$. To conclude we impose $\hat{v} \cdot n$ to be a continuous function across each interface. For each internal edge (face in 3-D), we then obtain the following equation:

$$\sum_{K \text{ adjacent to the edge}} \int_{\partial K} (-v \cdot n + \tau(u - \lambda))\, q\, dx = 0$$

with $q$ the test function associated with $\lambda$. For an edge located on the boundary, there is no longer a requirement to be continuous, but a requirement to satisfy a given boundary condition. We detail below how the different boundary conditions are treated.

### Dirichlet's condition

To set the Dirichlet boundary condition, we use the following relation:

$$\lambda = f_D.$$

In such way, the second equation in $v$ becomes (on each boundary element with the Dirichlet's condition $\Gamma_D$)

$$\frac{d}{dt}\int_K \mu^{-1} v \cdot \psi \, dx - \int_K \nabla u \cdot \psi + \int_{\partial K \setminus \Gamma_D} (u - \lambda)\psi \cdot n \, dx = \int_{\Gamma_D} f_D \, \psi \cdot n \, dx.$$

Then to maintain the symmetry of the global system in $\lambda$, the equation on $\Gamma_D$ becomes

$$\int_{\Gamma_D} \tau(u - \lambda)\, q \, dx = 0.$$

### Neumann's condition

For the Neumann boundary condition we impose

$$\hat{v} \cdot n = f_N.$$

The equations in $u$ and $v$ remain unchanged but the equation in $\lambda$ will be written as follows:

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

### Absorbing boundaries condition (ABC)

We use the relation

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

We can notice that $\tau = \sqrt{\rho\,\mu}$ here. Then, the equations in $u$ and $v$ remain the same while the equation in $\lambda$ becomes:

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

### Discrete system

After exploiting each boundary condition, we have obtained the following discrete system

$$\begin{cases} D_h \dfrac{dU}{dt} + (R^T - S_n)V + CU - C_\ell \Lambda = 0 \\ B_h \dfrac{dV}{dt} + (-R + S_n^T)U - S_d^T \Lambda = F_v(t) \\ C_\ell^T U - C_a \Lambda - S_d V = F_\Lambda(t) \end{cases} \tag{3.19}$$

where $F_v(t), F_\Lambda(t)$ are source terms coming from the inhomogeneous boundary conditions. We have the following elementary matrices:

$$(D_h^{\text{elem}})_{i,j} = \int_{K_i} \rho\, \varphi_j\, \varphi_i\, dx$$

$$(B_h^{\text{elem}})_{i,j} = \int_{K_i} \mu^{-1}\, \psi_j \cdot \psi_i\, dx$$

$$R_{i,j}^{\text{elem}} = \int_{K_i} \nabla\varphi_j \cdot \psi_i\, dx$$

$$(S_n)_{i,j}^{\text{elem}} = \int_{\partial K_i} \psi_j \cdot n\, \varphi_i\, dx$$

$$(S_d)_{i,j}^{\text{elem}} = \int_{\partial K_i \setminus \Gamma_D} \psi_j \cdot n\, q_i\, dx$$

$$C_{i,j}^{\text{elem}} = \tau \int_{\partial K_i} \varphi_j \varphi_i\, dx$$

$$(C_\ell^{\text{elem}})_{i,j} = \tau \int_{\partial K_i} q_j \varphi_i\, dx$$

$$(C_a^{\text{elem}})_{i,j} = \tau \int_{\partial K_i} \beta' q_j q_i\, dx$$

where $\beta'$ equals 2 for edges with an absorbing boundary condition and 1 for other edges. The final matrices $D_h, B_h, R, S_n, S_d, C, C_a, C_\ell$ are obtained by assembling the elementary matrices. For any kind of element $K_i$ (tetrahedron, hexahedron, etc), the mass matrices $B_h$ and $D_h$ are block-diagonal, each block being related to the degrees of freedom of an element. The matrix $C_a$ is also block-diagonal, each block being related to degrees of freedom of an edge of the mesh. For an explicit time-stepping algorithm, the matrices $R, S_n, S_d, C_a, C_\ell$ will not be stored, only the matrices $DF_i^{-T}$ will be stored on the quadrature points $\hat{\xi}_k$ and the matrix-vector product will be performed on the fly (see [9] for more details). For quadrilateral/hexahedral elements the matrices $D_h$, $B_h$ and $C_a$ will be diagonal, hence cheap to store. For other elements, they will be more costly to store, a solution proposed by Warburton [76] for curved tetrahedrals which consists in considering the following basis functions

$$\hat{\varphi}_i = \sqrt{J_i}\, \varphi_i \circ F_i$$

in order to have the same block for each tetrahedron. We have preferred to not use this option, since it can deteriorate the convergence for other elements (see [9]). In order to obtain the desired ODE, we use the relation:

$$\Lambda = C_a^{-1} \left( C_\ell^T U - S_d V - F_\Lambda(t) \right) \tag{3.20}$$

to obtain

$$M_h \frac{dX(t)}{dt} + K_h X(t) = F(t)$$

with

$$X(t) = (U, V)^T$$

and

$$M_h = \begin{pmatrix} D_h & 0 \\ 0 & B_h \end{pmatrix}, \qquad K_h = \begin{pmatrix} C - C_\ell C_a^{-1} C_\ell^T & R^T - S_n + C_\ell C_a^{-1} S_d \\ -R + S_n^T - S_d^T C_a^{-1} C_\ell^T & S_d^T C_a^{-1} S_d \end{pmatrix},$$

$$F = \begin{pmatrix} -C_\ell C_a^{-1} F_\Lambda \\ F_v - S_d^T C_a^{-1} F_\Lambda \end{pmatrix}.$$

Of course, these expressions are not used in practice, since the matrix $K_h$ is not stored. For an explicit time stepping, $\Lambda$ is computed with equation (3.20). Then the system (3.19) is used to obtain the product $K_h X$. For an implicit time stepping, we consider the whole system (3.19) (where $d/dt$ is replaced by $\beta$) to obtain the solution $U$ and $V$.

### Static condensation for HDG

For an implicit time scheme, we have to solve the following linear system

$$\begin{cases} \beta D_h U + (R^T - S_n) V + C U - C_\ell \Lambda = F_U \\ \beta B_h V + (-R + S_n^T) U - S_d^T \Lambda = F_V \\ C_\ell^T U - C_a \Lambda - S_d V = F_\Lambda \end{cases} \tag{3.21}$$

On each element, the unknowns $U$ and $V$ can be eliminated by static condensation. If we note

$$X = (U, V)^T, \qquad Y = \Lambda$$

we have the system

$$\begin{cases} A_{11} X + A_{12} Y = F_X \\ A_{21} X + A_{22} Y = F_Y \end{cases}$$

with

$$A_{11} = \begin{pmatrix} \beta D_h + C & R^T - S_n \\ -R + S_n^T & \beta B_h \end{pmatrix}, \quad A_{12} = \begin{pmatrix} -C_\ell \\ -S_d^T \end{pmatrix}, \quad A_{21} = \begin{pmatrix} C_\ell^T & -S_d \end{pmatrix}, \quad A_{22} = -C_a$$

$$F_X = \begin{pmatrix} F_U \\ F_V \end{pmatrix}, \quad F_Y = F_\Lambda.$$

The unknown $X$ is eliminated to obtain the linear system to solve in $\Lambda$

$$(A_{22} - A_{21} A_{11}^{-1} A_{12}) Y = F_Y - A_{21} F_X.$$

In practice, this procedure is performed element by element. For each element, the Schur complement $A_{22}^{\text{elem}} - A_{21}^{\text{elem}}(A_{22}^{\text{elem}})^{-1}A_{12}^{\text{elem}}$ is computed and added to the global matrix $A_h$. The elementary source $F_Y^{\text{elem}} - A_{21}^{\text{elem}}F_X^{\text{elem}}$ is also added to the global right hand side $F_h$. Finally, we have a linear system in $\Lambda$ to solve

$$A_h\Lambda = F_h.$$

**Optimal computation on quadrilateral or hexahedral mesh** In the case of quadrilaterals/hexahedra, the elementary matrices $D_h^{\text{elem}}$, $B_h^{\text{elem}}$, $C^{\text{elem}}$, $C_a^{\text{elem}}$, $C_\ell^{\text{elem}}$, $S_d^{\text{elem}}$ are diagonal, and the matrix $(R - S_n^T)^{\text{elem}}$ is sparse, and not stored. In order to take advantage of the particular structure of these matrices, the computation of Schur complement will be detailed in order to reduce the memory storage. In this paragraph, the superscript "elem" will be omitted. First we express $V$ in terms of $U$ and $\Lambda$ using the second equation of (3.21). We obtain:

$$V = B_h^{-1}\left(F_V + (R - S_n^T)U + S_d^T\Lambda\right).$$

Then we use this expression of $V$ and the first equation of (3.21) to compute $U$ in term of $\Lambda$, we obtain:

$$U = \widetilde{B}^{-1}\left[F_U - (R^T - S_n)B_h^{-1}\left(F_V + S_d^T\Lambda\right) + C_\ell\Lambda\right]$$

with $\widetilde{B}$ given by

$$\widetilde{B} = D_h + (R^T - S_n)B_h^{-1}(R - S_n^T) + C.$$

We observe that $\widetilde{B}$ is a symmetric matrix. We compute and store once for all $\widetilde{B}^{-1}$. We compute $V$ in terms of $\Lambda$ by replacing $U$ by its expression, we get:

$$\begin{aligned} V = \;& B_h^{-1}F_V - B_h^{-1}(-R + S_n^T)\widetilde{B}^{-1}(F_U - (R^T - S_n)B_h^{-1}F_V) \\ & + B_h^{-1}(-R + S_n^T)\widetilde{B}^{-1}\left((R^T - S_n)B_h^{-1}S_d^T - C_\ell\right)\Lambda + B_h^{-1}S_d^T\Lambda. \end{aligned}$$

$U$ and $V$ are then replaced by their expression to obtain the following linear system in terms of $\Lambda$ only:

$$\widetilde{C}\Lambda = F_\Lambda + S_dB_h^{-1}F_V - (C_\ell^T + S_dB_h^{-1}(-R + S_n^T))\widetilde{B}^{-1}\left(F_U - (R^T - S_n)B_h^{-1}F_V\right)$$

with $\widetilde{C}$ defined by

$$\widetilde{C} = -C_a + \left(-C_l^T + S_dB_h^{-1}(R - S_n^T)\right)\widetilde{B}^{-1}\left((R^T - S_n)B_h^{-1}S_d^T - C_l\right) - S_dB_h^{-1}S_d^T.$$

We note that $\widetilde{C}$ is also a symmetric matrix. We don't compute the inverse of $\widetilde{C}$, it is used to construct the global matrix $A_h$.

**Algorithm 1** Algorithm to construct the global matrix $A_h$

**for** i=0, $\cdots$, Mesh element number **do**
  Compute $\widetilde{B} = D_h + (R^T - S_n)B_h^{-1}(R - S_n^T) + C$
  Compute $\widetilde{C} = -C_a + \left(-C_l^T + S_d B_h^{-1}(R - S_n^T)\right)\widetilde{B}^{-1}\left((R^T - S_n)B_h^{-1}S_d^T - C_l\right) - S_d B_h^{-1} S_d^T$
  Add the matrix $\widetilde{C}$ into the global matrix $A_h$
**end for**

**Construction of the global matrix** To construct the global matrix $A_h$, we use the Algorithm 1. It should be noticed that the matrix $\widetilde{B}^{-1}$ will be stored for each quadrilateral/hexahedral. For other elements (e.g. tetrahedra), the blocks $A_{11}^{-1}, A_{12}$ and $A_{21}$ are stored, they require more storage than the block $\widetilde{B}^{-1}$, since the size of $\widetilde{B}^{-1}$ is $N_{dof}$, whereas the size of $A_{11}^{-1}$ is $(d+1)N_{dof}$ where $N_{dof}$ is the number of degrees of freedom for $U$ on the considered element. This algorithm allows us to save memory and also computational time.

**Computation of the RHS terms** For each solution of the linear system (with $F_U, F_V$ and $F_\Lambda$), we use Algorithm 2 to compute $F_h$. We store the vectors $\widetilde{F}_U$ and

**Algorithm 2** Algorithm to compute the RHS $F_h$

$F_h = F_\Lambda$
**for** i=0,$\cdots$, Mesh element number **do**
  Compute and store $\widetilde{F}_V = B_h^{-1} F_V$
  Compute and store $\widetilde{F}_U = F_U - (R^T - S_n)\widetilde{F}_V$
  Compute $\widetilde{G} = \widetilde{B}^{-1}\widetilde{F}_U$
  Compute $\widetilde{F}_\Lambda = S_d(\widetilde{F}_V - B_h^{-1}(S_n^T - R)\widetilde{G}) - C_\ell^T \widetilde{G}$
  Add the block $\widetilde{F}_\Lambda$ to $F_h$
**end for**

$\widetilde{F}_V$ which will replace the vectors $F_U$ and $F_V$.

**Reconstruction of the solution**

Once we have computed $\Lambda$ by solving the system $A_h\Lambda = F_h$, the solution $U$ and $V$ are reconstructed using the Algorithm 3.

## 3.3 Maxwell's equations

In the context of wave propagation problems, Maxwell's equations are a set of equations describing how an electric field and magnetic field interact each other to give rise to an electromagnetic wave. The set of equations has been

**Algorithm 3** Algorithm to compute $U$ and $V$

**for** i=0, $\cdots$, Mesh elements number **do**
Compute $\widetilde{\Lambda} = S_d^T \Lambda$
Compute $U = \widetilde{B}^{-1}\left(\widetilde{F}_U + C_\ell \Lambda - (R^T - S_n) B_h^{-1} \widetilde{\Lambda}\right)$
Compute $V = \widetilde{F}_V + B_h^{-1}\left[(R - S_n^T)U + \widetilde{\Lambda}\right]$
**end for**

put together by James Clerk Maxwell in 1860s, see [53]. Two formulations of Maxwell's equations are commonly used in the literature: the integral form and the differential form. Even though the differential form of these equations is more convenient for numerical simulation, we first present the integral form because physically they are easy to understand. We define $\mathbf{E}$ as the electric field (in volt per meter - $\mathrm{V}\cdot\mathrm{m}^{-1}$), $\mathbf{H}$ as the magnetic field (in Ampère per meter - $\mathrm{A}\cdot\mathrm{m}^{-1}$), $\mathbf{D}$ as the electric flux density (or electric displacement in Coulomb per square meter - $\mathrm{C}\cdot\mathrm{m}^{-2}$) and $\mathbf{B}$ as the magnetic flux density (or magnetic induction in Tesla - T). All the four variables are maps from $\mathbb{R}^3 \times \mathbb{R}$ to $\mathbb{R}^3$. Maxwell's equations read:

$$\int_{\partial V} \mathbf{D}\cdot \mathrm{d}S = \int_V \rho \,\mathrm{d}V, \text{ (Gauss' law)}, \tag{3.22a}$$

$$\int_{\partial V} \mathbf{B}\cdot \mathrm{d}S = 0, \text{ (Gauss' magnetism law)}, \tag{3.22b}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_S \mathbf{B}\cdot \mathrm{d}S\right) = -\int_{\partial S} \mathbf{E}\cdot \mathrm{d}l, \text{ (Faraday's law)}. \tag{3.22c}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_S \mathbf{D}\cdot \mathrm{d}S\right) + \int_{\partial S} \mathbf{J}\cdot \mathrm{d}S = \int_{\partial S} \mathbf{H}\cdot \mathrm{d}l, \text{ (Ampère's law)}, \tag{3.22d}$$

where $\mathbf{J}(\mathbf{x}, t)$, $(\mathbf{x}, t) \in \mathbb{R}^3 \times \mathbb{R}$ represents the current density (in $\mathrm{A}\cdot\mathrm{m}^{-2}$ SI unit) and $\rho(\mathbf{x}, t)$, $(\mathbf{x}, t) \in \mathbb{R}^3 \times \mathbb{R}$ denotes the charge (in $\mathrm{C}\cdot\mathrm{m}^{-3}$ SI unit) in the volume $V \subset \mathbb{R}^3$. $S$ is a surface of $\mathbb{R}^3$, $\partial V$ and $\partial S$ are respectively the boundaries of the volume $V$ and the surface $S$.

The first equation known as the Gauss law, states that the electric flux in a volume $V$ is proportional to the amount of charge inside. Gauss' law is basically equivalent to the force equation for electric charges (opposite-sign charges attract and same-sign charges repel each other). The second equation which is the Gauss' magnetism law states that the magnetic flux through a closed surface is zero (the divergence of the magnetic flux is zero). This means that there are no magnetic monopoles. In numerical simulation these first two equations are generally considered as initial data obtained from experimental measurement. We will be more focused on the last two equations. Roughly speaking, the third equation (3.22c), the Faraday's law, states that the voltage induced in a closed loop $\partial S$ (the sum of the electric field along the closed loop $\partial S$) gives rise to a

magnetic flux over the surface $S$ changing with time and vice versa. Concerning the fourth equation (3.22d), known as Ampère's law, James Clerk Maxwell added a new term (the displacement current) with the current density which allowed to put together these four laws and formed Maxwell's equations [70]. (3.22d) means that a flowing electric current plus the rate of change of electric field gives rise to a magnetic flux around the closed loop $\partial S$ and vice versa.

Let us now recall the following standard differential operators on a smooth field $\mathbf{v}$:

$$\operatorname{div}\mathbf{v} = \sum_{i=1}^{n} \partial_{x_i} v_i \quad \text{and} \quad \operatorname{curl}\mathbf{v} = (\partial_{x_2} v_3 - \partial_{x_3} v_2, \partial_{x_3} v_1 - \partial_{x_1} v_3, \partial_{x_1} v_2 - \partial_{x_2} v_1)$$

To derive the differential form of Maxwell's equation from (3.22), we use the Stokes-Ostrogradski formulas

$$\int_S \operatorname{curl}\mathbf{F}\cdot \mathrm{d}S = \int_{\partial S} \mathbf{F}\cdot \mathrm{d}l \quad \text{and} \quad \int_V \operatorname{div}\mathbf{F}\mathrm{d}V = \int_{\partial V} \mathbf{F}\cdot \mathrm{d}S, \tag{3.23}$$

for any surface $S$, any volume $V$ and a given vector field $\mathbf{F}$. The differential form of the Maxwell's equations is then given by

$$\operatorname{div}\mathbf{D} = \rho, \text{ (Gauss' law)}, \tag{3.24a}$$

$$\operatorname{div}\mathbf{B} = 0, \text{ (Gauss' magnetism law)}, \tag{3.24b}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \operatorname{curl}\mathbf{E} = 0, \text{ (Faraday's law)}. \tag{3.24c}$$

$$\frac{\partial \mathbf{D}}{\partial t} - \operatorname{curl}\mathbf{H} = -\mathbf{J}, \text{ (Ampère's law)}. \tag{3.24d}$$

For deeper understanding of the physics of Maxwell's equations we refer the reader to [70], [18] and [21].

The following result holds for the conservation of charge:

$$\frac{\partial \rho}{\partial t} + \operatorname{div}\mathbf{J} = 0, \tag{3.25}$$

which is obtained by differentiating (3.24a) with respect to the time, plugging this result into (3.24d) and noticing that $\operatorname{div}(\operatorname{curl}\mathbf{E}) = 0$.

### *3.3.1 The electromagnetic wave propagation problems*

#### The constitutive relations

As we have described in (3.24), Maxwell's equations are not sufficient to uniquely determine the electromagnetic field [57]. We need to introduce the constitutive equations to couple the fields $\mathbf{E}$, $\mathbf{H}$, $\mathbf{D}$ and $\mathbf{B}$. Those relations describe the

physics of the media where the electric and magnetic fields evolve. They are expressed as

$$\mathbf{D} = \mathbf{D}(\mathbf{E}, \mathbf{H}) \text{ and } \mathbf{B} = \mathbf{B}(\mathbf{E}, \mathbf{H}). \tag{3.26}$$

These relations can be extremely complex depending on the media. In this section we present some examples like in the vacuum or free space [57] where the fields are given by

$$\mathbf{D} = \epsilon_0 \mathbf{E} \text{ and } \mathbf{B} = \mu_0 \mathbf{H}, \tag{3.27}$$

with $\epsilon_0 \approx 8.854 \cdot 10^{-12}$ (Farads per meter - $\mathrm{F} \cdot \mathrm{m}^{-1}$) defined as the permittivity and $\mu_0 = 4\pi \cdot 10^{-7}$ (Henries per meter - H/m) as the permeability. These constants are linked with the speed of light in the vacuum trough the relation

$$c_0 = \frac{1}{\sqrt{\epsilon_0 \mu_0}}, \text{ in } \mathrm{m} \cdot \mathrm{s}^{-1}. \tag{3.28}$$

We can consider a little more complicated case of a heterogeneous and anisotropic media. This is given by the relation

$$\mathbf{D} = \underline{\underline{\epsilon}}\mathbf{E} \text{ and } \mathbf{B} = \underline{\underline{\mu}}\mathbf{H}, \tag{3.29}$$

where $\underline{\underline{\epsilon}}$ and $\underline{\underline{\mu}}$ are matrix-tensor functions from $\mathbb{R}^3$ to $\mathbb{R}^3 \times \mathbb{R}^3$. Most of the time we will be dealing with a heterogeneous and isotropic medium (there is no preferred direction). In such medium, the functions $\underline{\underline{\epsilon}}$ and $\underline{\underline{\mu}}$ are replaced by scalar valued functions $\epsilon$ and $\mu$ from $\mathbb{R}^3$ to $\mathbb{R}$. The Maxwell's equations are then given by

$$\mathrm{div}(\epsilon \mathbf{E}) = \rho, \tag{3.30a}$$

$$\mathrm{div}(\mu \mathbf{H}) = 0, \tag{3.30b}$$

$$\mu \frac{\partial \mathbf{H}}{\partial t} + \mathbf{curl}\, \mathbf{E} = 0, \tag{3.30c}$$

$$\epsilon \frac{\partial \mathbf{E}}{\partial t} - \mathbf{curl}\, \mathbf{H} = -\mathbf{J}. \tag{3.30d}$$

We decompose the current density as follows

$$\mathbf{J} = \mathbf{J}_c + \mathbf{J}_s,$$

where $\mathbf{J}_c$ is the current density related to the medium and $\mathbf{J}_s$ is considered as an external current source. When the medium has an electric conductivity given by $\sigma : \mathbb{R}^3 \to \mathbb{R}$, which is measured in Siemens per meter, Ohm's law allows to write

$$\mathbf{J} = \sigma \mathbf{E} + \mathbf{J}_s,$$

since $\mathbf{J}_c = \sigma \mathbf{E}$. In fact, the conductivity $\sigma$ is the inverse of resistance. The electric field $\mathbf{E}$ and the medium current density $\mathbf{J}_c$ are respectively analogous to the voltage and the current in a circuit.

### Mathematical problems

To study the time evolution of Maxwell's equations in a bounded domain $\Omega$, we need to introduce initial and boundary conditions. A good choice of the initial conditions is helpful to satisfy the equations (3.30a) and (3.30b). In fact, we observe that the divergence of (3.30c) and (3.30d) leads to

$$\frac{\partial}{\partial t}\,\mathrm{div}(\mu\mathbf{H}) = 0 \quad \text{and} \quad \frac{\partial}{\partial t}\,\mathrm{div}(\epsilon\mathbf{E}) = 0, \tag{3.31}$$

where we have assumed that $\mathrm{div}(\mathbf{J}) = 0$, because $\mathrm{div}(\mathrm{curl}(\mathbf{U}) = 0$ for any vector function $\mathbf{U}$. As a consequence, we assume that

$$\frac{\partial}{\partial t}\,\mathrm{div}(\mu\mathbf{H_0}) = 0 \quad \text{and} \quad \frac{\partial}{\partial t}\,\mathrm{div}(\epsilon\mathbf{E_0}) = 0, \quad \text{at}\, t = 0, \tag{3.32}$$

and $\mathrm{div}(\mathbf{J}) = 0$. Then (3.30c) and (3.30d) state that (3.30a) and (3.30b) will be satisfied at each time $t \geq 0$.

On the boundary $\partial\Omega$ of the domain $\Omega$, there are several conditions that can be considered. As an example we consider the case where the medium is surrounded by a perfect conductor, which corresponds to the homogeneous Dirichlet boundary condition. Then, the expression of the boundary conditions reads:

$$\mathbf{E} \times \mathbf{n} = 0 \quad \text{and} \quad \mathbf{n}\cdot(\mu\mathbf{H}) = 0, \tag{3.33}$$

where $n$ denotes the outward unit normal to $\partial\Omega$.

Taking into account the initial conditions, the boundary conditions and considering a conducting medium the evolution problems for the Maxwell's equations becomes

$$\begin{cases} \mu\dfrac{\partial \mathbf{H}}{\partial t} + \mathbf{curl}\,\mathbf{E} = 0, \quad \text{in } \Omega \times \mathbb{R}^+ \\ \epsilon\dfrac{\partial \mathbf{E}}{\partial t} + \sigma\mathbf{E} - \mathbf{curl}\,\mathbf{H} = -\mathbf{J}_s, \quad \text{in } \Omega \times \mathbb{R}^+ \\ \mathbf{E} \times \mathbf{n} = 0 \text{ and } \mathbf{n}\cdot(\mu\mathbf{H}) = 0, \quad \text{on } \partial\Omega \times \mathbb{R}^+ \\ \mathbf{E}(\mathbf{x}, 0) = \mathbf{E}_0(\mathbf{x}) \text{ and } \mathbf{H}(\mathbf{x}, 0) = \mathbf{H}_0(\mathbf{x}), \quad \mathbf{x} \in \Omega. \end{cases} \tag{3.34}$$

### The 2D problems

We define the rotational of a 2D vector field $\mathbf{v} \in \mathbb{R}^2$ and a scalar field $u \in \mathbb{R}$ as follows:

$$curl\,\mathbf{v} = \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \quad \text{and} \quad \mathbf{curl}\,u = \left(\frac{\partial u}{\partial y}, -\frac{\partial u}{\partial x}\right). \tag{3.35}$$

To describe Maxwell's equations in 2D, we consider two possibilities. The first one consists of taking $\mathbf{E} = (E_x, E_y) \in \mathbb{R}^2$ and $\mathbf{H} = H_z \in \mathbb{R}$. In this case the electric field lies in the plane of propagation; it is called transverse-magnetic (TM)

polarization. The second case, which is called the transverse-electric (TE) polarization, the magnetic field $\mathbf{H} = (H_x, H_z) \in \mathbb{R}^2$ lies in the plane of propagation and the electric field $\mathbf{E} = E_z \in \mathbb{R}$ is perpendicular to this plane. In both cases TM or TE polarizations, solving the Maxwell's equations in 2D is equivalent to solve the acoustic wave equation (3.1). So, in 1D and 2D we will consider only the acoustic wave equation presented in its first order formulation (3.1).

### 3.3.2 Edge elements for Maxwell's equations

Let $\Omega \subset \mathbb{R}^n$ be an open set. We let $L^2(\Omega, \mathbb{R}^n)$ denote the space of square-integrable vector fields of $\Omega$ with the associated norm $\|\cdot\|$ and the inner-product $<\cdot,\cdot>$. We define the following Sobolev spaces:

$$H(\mathbf{div}; \Omega) = \{\mathbf{v} \in L^2(\Omega, \mathbb{R}^n) : \mathbf{div}\, \mathbf{v} \in L^2(\Omega, \mathbb{R}^1)\},$$
$$H(\mathbf{curl}; \Omega) = \{\mathbf{v} \in L^2(\Omega, \mathbb{R}^3) : \mathbf{curl}\, \mathbf{v} \in L^2(\Omega, \mathbb{R}^3)\}$$

with derivatives taken in the distributional sense. For more details on the definitions and properties of these spaces we refer the reader to [57].
In this section we present the variational formulation of Maxwell's equations without PML and with PML. In order to avoid confusion with the damping coefficient $\sigma$ in the PML, in the reminder of this chapter we replace the conductivity $\sigma$ by $\widetilde{\sigma}$ in the Maxwell's equations. The reader can find more details on the mixed spectral finite elements method for the time-harmonic Maxwell's equations in [31].

#### Case without PML

We consider Maxwell's equation in 3D with Dirichlet, Neumann and Silver-Müler conditions

$$\begin{cases} \mu \dfrac{\partial \mathbf{H}}{\partial t} + \mathbf{curl}\ \mathbf{E} = 0 \\ \varepsilon \dfrac{\partial \mathbf{E}}{\partial t} + \widetilde{\sigma}\mathbf{E} - \mathbf{curl}\ \mathbf{H} = 0 \\ \mathbf{E}(x,0) = \mathbf{H}(x,0) = 0, \quad \forall x \in \Omega \quad \text{(initial conditions)} \\ \mathbf{n} \times (\mathbf{E} \times \mathbf{n}) = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\ n \times \mathbf{H} = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\ \mathbf{n} \times \mathbf{H} + \sqrt{\dfrac{\varepsilon}{\mu}}(\mathbf{n} \times \mathbf{E}) \times \mathbf{n} = f_A, \quad x \in \Gamma_A \quad \text{(Silver-Müller condition)} \end{cases} \tag{3.36}$$

The computational domain $\Omega$ is meshed using tetrahedra, prisms, pyramids and hexahedrons:

$$\Omega = \bigcup K_i$$

where $K_i$ is an element of the mesh. We use the following finite elements space for $E$ and $H$ (for hexahedrons):

$$U_h = \left\{ u \in H(\mathbf{curl}, \Omega) \text{ such that } DF_i^T \, u \circ F_i \in \mathbb{Q}_{r-1,r,r} \times \mathbb{Q}_{r,r-1,r} \times \mathbb{Q}_{r,r,r-1} \right\}$$

$$V_h = \left\{ v \in (L^2(\Omega))^3 \text{ such that } v \circ F_i \in (\mathbb{Q}_r)^3 \right\}$$

Here we have chosen the first family of Nédélec's elements which allows to avoid spurious modes (see thesis [31]). We note $\varphi_i$ the basis functions associated with the space $U_h$ and $\psi_i$ the basis functions associated with the space $V_h$. For the hexahedrons, the basis functions $\varphi_i$ are based on the Gauss-Legendre and Gauss-Lobatto quadrature nodes (details are given in [31]). We obtain the followings relations:

$$DF_i^{-T} \hat{\varphi}_i = \varphi_i \circ F_i, \quad \hat{\psi}_i = \psi_i \circ F_i$$

where $DF_i^{-T}$ is the Piola transformation map. This transformation is used to ensure the continuity of the induced tangential electric field $\mathbf{E}$ between two elements in the mesh to make sure that $\mathbf{E} \in H(\mathbf{curl}, \Omega)$. For pyramids, prisms and tetrahedra, we choose basis functions based on nodal points as presented in [8]. The choice of basis functions $\psi_i$, are based on quadrature points $\hat{\xi}_k$ such that:

$$\hat{\psi}_i^1(\hat{\xi}_k) = \delta_{i,k} \, e_x, \quad \hat{\psi}_i^2(\hat{\xi}_k) = \delta_{i,k} \, e_y, \quad \hat{\psi}_i^3(\hat{\xi}_k) = \delta_{i,k} \, e_z$$

To obtain the variational formulation we multiply the first two equations of (3.36) by the basis functions and integrate by parts the first equation:

$$\begin{cases} \dfrac{d}{dt} \displaystyle\int_\Omega \varepsilon \, \mathbf{E} \cdot \varphi_i \, dx + \int_\Omega \widetilde{\sigma} \mathbf{E} \cdot \varphi_i \, dx - \int_\Omega \mathbf{H} \cdot \mathbf{curl} \ \varphi_i \, dx \\ \qquad\qquad + \displaystyle\int_{\Gamma_A} \sqrt{\frac{\epsilon}{\mu}} \, \mathbf{E} \times \mathbf{n} \cdot \varphi_i \times \mathbf{n} \, dx = \int_\Gamma f \cdot \varphi_i \, dx \\ \dfrac{d}{dt} \displaystyle\int_\Omega \mu \mathbf{H} \cdot \psi_i \, dx + \int_\Omega \mathbf{curl} \ \mathbf{E} \cdot \psi_i \, dx = 0 \end{cases} \tag{3.37}$$

where $f$ depends on the sources $f_N$ and $f_A$.
The semi-discrete system then reads:

$$\begin{cases} D_h \dfrac{d\mathbf{E}}{dt} - R_h \mathbf{H} + S_h \mathbf{E} = F \\ B_h \dfrac{d\mathbf{H}}{dt} + R_h^T \mathbf{E} = 0 \end{cases}$$

with

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

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

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

$$(S_h)_{i,j} = \int_\Omega \widetilde{\sigma}\varphi_j \cdot \varphi_i \, dx + \int_{\Gamma_A} \sqrt{\frac{\varepsilon}{\mu}} \, (\varphi_j \times \mathbf{n}) \cdot (\varphi_i \times \mathbf{n}) \, dx$$

Since the basis functions are associated with the quadrature nodes, the obtained matrix $B_h$ will be diagonal in the isotropic case ($\mu$ isotropic). In fact,

$$\int_{K_e} \mu \, \psi_j^\ell \cdot \psi_i^m = \mu_{\ell,m} \, \omega_i \, J_e(\hat{\xi}_i) \, \delta_{i,j}$$

In the anisotropic case, the matrix $B_h$ is block diagonal, each block being a $3 \times 3$ matrix. The stiffness matrix in each element is given by

$$\int_{K_e} \psi_j^\ell \cdot \mathbf{curl}\ \varphi_i \, dx = \omega_j \, DF_e(\hat{\xi}_j) \, \hat{\mathbf{curl}} \, \hat{\varphi}_i(\hat{\xi}_j) \cdot e_\ell$$

The stiffness matrix $R_h$ is not stored, the matrix-vector product is performed on the fly by using Jacobian matrices $DF_e$ computed on quadrature points. The mass matrix $D_h$ is not diagonal (except for rectangular elements), therefore this discretization of Maxwell's equations will only be used for implicit time-stepping where it is not so crucial to have mass lumping.

### Static condensation

When we use implicit time schemes, we will have to solve the following linear system

$$\begin{cases} \beta D_h \mathbf{E} - R_h \mathbf{H} + S_h \mathbf{E} = F_\mathbf{E} \\ \beta B_h \mathbf{H} + R_h^T \mathbf{E} = F_\mathbf{H} \end{cases} \tag{3.38}$$

where $\beta$ is a constant depending on the time scheme. We extract $\mathbf{H}$ from the second equation and substitute the result into the first equation to obtain:

$$\beta^2 D_h \mathbf{E} + R_h B_h^{-1} R_h^T \mathbf{E} + \beta S_h \mathbf{E} = \beta F_\mathbf{E} - R_h B_h^{-1} F_\mathbf{H}$$

We note

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

$$F_h = \beta F_E - R_h B_h^{-1} F_\mathbf{H}$$

The global matrix $K_h$ is formed using the following elementary matrices:

$$\sum_k \omega_k \frac{1}{J_e(\hat{\xi}_k)} \mu^{-1}(\hat{\xi}_k) \left[ DF_e(\hat{\xi}_k) \, \hat{\mathbf{curl}} \, \hat{\varphi}_i(\hat{\xi}_k) \right] \cdot \left[ DF_e(\hat{\xi}_k) \, \hat{\mathbf{curl}} \, \hat{\varphi}_j(\hat{\xi}_k) \right]$$

We deduce that

$$K_h = \int_\Omega \mu^{-1} \, \mathbf{curl}\ \varphi_j \cdot \mathbf{curl}\ \varphi_i \, dx$$

This equality comes from the fact that we have chosen basis functions $\psi_i$ based on quadrature nodes. For this reason the equality is true for any element $K_i$ (tetrahedron, prism, pyramid or hexahedron). Finally the linear system to be solved at each iteration is symmetric and positive definite:

$$(\beta^2 D_h + \beta S_h \mathbf{E} + K_h)\mathbf{E} = F_h.$$

In addition, when we increase the finite element order, the interior degrees of freedom of each element can be removed by static condensation for the system in $\mathbf{E}$.

### Case with PML

We consider the modified Maxwell's equations with PML formulation (see [65]):

$$\begin{cases} \mu \dfrac{\partial \mathbf{H}^*}{\partial t} + \mu\, T_{2,3,1}\mathbf{H}^* + \textbf{curl } \mathbf{E} = 0 \\ \varepsilon \dfrac{\partial \mathbf{E}^*}{\partial t} + \varepsilon\, T_{2,3,1}\mathbf{E}^* - \textbf{curl } \mathbf{H} = 0 \\ \dfrac{\partial \mathbf{H}}{\partial t} + T_{3,1,2}\mathbf{H} - \dfrac{\partial \mathbf{H}^*}{\partial t} - T_{1,2,3}\mathbf{H}^* = 0 \\ \dfrac{\partial \mathbf{E}}{\partial t} + T_{3,1,2}\mathbf{E} - \dfrac{\partial \mathbf{E}^*}{\partial t} - T_{1,2,3}\mathbf{E}^* = 0 \end{cases} \tag{3.39}$$

where $T_{i,j,k}$ are damping tensors given by

$$T_{i,j,k} = \begin{pmatrix} \sigma_i & 0 & 0 \\ 0 & \sigma_j & 0 \\ 0 & 0 & \sigma_k \end{pmatrix}$$

with the following damping function

$$\sigma_1 = \sigma_x = \frac{3\,\log 1000}{2a^3}(x - x_0)^2\,\sigma\; v\mathbf{max}$$

$$\sigma_2 = \sigma_y = \frac{3\,\log 1000}{2a^3}(y - y_0)^2\,\sigma\; v\mathbf{max}$$

$$\sigma_3 = \sigma_z = \frac{3\,\log 1000}{2a^3}(z - z_0)^2\,\sigma\; v\mathbf{max}$$

We note that the modified Maxwell system can be considered as the classical system with source terms. The physical unknowns $\mathbf{E}$ and $\mathbf{H}$ are solutions to the Maxwell's equations

$$\begin{cases} \mu \dfrac{\partial \mathbf{H}}{\partial t} + \textbf{curl } \mathbf{E} = 0 \\ \varepsilon \dfrac{\partial \mathbf{E}}{\partial t} + \widetilde{\sigma}\mathbf{E} - \textbf{curl } \mathbf{H} = 0 \end{cases}$$

in the physical domain $\Omega$. Here $\widetilde{\sigma}$ which represents the medium conductivity is assumed to be equal to zero in the PML region. Since we have $\varepsilon\mathbf{E}$ and $\mu\mathbf{H}$ in the physical domain which are different from $\varepsilon\mathbf{E}^*$ and $\mu\mathbf{H}^*$ in the PML region, the mass terms will differ. To unify the mass terms, we propose to choose the following unknowns:

$$\hat{\mathbf{E}} = \begin{cases} \mathbf{E} \text{ in the physical domain} \\ \mathbf{E}^* \text{ in the PML region} \end{cases}, \quad \hat{\mathbf{E}}^* = \begin{cases} \text{not defined in the physical domain} \\ \mathbf{E} \text{ in the PML region} \end{cases}$$

By the same way we define $\hat{\mathbf{H}}$ and $\hat{\mathbf{H}}^*$. Since the damping functions $\sigma_1, \sigma_2, \sigma_3$ are vanishing on the interface between the physical domain and the PML region, we have $\mathbf{E} = \mathbf{E}^*, \mathbf{H} = \mathbf{H}^*$ on this interface. This implies that $\hat{\mathbf{E}}, \hat{\mathbf{H}} \in H(\text{curl})$ and (3.39) along with the new unknown reads

$$\begin{cases} \mu\dfrac{\partial\hat{\mathbf{H}}}{\partial t} + \mu\, T_{2,3,1}\hat{\mathbf{H}} + \textbf{curl}\ \hat{\mathbf{E}}^* = 0 \\ \varepsilon\dfrac{\partial\hat{\mathbf{E}}}{\partial t} + \varepsilon\, T_{2,3,1}\hat{\mathbf{E}} - \textbf{curl}\ \hat{\mathbf{H}}^* = 0 \\ \dfrac{\partial\hat{\mathbf{H}}^*}{\partial t} + T_{3,1,2}\hat{\mathbf{H}}^* - \dfrac{\partial\hat{\mathbf{H}}}{\partial t} - T_{1,2,3}\hat{\mathbf{H}} = 0 \\ \dfrac{\partial\hat{\mathbf{E}}^*}{\partial t} + T_{3,1,2}\hat{\mathbf{E}}^* - \dfrac{\partial\hat{\mathbf{E}}}{\partial t} - T_{1,2,3}\hat{\mathbf{E}} = 0 \end{cases} \tag{3.40}$$

Only the first two equations hold in the physical domain (and $\hat{\mathbf{E}}^*$, $\hat{\mathbf{H}}^*$ are replaced by $\mathbf{E}$ and $\mathbf{H}$). For notational convenience we will rather use $\mathbf{E}$, $\mathbf{H}$, $\mathbf{E}^*$ and $\mathbf{H}^*$ instead of $\hat{\mathbf{E}}$, $\hat{\mathbf{H}}$, $\hat{\mathbf{E}}^*$ and $\hat{\mathbf{H}}^*$.

### Static condensation

When using an implicit time scheme the linear to solve takes the form:

$$\begin{cases} \beta\,\mu\,\mathbf{H} + \mu\, T_{2,3,1}\mathbf{H} + \textbf{curl}\ \mathbf{E}^* = f_{\mathbf{H}} \\ \beta\,\varepsilon\,\mathbf{E} + \varepsilon\, T_{2,3,1}\mathbf{E} - \textbf{curl}\ \mathbf{H}^* = f_{\mathbf{E}} \\ \beta\,\mathbf{H}^* + T_{3,1,2}\mathbf{H}^* - \beta\mathbf{H} - T_{1,2,3}\mathbf{H} = f_{\mathbf{H}^*} \\ \beta\,\mathbf{E}^* + T_{3,1,2}\mathbf{E}^* - \beta\mathbf{E} - T_{1,2,3}\mathbf{E} = f_{\mathbf{E}^*} \end{cases}$$

From the third equation we obtain

$$\mathbf{H}^* = \frac{\beta + T_{1,2,3}}{\beta + T_{3,1,2}}\mathbf{H} + \frac{f_{\mathbf{H}*}}{\beta + T_{3,1,2}}$$

in which we plug the value

$$\mathbf{H} = \frac{\mu^{-1}}{\beta + T_{2,3,1}}\left(f_{\mathbf{H}} - \textbf{curl}\ \mathbf{E}^*\right)$$

to obtain

$$\mathbf{H}^* = \mu^{-1}\frac{(\beta + T_{1,2,3})}{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}\left(f_{\mathbf{H}} - \mathbf{curl}\ \mathbf{E}^*\right) + \frac{f_{\mathbf{H}^*}}{\beta + T_{3,1,2}}$$

Next we extract $\mathbf{E}$ from the fourth equation as follows

$$\mathbf{E} = \frac{\beta + T_{3,1,2}}{\beta + T_{1,2,3}}\mathbf{E}^* - \frac{f_{\mathbf{E}^*}}{\beta + T_{1,2,3}}$$

Finally we obtain the following system that has to be solved for $\mathbf{E}^*$ only:

$$\begin{aligned}\varepsilon\frac{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}{(\beta + T_{1,2,3})}\mathbf{E}^* + \mathbf{curl}\left(\mu^{-1}\frac{(\beta + T_{1,2,3})}{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}\,\mathbf{curl}\,\mathbf{E}^*\right)\\ = \ f_{\mathbf{E}} + \varepsilon\frac{(\beta + T_{2,3,1})}{(\beta + T_{1,2,3})}f_{\mathbf{E}^*} + \mathbf{curl}\left(\frac{f_{\mathbf{H}^*}}{\beta + T_{3,1,2}} + \mu^{-1}\frac{(\beta + T_{1,2,3})}{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}f_{\mathbf{H}}\right)\end{aligned} \tag{3.41}$$

In the discrete level, the linear system to be solved takes the form

$$(M_h + K_h)\mathbf{E}^* = F_h \tag{3.42}$$

with the mass matrix given by

$$(M_h)_{i,j} = \int_\Omega \widetilde{\varepsilon}\,\varphi_j \cdot \varphi_i\,dx$$

and the stiff matrix given by

$$(K_h)_{i,j} = \int_\Omega \widetilde{\mu}^{-1}\,\mathbf{curl}\ \varphi_j \cdot \mathbf{curl}\ \varphi_i\,dx$$

The values of $\widetilde{\varepsilon}$ and $\widetilde{\mu}$ are computed as follows

$$\widetilde{\varepsilon} = \varepsilon\frac{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}{(\beta + T_{1,2,3})}, \quad \widetilde{\mu} = \mu\frac{(\beta + T_{3,1,2})(\beta + T_{2,3,1})}{(\beta + T_{1,2,3})}$$

and the source is given by

$$F_h = \int_\Omega \left(f_{\mathbf{E}} + \varepsilon\frac{(\beta + T_{2,3,1})}{(\beta + T_{1,2,3})}f_{\mathbf{E}^*}\right)\cdot\varphi_i + \left(\frac{f_{\mathbf{H}^*}}{\beta + T_{3,1,2}} + \widetilde{\mu}^{-1}f_{\mathbf{H}}\right)\cdot\mathbf{curl}\,\varphi_i\,dx$$

With this method we obtain a symmetric positive definite linear system. The system is solved only on $\mathbf{E}^*$. Then we reconstruct the others unknowns $\mathbf{E}$, $\mathbf{H}$, $\mathbf{H}^*$ from their expressions.

### 3.3.3 Hybridizable discontinuous Galerkin (HDG) methods

As discussed previously, the mass matrix $D_h$ is neither diagonal nor block-diagonal when edge elements of Nédélec's first family are used. That is why, we have investigated the use of HDG formulation that allows to obtain a block-diagonal mass matrix for any kind of mesh and with variable orders of approximation.

We briefly present here the variational formulation of Maxwell's equations (3.36) in 3D and recall how boundary conditions are treated. The reader can find more details on the HDG method for the time-domain 3D Maxwell's equations in [20]. In [51] it is also detailed for the time-harmonic Maxwell's equations.

Contrary to the previous subsection, Nédélec's second family is chosen for hexahedral elements:

$$\mathbf{E}, \mathbf{H} \in U_h = \left\{ u \in \mathbf{H}(\mathbf{curl}, \Omega) \text{ such that } DF_i^T \, u \circ F_i \in (\mathbb{Q}_r)^3 \right\}$$

For other elements, we will choose the same space as used for acoustics (hence without the Piola transform $DF_i^T$). For example, the space for tetrahedral elements will be

$$\mathbf{E}, \mathbf{H} \in U_h = \left\{ u \in H(\mathbf{curl}, \Omega) \text{ such that } \; u \circ F_i \in (\mathbb{P}_r)^3 \right\}$$

The use of the second Nédélec's family is less problematic than with edge elements since the HDG formulation will be equivalent to Local Discontinuous Galerkin formulation with upwind fluxes. With upwind fluxes, the spurious modes are attenuated (see illustrations in [31]). The main advantage is that the mass matrices will be block-diagonal (with blocks of size $3 \times 3$) for hexahedral elements. We multiply the first two equations of (3.36) by a function test $\varphi_i$ or $\psi_i$. From integration by parts we obtain:

$$\left\{ \begin{array}{l} \dfrac{d}{dt} \displaystyle\int_K \varepsilon \, \mathbf{E} \cdot \varphi_i \, dx + \int_K \widetilde{\sigma} \, \mathbf{E} \cdot \varphi_i \, dx - \int_K \mathbf{H} \cdot \mathbf{curl} \, \varphi_i \, dx - \int_{\partial K} \mathbf{n} \times \hat{\mathbf{H}} \cdot \varphi_i \, dx = 0 \\ \dfrac{d}{dt} \displaystyle\int_K \mu \mathbf{H} \cdot \psi_i \, dx + \int_K \mathbf{E} \cdot \mathbf{curl} \, \psi_i + \int_{\partial K} \mathbf{n} \times \lambda \cdot \psi_i \, dx = 0 \end{array} \right. \tag{3.43}$$

where $\hat{\mathbf{H}}$ is the numerical trace and $\lambda$ is an intermediary unknown. This unknown is discretized with Nédélec's second family of elements:

$$\Lambda_h = \left\{ u \in L^2(\Sigma_h) \text{ such that } DF_i^T u \circ F_i|_{E_i} \in \begin{array}{c} (\mathbb{Q}_r)^2 \text{ if } E_i \text{ quadrangle,} \\ (\mathbb{P}_r)^2 \text{ if } E_i \text{ triangle} \end{array} \right\},$$

The Piola transform is introduced to ensure the tangential continuity of the basis functions $q_i$ used for $\lambda$ across the two adjacent elements of each face. Let $\tau$ be a local stabilization parameter given by

$$\tau = \sqrt{\frac{\varepsilon}{\mu}}.$$

Notice that in heterogeneous medium the value of $\tau$ will be different on each face. Then we consider numerical traces $\hat{\mathbf{H}}$ of the form:

$$\hat{\mathbf{H}} = \mathbf{H} - \tau \mathbf{n} \times (\lambda - \mathbf{E}).$$

From a second integration by parts of the second stage of (3.43) and replacing $\hat{\mathbf{H}}$ by its expression, we obtain

$$\begin{cases} \dfrac{d}{dt}\displaystyle\int_K \varepsilon\, \mathbf{E}\cdot\varphi_i\, dx + \int_K \widetilde{\sigma}\, \mathbf{E}\cdot\varphi_i\, dx - \int_K \mathbf{H}\cdot \mathbf{curl}\,\varphi_i\, dx - \int_{\partial K} \mathbf{n}\times\mathbf{H}\cdot\varphi_i\, dx \\ \qquad\qquad + \displaystyle\int_{\partial K} \tau\left((\mathbf{E}-\lambda)\times\mathbf{n}\right)\cdot\left(\varphi_i\times\mathbf{n}\right)\, dx = 0 \\ \dfrac{d}{dt}\displaystyle\int_K \mu\mathbf{H}\cdot\psi_i\, dx + \int_K \mathbf{curl}\,\mathbf{E}\cdot\psi_i + \int_{\partial K} \mathbf{n}\times(\lambda-\mathbf{E})\cdot\psi_i\, dx = 0 \end{cases} \tag{3.44}$$

Even though the integrals are approximated numerically, the second integration by parts allows to obtain a symmetric system in $\lambda$. We observe that (3.44) is a system of two equations with three unknowns $\mathbf{E}, \mathbf{H}$ and $\lambda$. To have the third equation, we impose $\hat{\mathbf{H}} \times n$ to be continuous along each interface (called conservative condition). For an internal face of the mesh, the conservative condition is satisfied by considering the following equation

$$\sum_{K \text{ adjacent to the face}} \int_{\partial K} (\mathbf{H} - \tau\mathbf{n}\times(\lambda-\mathbf{E}))\cdot(\mathbf{n}\times q_i)\, dx = 0. \tag{3.45}$$

In (3.45), $q_i$ represents the test function associated with $\lambda$. For an external face, we use the boundary condition as described below. (3.44) and (3.45) represent the variational formulation in which the main unknowns are $\mathbf{E}, \mathbf{H}$ and $\lambda$.

### Dirichlet's condition

We consider the following relation for Dirichlet boundary condition

$$\lambda \times \mathbf{n} = f_D.$$

With this boundary condition, the second equation of (3.44) becomes:

$$\frac{d}{dt}\int_K \mu\mathbf{H}\cdot\psi_i\, dx + \int_K \mathbf{curl}\,\mathbf{E}\cdot\psi_i + \int_{\partial K\setminus\Gamma_D} \mathbf{n}\times(\lambda-\mathbf{E})\cdot\, dx = \int_{\Gamma_D} f_D\cdot\psi_i\, dx$$

where $\Gamma_D$ is the boundary on which we have a Dirichlet condition. To keep the symmetry of the system in $\lambda$, the equation over $\Gamma_D$ is then written as

$$\int_{\Gamma_D} \tau(E-\lambda)\times\mathbf{n}\cdot(q_i\times\mathbf{n})\, dx = 0.$$

### Neumann's condition

To set the Neumann condition on the boundary, we consider

$$\mathbf{n} \times \hat{\mathbf{H}} = f_N.$$

The expression for $\mathbf{E}$ and $\mathbf{H}$ doesn't change. But the equation in $\lambda$ takes the form

$$\int_{\Gamma_N} (\mathbf{H} - \tau \mathbf{n} \times (\lambda - \mathbf{E})) \cdot (\mathbf{n} \times q_i)\, dx = -\int_{\Gamma_N} f_N \cdot q_i\, dx,$$

where $\Gamma_N$ is the boundary on which we have considered the Neumann condition.

### Silver-Müller condition

For the Silver-Müller condition we use the following equation

$$\mathbf{n} \times \hat{\mathbf{H}} + \sqrt{\frac{\varepsilon}{\mu}} \mathbf{n} \times (\mathbf{E} \times \mathbf{n}) = f_A.$$

This leads to the following change in the equation for $\lambda$:

$$\int_{\Gamma_A} (\mathbf{H} - \tau \mathbf{n} \times (2\lambda - \mathbf{E})) \cdot (\mathbf{n} \times q_i)\, dx = -\int_{\Gamma_A} f_A \cdot q_i\, dx.$$

Here $\Gamma_A$ represents the boundary in which we have set the ABC of Silver-Müller.

### Discrete system

After exploiting each boundary condition, we have obtained the following discrete system

$$\begin{cases} D_h \dfrac{d\mathbf{E}}{dt} + (R^T - S_n)\mathbf{H} + C\mathbf{E} - C_\ell \Lambda = 0 \\ B_h \dfrac{d\mathbf{H}}{dt} + (-R + S_n^T)\mathbf{E} - S_d^T \Lambda = F_v(t) \\ C_\ell^T \mathbf{E} - C_a \Lambda - S_d \mathbf{H} = F_\Lambda(t) \end{cases} \tag{3.46}$$

where $F_v(t), F_\Lambda(t)$ are source terms coming from the inhomogeneous boundary conditions. We have the following elementary matrices:

$$(D_h^{\text{elem}})_{i,j} = \int_{K_i} \varepsilon\, \varphi_j \cdot \varphi_i\, dx$$

$$(B_h^{\text{elem}})_{i,j} = \int_{K_i} \mu\, \psi_j \cdot \psi_i\, dx$$

$$R_{i,j}^{\text{elem}} = -\int_{K_i} \mathbf{curl}\, \varphi_j \cdot \psi_i\, dx$$

$$(S_n)_{i,j}^{\text{elem}} = \int_{\partial K_i} \mathbf{n} \times \psi_j \cdot \varphi_i \, dx$$

$$(S_d)_{i,j}^{\text{elem}} = \int_{\partial K_i \setminus \Gamma_D} \mathbf{n} \times q_i \cdot \psi_j \, dx$$

$$C_{i,j}^{\text{elem}} = \tau \int_{\partial K_i} (\varphi_j \times \mathbf{n}) \cdot (\varphi_i \times \mathbf{n}) \, dx$$

$$(C_\ell^{\text{elem}})_{i,j} = \tau \int_{\partial K_i} (q_j \times \mathbf{n}) \cdot (\varphi_i \times \mathbf{n}) \, dx$$

$$(C_a^{\text{elem}})_{i,j} = \tau \int_{\partial K_i} \beta' (q_j \times \mathbf{n}) \cdot (q_i \times \mathbf{n}) \, dx$$

where $\beta'$ equals 2 for faces with an absorbing boundary condition. The final matrices $D_h, B_h, R, S_n, S_d, C, C_a, C_\ell$ are obtained by assembling the elementary matrices. For any kind of element $K_i$ (tetrahedron, hexahedron, etc), the mass matrices $B_h$ and $D_h$ are block-diagonal, each block being related to the degrees of freedom of an element. The matrix $C_a$ is also block-diagonal, each block being related to degrees of freedom of a face of the mesh. For an explicit time-stepping algorithm, the matrices $R, S_n, S_d, C_a, C_\ell$ will not be stored. For hexahedral elements the matrices $D_h$, $B_h$ and $C_a$ will be block-diagonal with blocks of size 3x3 and 2x2. For other elements, $B_h$ and $D_h$ will be block-diagonal of size $N \times N$ where $N$ is the number of degrees of freedom for a component of $\mathbf{E}$. That is the reason why Piola transform has not been included for other elements, in order to have smaller blocks for $D_h$ and $B_h$. In order to obtain the desired ODE, we use the relation:

$$\Lambda = C_a^{-1} \left( C_\ell^T \mathbf{E} - S_d \mathbf{H} - F_\Lambda(t) \right) \tag{3.47}$$

to get

$$M_h \frac{dX(t)}{dt} + K_h X(t) = F(t)$$

with the expression of $K_h$ as introduced in subsubsection 3.2.2. A static condensation can be performed to remove the unknowns $\mathbf{E}$ and $\mathbf{H}$ for each element. The algorithm is identical to the algorithm given in the subsubsection 3.2.2, $U$ and $V$ being replaced by $\mathbf{E}$ and $\mathbf{H}$.

Chapter 4

# Review of high-order Runge Kutta schemes, stability, dispersion and dissipation analysis

This chapter gives a review of classical Runge-Kutta (RK) schemes commonly used in the literature. After recalling some properties and definitions, we present results related to the numerical stability, the dissipation and dispersion errors of these schemes. The chapter is concluded by comparison results of the different schemes when they are combined with finite element methods to solve the acoustic wave equation.

## Contents

## 4.1 Introduction

The spatial discretization of the first order formulation of acoustic wave equation or Maxwell's equations results in a semi-discrete problem which corresponds to a system of ODEs and has to be integrated in time. As we have introduced in the Chapter 2, dissipation and dispersion must be avoided to have accurate simulation of wave propagation problems. Therefore, in this chapter, we are concerned with the numerical analysis of the Runge-Kutta methods for ODEs, with the aim to find a high order, stable, low-dispersive and low dissipative scheme.

As an example we consider the initial value problem

$$\begin{aligned} &\frac{dy(t)}{dt} = f(t, y(t)), \quad \text{for } t \in [t_0, T], \\ &y(t_0) = y_0. \end{aligned} \tag{4.1}$$

Let $t_0 < t_1 < \cdots < t_{N-1} < t_N = T$ be a grid discretization of $[t_0, T]$. Integrating the differential equation in (4.1) between two grid points, $t_n$ and $t_{n+1}$, yields

$$y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+1}} f(t, y(t))dt. \tag{4.2}$$

## 4.2 Runge-Kutta (RK) schemes

### *4.2.1 Principle of Runge-Kutta methods*

We now seek the best approximation to the right-hand side term in (4.2). Let's first denote by $h = t_{n+1} - t_n$ the discretization step, and consider the following variable change $t = t_n + ch$, then

$$\int_{t_n}^{t_{n+1}} f(t, y(t)) = h \int_0^1 f(t_n + ch, y(t_n + ch))\, dc \tag{4.3}$$

It remains to approximate the integral over $[0, 1]$ by a quadrature formula. Using a quadrature formula with $s$ points ($b_i$ and $c_i$ denote the quadrature weights and points respectively), we obtain

$$y(t_{n+1}) \approx y(t_n) + h \sum_{i=1}^{s} b_i f(t_n + c_i h, y(t_n + c_i h)). \tag{4.4}$$

We also have

$$y(t_n + c_i h) = y(t_n) + h \int_0^{c_i} f(t_n + ch, y(t_n + ch))dc \tag{4.5}$$

which is approximated using an interpolation formula based on the same points. We first use the interpolation of $f$ over the quadrature nodes $c_j$. We obtain

$$y(t_n + c_i h) \approx y(t_n) + h \int_0^{c_i} \sum_{j=1}^{s} f(t_n + c_j h, y(t_n + c_j h))\varphi_j(c)$$

where $\varphi_j$ are the Lagrange polynomials based on the quadrature points $c_j$. We then find:

$$y(t_n + c_i h) \approx y(t_n) + h \sum_{j=1}^{s} a_{ij} f(t_n + c_j h, y(t_n + c_j h)) \tag{4.6}$$

where

$$a_{ij} = \int_0^{c_i} \varphi_j(c) dc.$$

This formula works both for Gauss-Lobatto and Gauss-Radau Runge-Kutta schemes. (4.6 & 4.4) together represent the $s$-stage Runge-Kutta methods. Using the Butcher table [14], one can represent (4.6 & 4.4) as follows

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s1} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array} \quad \textbf{ou} \quad \begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^T \end{array}, \tag{4.7}$$

where **c** corresponds to the position of stage value within the time-step. Matrix **A** is the matrix in (4.6) and vector **b** corresponds to the weight coefficients of the quadrature formula in (4.4). This approach leads to fully-implicit Runge-Kutta schemes. To find diagonally implicit schemes or explicit schemes, one can consider different interpolation formula by taking fewer quadrature points at each step for example.

Another approach which is much more general consists in finding the coefficients $b_i$, $c_i$ and $a_{ij}$ of a Runge-Kutta scheme satisfying the order conditions (conditions for a given order of accuracy) directly. This can be done by using the Taylor expansion. First we expand the exact solution of the ODE in a Taylor's series. We substitute the exact solution of the ODE into the Runge-Kutta formula and expand the result in a Taylor's series. Then we match the two Taylor's series expansion at a desired order to find $b_i$, $c_i$ and $a_{ij}$ see [14]. The coefficients of the Runge-Kutta scheme for a given order are usually not uniquely determined by this process; we generally find a family of methods. A Runge-Kutta scheme of order $k$ will match the term of order $h^k$ in both series. The algebra involved in obtaining these formulas becomes more and more complex with increasing order. There are many works in the literature on the construction of Runge-Kutta schemes. Among these works, we refer the reader to the book

[14] in which the author details the algebra needed to construct Runge-Kutta schemes at any order of accuracy and provides many examples.

If we go back to the Butcher table 4.7 for Runge-Kutta methods, we can distinguish Explicit Runge-Kutta (ERK) methods from implicit ones as follows: when $a_{ij} = 0$ for $i \leq j$, $i = 1, \ldots, s$ and $j = 1, \ldots, s$

$$\begin{array}{c|cccc} c_1 & & & & \\ c_2 & a_{21} & & & \\ \vdots & \vdots & \ddots & & \\ c_s & a_{s1} & \ldots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \ldots & b_s \end{array} \tag{4.8}$$

corresponds to ERK methods while when there is $j \geq i$ such that $a_{ij} \neq 0$ we have Implicit Runge-Kutta (IRK) methods.

ERK schemes require a stability condition that implies a small time step, which in some situations may lead to prohibitive computational costs. On the contrary, IRK schemes have good stability properties but their implicitness implies to solve linear or non-linear systems at each time-step.

### *4.2.2 Stability function of RK schemes*

In this section, we investigate the stability properties of the RK schemes. We will recall the stability function of any $s$-stages RK scheme when it is applied to the following test equation:

$$y'(t) = \lambda y(t), \quad y_0 = 1. \tag{4.9}$$

The exact solution of this ODE is given by

$$y(t) = \mathrm{e}^{\lambda t}. \tag{4.10}$$

Let $K_i = f(t_n + c_i h, y(t_n + c_i h))$. The $s$-stage RK applied to (4.9) gives rise to

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i K_i, \quad K_i = \lambda \left( y_n + h \sum_{j=1}^{s} a_{ij} K_j \right) \tag{4.11}$$

Let $\mathbf{K} = [K_1, ..., K_s]^T$, $\mathbf{b^T} = [b_1, \ldots, b_s]^T$, $\mathbf{A} = (a_{ij})_{i,j=1,\ldots,s}$ and $\mathbf{1} = [1, \ldots, 1]^T \in \mathbb{R}^s$, rewrite (4.11) as follows

$$y_{n+1} = y_n + h\mathbf{b^T}\mathbf{K}, \quad \mathbf{K} = \lambda \left( y_n \mathbf{1} + h\mathbf{A}\mathbf{K} \right). \tag{4.12}$$

Solving in $\mathbf{K}$ the second equation of (4.12) yields

$$\mathbf{K} = \lambda \left( I - h\lambda \mathbf{A} \right)^{-1} \mathbf{1} y_n.$$

By introducing the value of $\mathbf{K}$ into the first equation of (4.12), we end up with

$$\begin{aligned} y_{n+1} &= (\mathbf{1} + h\lambda \mathbf{b^T}(I - h\lambda \mathbf{A})^{-1}\mathbf{1})y_n \\ &= R(h\lambda)y_n, \end{aligned} \tag{4.13}$$

where

$$R(z) = \mathbf{1} + z\mathbf{b^T}(I - z\mathbf{A})^{-1}\mathbf{1}. \tag{4.14}$$

**Remark** The function $R(z)$ is the stability function of the RK method according to the Definition 2.3.1. It can be interpreted as the numerical solution after one step for (4.9) and $z = h\lambda$.

We also recall that implicit RK methods have a rational stability function $R(z)$, with numerator and denominator of degree $\leq s$ (see [42]):

$$R(z) = \frac{P(z)}{Q(z)}, \quad \text{deg}P \leq s, \quad \text{deg}Q \leq s$$

And if the method is of order $p$, then

$$\mathrm{e}^z - R(z) = cz^{p+1} + O(z^{p+2}) \quad \text{for } z \to 0 \tag{4.15}$$

where $c$ is the error constant. We have the following result for the stability function of Runge-Kutta schemes

**Proposition 4.2.1** *Let $(A, b, c)$ be the coefficients defining a Runge-Kutta scheme as in (4.7). Then its stability function is given by*

$$R(z) = \frac{det(I - zA + z\mathbf{1}b^T)}{det(I - zA)}. \tag{4.16}$$

We refer the reader to [14] and [42] for the proof of the Proposition 4.2.1.

For explicit Runge-Kutta schemes, the numerical solution at $t_{n+1}$ depends only on the solution of the previous step. The scheme (4.11) becomes

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i K_i, \quad K_i = \lambda\left(y_n + h\sum_{j=1}^{i-1} a_{ij}K_j\right) \tag{4.17}$$

Inserting $K_i$ repeatedly in the first equation of (4.17) leads to

$$y_{n+1} = R(h\lambda)y_n$$

where

$$R(z) = 1 + z\sum_{j=1}^{s} b_j + z^2 \sum_{j>k} b_j a_{jk} + \ldots \tag{4.18}$$

**Theorem 4.2.2** *For any ERK method of order $p$, the stability function is given by*

$$R(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^p}{p!} + O(z^{p+1}). \tag{4.19}$$

We refer the reader to [42] for the proof of this theorem.

In the next Sections we describe the stability region of some classical explicit and implicit RK methods found in the literature. We also represent the amplitude (dissipation) and phase (dispersion) errors for each scheme according to the Definition 2.4.1.

## 4.3 Explicit Runge-Kutta (ERK) methods

We present here the stability region, the curves of the dissipation and dispersion errors of different ERK schemes. We consider schemes from order 2 to order 8.

### *4.3.1 Stability region of classical ERK schemes*

Using the midpoint quadrature formula that is $\int_0^1 g(c)dc \approx g(\frac{1}{2})$ for a continuous function $g \in [0,1]$, we can construct an ERK scheme of order 2 as follows

$$\begin{aligned} y(t_n + h) - y(t_n) &= h \int_0^1 f(t_n + ch, y(t_n + ch))dc \\ &\approx h(\underbrace{f(t_n + \frac{1}{2}h, y(t_n + \frac{1}{2}h))}_{K_2}) \end{aligned}$$

An ERK scheme of order 2 (called ERK22) is then given by

$$\begin{aligned} &y_{n+1} = y_n + hK_2 \quad \textit{with} \\ &K_1 = f(t_n, y_n) \quad \textit{and} \\ &K_2 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_1). \end{aligned}$$

and has the following Butcher table

$$\text{ERK22:} \quad \begin{array}{c|cc} 0 & & \\ 0.5 & 0.5 & \\ \hline & 0 & 1 \end{array} \tag{4.20}$$

In the book [14], the author details the construction of explicit Runge-Kutta schemes up to order 8 and provides many examples. Herein, we recall some of these schemes implemented in the code Montjoie [31]. First we present one

third order ERK scheme which is represented in the following Butcher's table by

$$\textbf{ERK33:}\quad \begin{array}{c|ccc} 0 & & & \\ 0.5 & 0.5 & & \\ 1 & -1 & 2 & \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array} \tag{4.21}$$

One of the well known fourth order ERK scheme is given in the following table

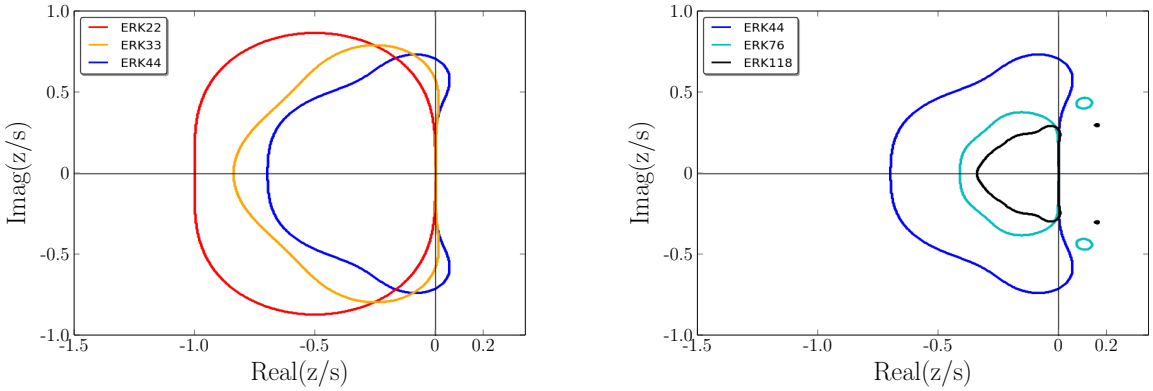
$$\textbf{ERK44:}\quad \begin{array}{c|cccc} 0 & & & & \\ 0.5 & 0.5 & & & \\ 0.5 & 0 & 0.5 & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}\ . \tag{4.22}$$

And one sixth order scheme with seven stages is given by

$$\textbf{ERK76:}\quad \begin{array}{c|ccccccc} 0 & & & & & & & \\ \frac{1}{3} & \frac{1}{3} & & & & & & \\ \frac{2}{3} & 0 & \frac{2}{3} & & & & & \\ \frac{1}{3} & \frac{1}{12} & \frac{1}{3} & -\frac{1}{12} & & & & \\ \frac{5}{6} & \frac{25}{48} & -\frac{55}{24} & -\frac{35}{48} & \frac{15}{8} & & & \\ \frac{1}{6} & \frac{3}{20} & -\frac{11}{24} & -\frac{1}{8} & \frac{1}{2} & \frac{1}{10} & & \\ 1 & -\frac{261}{260} & \frac{33}{13} & \frac{43}{156} & -\frac{118}{39} & \frac{32}{195} & \frac{80}{39} & \\ \hline & \frac{13}{200} & 0 & \frac{11}{40} & \frac{11}{40} & \frac{4}{25} & \frac{4}{25} & \frac{13}{200} \end{array}\ . \tag{4.23}$$

The scheme of order 8 with eleven stages (ERK118 - Cooper-Verner) can be seen in [14] at the page 197.

The stability region contour of each ERK scheme is presented in the Figure 4.1. We have drawn them by choosing $\frac{z}{s}$ where $z = x + iy$, $x, y \in \mathbb{R}$. By this way we take into account the complexity of each scheme since $s$ is the number of stages. From the Figure 4.1(a), we claim that ERK44 is the more efficient because its contour includes a larger part of the imaginary axis, which is the good property for solving hyperbolic problems. For the same reasons, Figure 4.1(b) shows that ERK44 is more efficient than ERK76 and ERK118. In any case the efficiency of these schemes is closely related to the time evolution problem considered.




(a) ERK schemes of order 2, 3 and 4

(b) ERK schemes of order 4, 6 and 8

Figure 4.1: Stability region of ERK schemes.

Briefly, all these schemes have a bounded stability region. As a result, in order to have a stable solution we must choose the time step $h$ such that $z = h\lambda$ gets into the stability region. With this process the time step is always restricted by the CFL (Courant-Friedrichs-Lewy) condition, which depends on the time scheme used.

### 4.3.2 *Dissipation and dispersion of classical ERK schemes up to order 8*

Now we want to show that choosing the time step regarding the stability limit only doesn't guaranty the accuracy of the numerical solution. In the Figures 4.2 and 4.3 we present respectively the relative dissipation and dispersion errors of the ERK schemes of order 2, 3 and 4. As previously, we have chosen $\frac{z}{s}$ for the abscissa, $s$ being the number of stages. We observe that both the dissipation and the dispersion errors are lower for the scheme ERK44 than for ERK22 and ERK33.

Also in order to minimize the dissipation and the dispersion errors, the time step must be restricted to an accuracy limit as shown in the the Figures 4.2 and 4.3. We obsere on these figures that the dissipation and dispersion errors will be important for large time-step. Like for the stability issue, the accuracy issues can be overcome by taking a small step-size as well. By this way we can get a stable solution which is not polluted by the dissipation and dispersion errors. But it might significantly increase the computational cost.

In Figures 4.4 and 4.5 we also present comparative curves of the relative dissipation and relative dispersion errors of the ERK44, ERK76 and ERK118. Here again the ERK44 seems more efficient.

By drawing the stability region, the dissipation and dispersion curves of some ERK methods, we have shown their major drawbacks in solving ODEs.

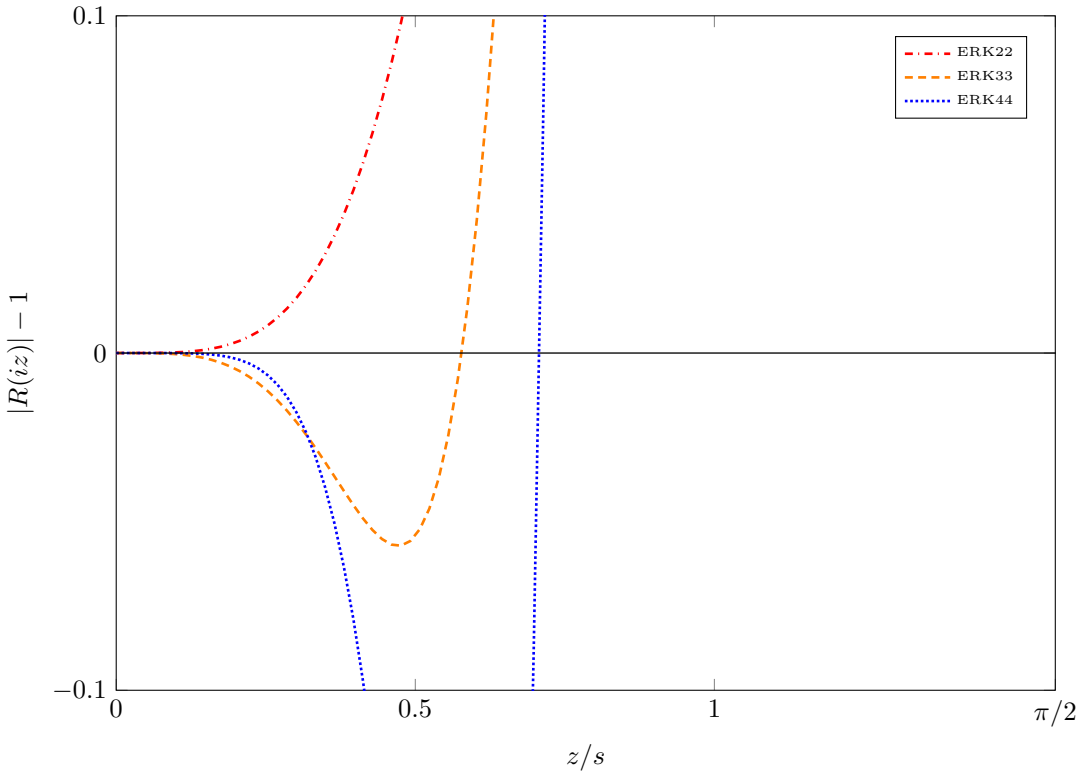



Figure 4.2: Dissipation errors of ERK schemes of order 2, 3 and 4

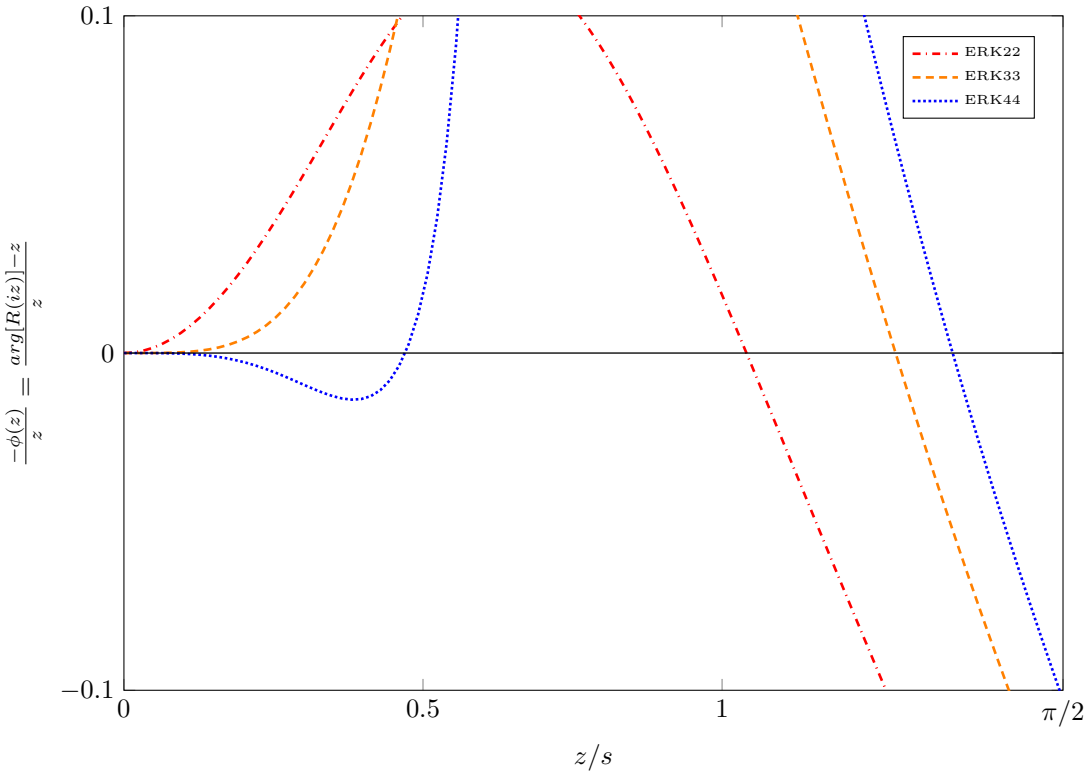



Figure 4.3: Dispersion errors of ERK schemes of order 2, 3 and 4

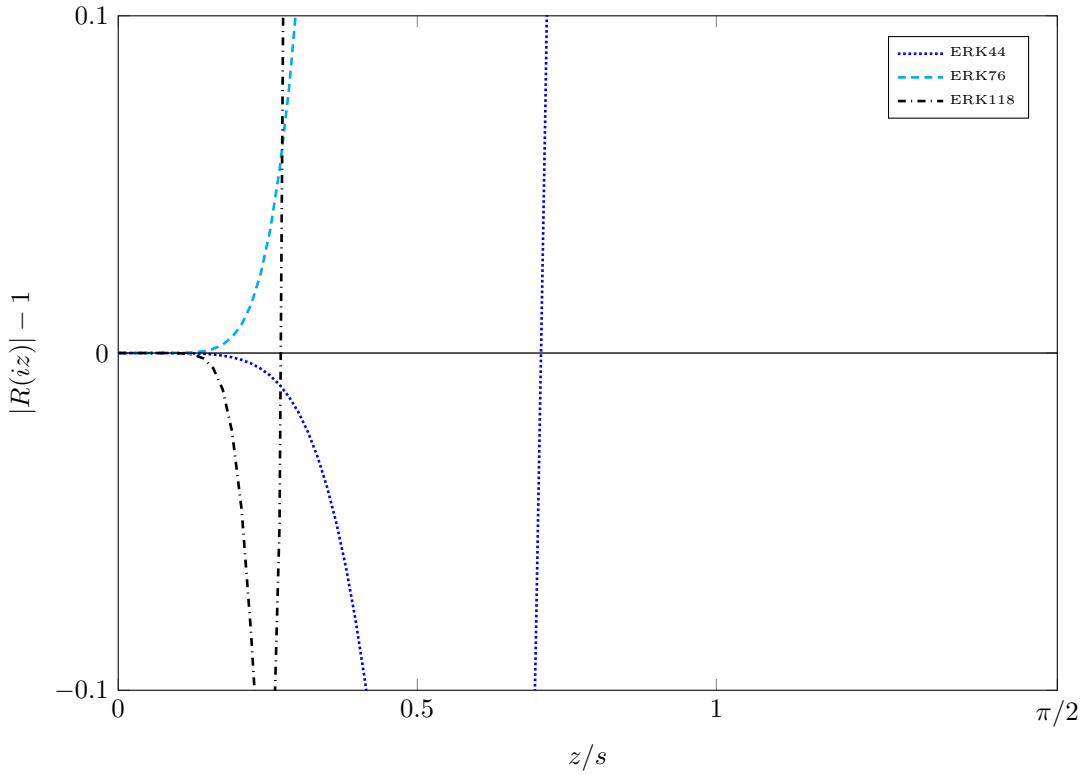



Figure 4.4: Relative dissipation errors of ERK schemes of order 4, 6 and 8

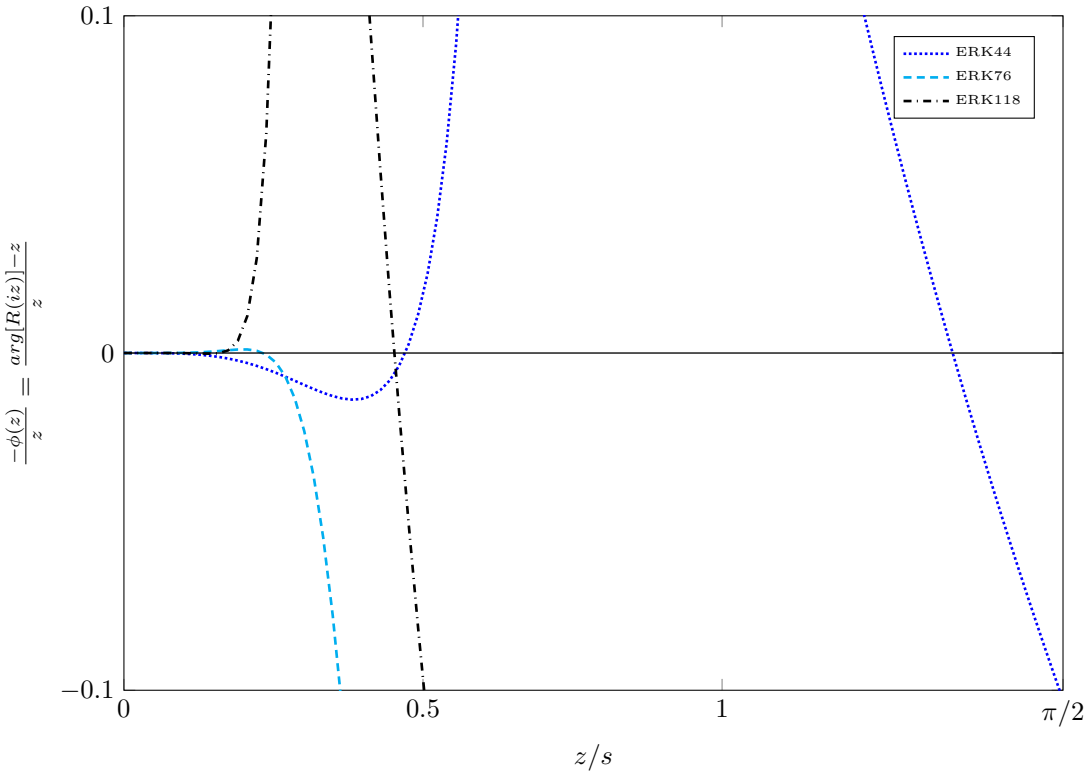



Figure 4.5: Relative dispersion errors of ERK schemes of order 4, 6 and 8

The first one is the existence of a stability condition. The second one is the dissipation and dispersion issues. As a result, the time step must be chosen carefully to obtain a stable scheme and an accurate solution.

Implicit schemes are known to have better stability over the explicit ones. Are they less dispersive and less dissipative than explicit methods ? Is a large stability region the whole answer to the problem? The next section is devoted to answer those questions.

## 4.4 Fully implicit Runge-Kutta (IRK) methods

> To be $A$-Stable, and possibly useful for stiff systems, a Runge-Kutta formula must be implicit.
>
> *R. Alexander 1977* [1]

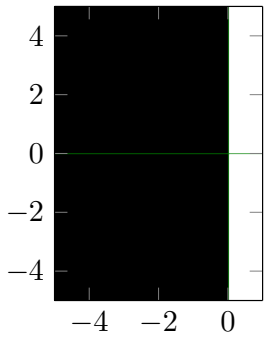
In the previous section, we have seen that ERK methods for solving ODEs can be numerically unstable, unless the step size is chosen very small. This is the case in general for a class of equations called stiff equations. Historically stiff equations or stiff problems are defined to be a kind of problem where certain implicit methods perform better results than the explicit ones [42]. In other words implicit methods are required for stiff equations. But for wave propagation problems, stability is not the whole answer. As formerly discussed in the Chapter 2 we must be aware of the dissipation and dispersion errors when solving the wave propagation problems [74]. Here we recall fully implicit RK methods with a minimal number of stages and a maximal order. We present their stability region, their dissipation and dispersion errors. We recall also the implementation of these schemes to highlight their main drawback.

### *4.4.1 The implicit Gauss Runge-Kutta schemes*

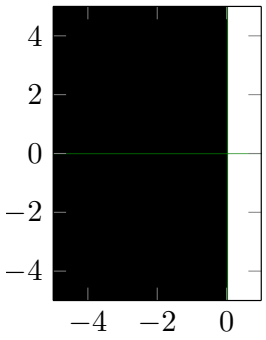
The family of collocation methods based on Gauss-Legendre quadrature formula forms the Gauss Runge-Kutta (Gauss-RK) schemes. These schemes with $s$-stage have the maximal order of accuracy $2s$. In this section we recall the second and fourth order Gauss-RK schemes. We present some interesting properties of these schemes regarding their stability and their dissipation.

The second order Gauss-RK scheme is presented in the following Butcher table

$$\text{GaussRK2:}\quad \begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}. \tag{4.24}$$




(a) GaussRK2




(b) GaussRK4

Figure 4.6: Stability region of the Gauss Runge-Kutta schemes of order 2 and 4.

And the stability function of the second order Gauss-RK scheme is given by

$$R(z) = \frac{1 + \frac{z}{2}}{1 - \frac{z}{2}}. \tag{4.25}$$

Compared to the ERK schemes, the stability region of this scheme, drawn in the Figure 4.6, covers the entire negative half plane ($\mathbb{C}^-$). According to the A-stability definition, the second order Gauss-RK is thus A-stable.

The fourth order Gauss-RK scheme with two stages [14] is given by Butcher table as follows

$$\text{GaussRK4:} \quad \begin{array}{c|cc} \frac{1}{2} - \frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}. \tag{4.26}$$

The stability function of the fourth order Gauss-RK scheme is given by

$$R(z) = \frac{1 + \frac{z}{2} + \frac{z^2}{12}}{1 - \frac{z}{2} + \frac{z^2}{12}}. \tag{4.27}$$

Like the second order Gauss-RK, the fourth order Gauss-RK scheme is also unconditionally stable. Both have an unbounded stability region which covers the negative half plane (see 4.6). In Figure 4.6, we recall that the region of absolute stability is the set of all complex numbers $z$ such that $|R(z)| \leq 1$.

These results come true for any Gauss-Runge-Kutta scheme at any order of accuracy. To construct the $s$-stages Gauss-RK schemes it suffices to choose $c_i$

and $b_i$, $i = 1, \ldots, s$ respectively as the Gauss-Legendre quadrature points and their corresponding weights. Then we compute the coefficients $a_{ij}$, $i, j = 1, \ldots, s$ using (4.4) and (4.6).

We have seen in the Subsection 4.2.2 that for explicit methods $R(z)$ is polynomial, while it is a rational function for implicit methods. From the equation (4.15), we know that a method of order $p$ will satisfy

$$\mathrm{e}^z - R(z) = cz^{p+1} + O(z^{p+2}) \quad \text{for } z \to 0.$$

Knowing this, we briefly introduce the rational approximation of the exponential function commonly known as Padé approximations.

**Definition 4.4.1** *The Padé approximation $R_{r,s}(z)$ (or $[r/s]$) of the exponential function, is the maximum-order approximation of $\mathrm{e}^z$ [14] given by*

$$R_{r,s}(z) = \frac{N_{r,s}(z)}{D_{r,s}(z)} = \frac{\sum_{i=0}^{s} \frac{s!\,(r+s-i)!}{(r+s)!\,i!\,(s-i)!}(z)^i}{\sum_{i=0}^{r} \frac{r!\,(r+s-i)!}{(r+s)!\,i!\,(r-i)!}(-z)^i}. \tag{4.28}$$

*$N_{r,s}(z)$ and $D_{r,s}(z)$ have no common factors and*

$$\mathrm{e}^z - R_{r,s}(z) = cz^{r+s+1} + O(z^{r+s+2}) \quad \text{for } z \to 0. \tag{4.29}$$

**Definition 4.4.2** *A Padé approximation of an exponential function is called*

- *diagonal Padé approximation if $r = s = m$, $m \in \mathbb{N}$. We note them $R_m(z)$ or $[m/m]$.*
- *sub-diagonal Padé approximation if $s = m$, $r = m + 1$, $m \in \mathbb{N}$*
- *two sub-diagonal Padé approximation if $s = m$, $r = m + 2$, $m \in \mathbb{N}$.*

**Remark** The stability functions of the second and fourth order Gauss-RK are respectively the same as $R_{1,1}(z)$ (noted also $[1/1](z)$) and $R_{2,2}(z)$ (noted also $[2/2]$).

**Theorem 4.4.3** *The $s$-stages Gauss-RK scheme has the same stability function as the $[s/s]$ diagonal Padé approximation of the exponential function.*

We refer the reader to [14] for the proof of this theorem.

In the paper [34], Ehle showed that the diagonal, sub-diagonal and two sub-diagonal Padé approximations of the exponential function are $A$-stable. That confirms the results shown in the Figure 4.6. In the Chapter 5 we present schemes derived from the diagonal Padé approximations and we will prove that they are equivalent to the Gauss-RK schemes for linear ODEs.

The dissipation curves shown in Figure 4.7, obtained by applying Gauss-RK schemes to the test equation $y' = i\lambda y$, demonstrate that Gauss-RK schemes are

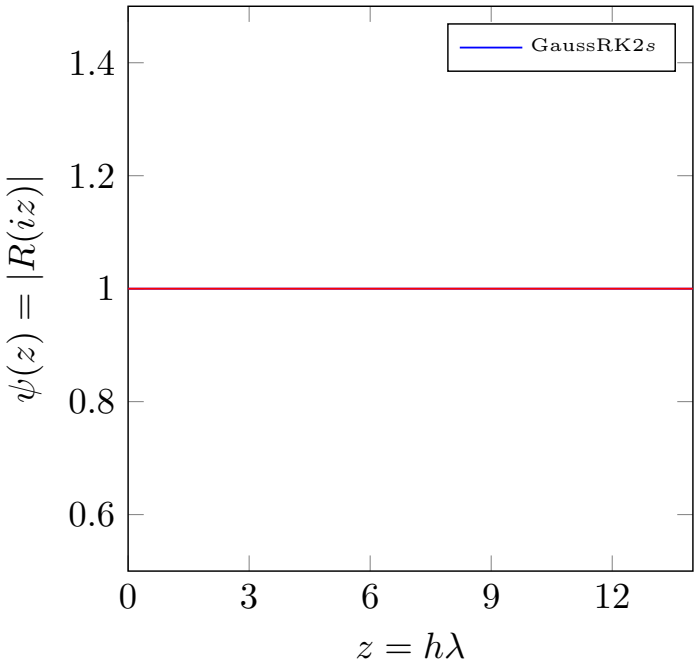



Figure 4.7: Dissipation errors of the Gauss Runge-Kutta schemes of order $2s$.

not dissipative. This can be demonstrated by just observing that the stability function of Gauss-Runge-Kutta methods satisfies

$$|R(iz)| = 1, \, \forall z \in \mathbb{R}.$$

However, in Figure 4.8 we observe that Gauss-RK methods are dispersive unless the step size is restricted to an accuracy limit. Like for explicit schemes we notice that increasing order of the scheme is helpful to minimize the dispersion error and allows to take larger time step. The good point here is that the stability and the dissipation are not an issue. The time step is only constrained by the dispersion. While this is not an issue when solving some equations like the heat equation, the dispersion represents a serious issue for wave propagation problems.

### *4.4.2 Complexity of the $s$-stages Gauss-RK schemes*

As we have seen in the previous sub-section, Gauss-RK schemes are $A$-stable. On top of that, they are not dissipative. The main constraint for the choice of the time step is the dispersion error. To overcome this, the only cure is to increase the order of the scheme. In this subsection, we want to show that using high order Gauss-RK schemes is not always feasible due to computational resources needed.

Explicit ERK schemes are easily solved because they involve only a matrix vector product and evaluation of source terms. However implicit methods re-

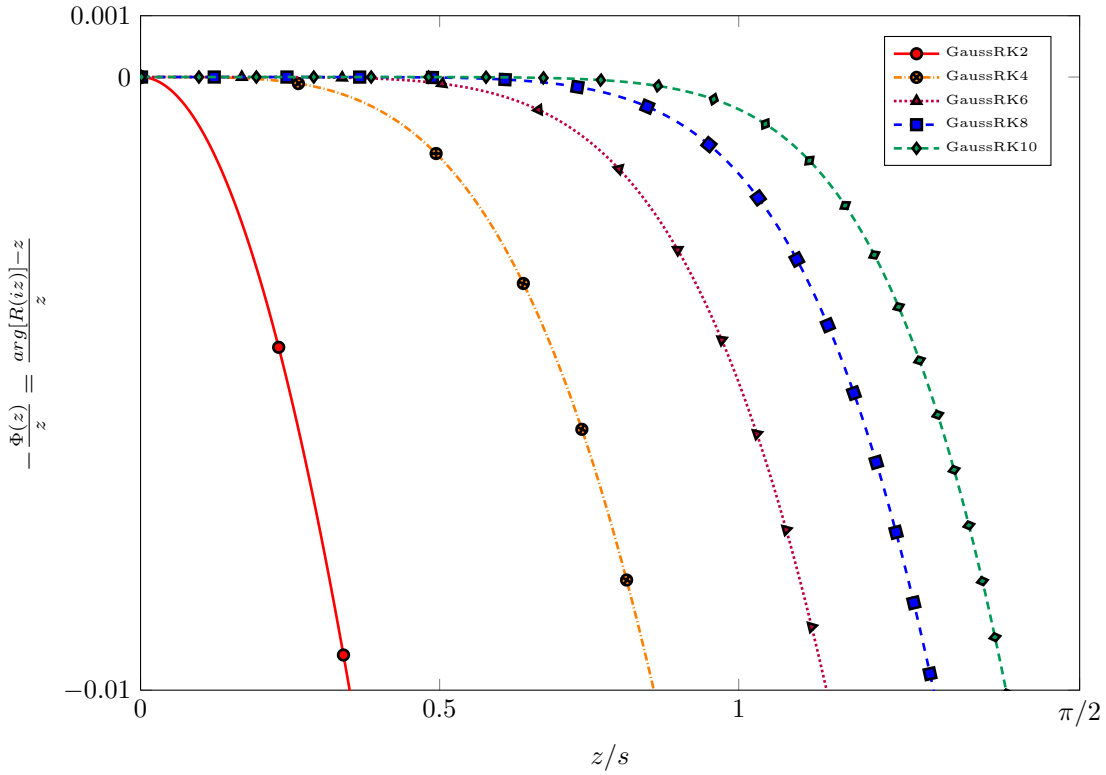



Figure 4.8: Dispersion errors of the Gauss Runge-Kutta schemes of order 2, 4, 6, 8 and 10.

quire a direct solver or an iterative solution. To emphasize the difficulty of computing solution with implicit schemes, we use Newton's method to solve the non-linear system. To illustrate this, let us consider the following initial value problem (IVP)

$$y'(t) = f(t, y(t)),\ y(0) = y_0 \in \mathbb{R}^N, \tag{4.30}$$

where $y$ is a vector of $\mathbb{R}^N$ and $t \in [0, T]$, $T \in \mathbb{R}$. We consider the $s$-stages Gauss-RK schemes of order $2s$. The solution of (4.30) at one step using an $s$-stages RK (4.7), like Gauss-RK, is computed as follows

$$K_i = f\left(t_n + c_i\Delta t, y_n + \Delta t \sum_{j=1}^{s} a_{i,j}K_j\right),\ i = 1, \dots, s, \tag{4.31}$$

$$y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i K_i. \tag{4.32}$$

Whenever the solutions $K_1, \dots, K_N$ of (4.31) are known, then (4.32) is an explicit formula for $y_{n+1}$ involving a simple combination of $K_i$. We denote $\delta K_i^{(\mu)} = K_i^{(\mu+1)} - K_i^{(\mu)}$, $K_i^{(\mu)} \in \mathbb{R}^N$ and $\delta K = (\delta K_1^{(\mu)}, \dots, \delta K_s^{(\mu)})$, $K \in \mathbb{R}^{sN}$. We define the vectorial function

$$F_i^{(\mu)}(K_1^{(\mu)}, \dots, K_s^{(\mu)}) = K_i^{(\mu)} - f\left(t_n + c_i\Delta t, y_n + \Delta t \sum_{j=1}^{s} a_{i,j}K_j^{(\mu)}\right),\ i = 1, \dots, s. \tag{4.33}$$

The Newton's method applied to (4.31) involves at each iteration the solution of the following linear system:

$$\underbrace{\begin{pmatrix} I - a_{1,1}\Delta t\mathbf{J}_1^{(\mu)} & -a_{1,2}\Delta t\mathbf{J}_1^{(\mu)} & \dots & -a_{1,s}\Delta t\mathbf{J}_1^{(\mu)} \\ -a_{2,1}\Delta t\mathbf{J}_2^{(\mu)} & (I - a_{2,2}\Delta t\mathbf{J}_2^{(\mu)}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & -a_{s-1,s}\Delta t\mathbf{J}_{s-1}^{(\mu)} \\ -a_{s,1}\Delta t\mathbf{J}_s^{(\mu)} & \dots & -a_{s,s-1}\Delta t\mathbf{J}_s^{(\mu)} & (I - a_{s,s}\Delta t\mathbf{J}_s^{(\mu)}) \end{pmatrix}}_{=:B} \delta\mathbf{K}^{(\mu)} = \mathbf{F}^{(\mu)} \tag{4.34}$$

where

$$\delta\mathbf{K}^{(\mu)} = \begin{pmatrix} \delta K_1^{(\mu)} \\ \delta K_2^{(\mu)} \\ \vdots \\ \delta K_3^{(\mu)} \end{pmatrix} \quad , \quad \mathbf{F}^{(\mu)} = \begin{pmatrix} -F_1^{(\mu)} \\ -F_2^{(\mu)} \\ \vdots \\ -F_3^{(\mu)} \end{pmatrix}$$

and

$$\mathbf{J}_i^{(\mu)} = \partial_y f(t_n + c_i\Delta t, y_n + \Delta t \sum_{j=1}^{s} a_{i,j}K_j^{(\mu)}),\ i = 1, \dots, s.$$

For a $s$-stages Gauss-RK applied to the $N$-dimensional system (4.30), the Jacobian matrix $B$ in the (4.34) is a $sN \times sN$ matrix. This will be increasingly expensive for high-order methods and high-dimensional ODEs. This is why these schemes will become somehow quite inefficient.

Some simplifications can be done by approximating all Jacobians $\mathbf{J}_i^{(\mu)}$ as follows:

$$\mathbf{J}_i^{(\mu)} \approx \mathbf{J} = \partial_y f(t_n, y_n),\ i = 1, \dots, s.$$

This approach leads to the following simplified Jacobian matrix

$$B := \begin{pmatrix} I - a_{1,1}\Delta t\mathbf{J} & -a_{1,2}\Delta t\mathbf{J} & \dots & -a_{1,s}\Delta t\mathbf{J} \\ -a_{2,1}\Delta t\mathbf{J} & (I - a_{2,2}\Delta t\mathbf{J}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & -a_{s-1,s}\Delta t\mathbf{J} \\ -a_{s,1}\Delta t\mathbf{J} & \dots & -a_{s,s-1}\Delta t\mathbf{J}_{s-1} & (I - a_{s,s}\Delta t\mathbf{J}) \end{pmatrix} = I - \Delta t A \otimes \mathbf{J} \tag{4.35}$$

where $\otimes$ represents the Kronecker's product. We can notice that each iteration of the Newton method for (4.31) requires $s$ evaluations of $f$ and the solution of a $sN \times sN$ system. With the simplified Jacobian matrix, $(I - \Delta t A \otimes \mathbf{J})$ remains the same for all iterations. Even though we have one LU-decomposition (see [52]) to perform once for all, the Gauss-RK method is still expensive due to the large size of the system. We can also reduce the linear system into a polynomial equation of J. For non-linear problems, we will have to solve a polynomial equation with multiple right-hand-side (RHS) member for each $K_i$, while for a linear case, we will have a polynomial equation representing $y_{n+1} - y_n$ with only one RHS. So instead of solving the large system of size $sN \times sN$, we can solve $s$ smaller systems of size $N$ by factorizing the polynomial (see the Section 5.3 in the Chapter 5).

In the next section, we present another family of A-stable implicit Runge-Kutta schemes. They require more stages for a given order of accuracy, but they allow the solution of a smaller system compared to the family of Gauss-RK schemes we have presented before.

## 4.5 Diagonally Implicit Runge-Kutta (DIRK) methods

In the previous section we have shown some fully implicit RK schemes which are A-stable. Although they have good stability properties and are not dissipative regarding our test equation, the computational cost to pay for can make them inefficient.

The Diagonally Implicit Runge-Kutta (DIRK) methods are developed to overcome these issues. Instead of solving a large system of size $s \times N$, DIRK schemes

allow to compute numerical solution at each time step by solving $s$ times smaller system of size $N$ ($s$ being the number of stages and $N$ the number of unknowns). DIRK schemes are Runge-Kutta schemes where the upper-diagonal terms of the matrix $\mathbf{A}$ in (4.7) are set to zero

$$a_{ij} = 0, \text{ if } i < j, \quad \forall i, j = 1, \dots, s. \tag{4.36}$$

In the form of Butcher table they are given by

$$\begin{array}{c|cccc} c_1 & a_{11} & & & \\ c_2 & a_{21} & a_{22} & & \\ \vdots & \vdots & \vdots & \ddots & \\ c_s & a_{s1} & a_{s1} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array} \tag{4.37}$$

Such schemes, with a lower-triangular $\mathbf{A}$, allow decoupling the solution at a given stage from the solution at future stages. In fact, the system in the Newton methods is given by

$$\begin{pmatrix} (I - a_{1,1}\Delta t \mathbf{J}_1^{(\mu)}) & 0 & \dots & 0 \\ -a_{2,1}\Delta t \mathbf{J}_2^{(\mu)} & (I - a_{2,2}\Delta t \mathbf{J}_2^{(\mu)}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ -a_{s,1}\Delta t \mathbf{J}_s^{(\mu)} & \dots & -a_{s,s-1}\Delta t \mathbf{J}_s^{(\mu)} & (I - a_{s,s}\Delta t \mathbf{J}_s^{(\mu)}) \end{pmatrix} \delta \mathbf{K}^{(\mu)} = \mathbf{F}^{(\mu)} \tag{4.38}$$

The system (4.38) is lower block triangular and can be solved by forward substitution. The solution of the first block involves the linear system of size $N$:

$$(I - a_{1,1}\Delta t \mathbf{J}_1^{(\mu)}) \delta K_1^{(\mu)} = -F_1^{(\mu)}. \tag{4.39}$$

Knowing $K_1$ the second system is given by

$$(I - a_{2,2}\Delta t \mathbf{J}_2^{(\mu)}) \delta K_2^{(\mu)} = -F_2^{(\mu)} + a_{2,1}\Delta t \mathbf{J}_2^{(\mu)}, \tag{4.40}$$

and so on. This facilitates the implementation, and enhances the convergence of the iterative method.

> There is a significant computational advantage in diagonally implicit formulae, whose coefficient matrix is lower triangular with all diagonal element equal.
>
> *R. Alexander 1977* [1]

Different DIRK methods have been developed in the literature. Here we consider a particular case of DIRK method called Singly-Diagonally Implicit

Runge-Kutta (SDIRK) method. SDIRK schemes are a particular case of DIRK schemes, where $a_{i,i} = \gamma$, $\gamma \in \mathbb{R}$ in (4.37)

$$\begin{array}{c|cccc} c_1 & \gamma & & & \\ c_2 & a_{21} & \gamma & & \\ \vdots & \vdots & \vdots & \ddots & \\ c_s & a_{s1} & a_{s1} & \dots & \gamma \\ \hline & b_1 & b_2 & \dots & b_s \end{array} . \tag{4.41}$$

If the Jacobian $\mathbf{J}_i^{(\mu)}$, $i = 1, \dots, s$ is the same for all stages, the diagonal blocks need to be factored only once. Then the resulting system of size $N$ has to be solved $s$ times at each time step and only the right hand side term changes at each time iteration.

**Remark** Regarding the equation (4.16) of the Proposition 4.2.1, the stability function of a $s$-stages SDRIK scheme defined as in (4.41) is given by

$$R(z) = \frac{\det(I - zA + z\mathbf{1}b^T)}{(1 - \gamma z)^s}. \tag{4.42}$$

We can notice that this stability function has only one pole $\gamma$ compared to the DIRK and the Gauss-RK schemes which have multiple poles.

In this section, we present A-stable SDIRK schemes that have been found in the literature. It is worth mentioning that we did not find in the literature A-stable SDIRK schemes for orders greater or equal to 6.

### 4.5.1 Third, fourth and fifth order A-stable SDIRK methods

The only 2-stages A-stable SDIRK scheme of order 3 has been given by Crouzeix [42]. For $(s, p) = (2, 3)$ there is exactly one $A-$stable DIRK formula (see [1]) and it is given by

$$\text{SDIRK23:} \quad \begin{array}{c|cc} \frac{1}{2} + \frac{1}{2\sqrt{3}} & \frac{1}{2} + \frac{1}{2\sqrt{3}} & \\ \frac{1}{2} - \frac{1}{2\sqrt{3}} & \frac{-1}{\sqrt{3}} & \frac{1}{2} + \frac{1}{2\sqrt{3}} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} . \tag{4.43}$$

The only 3-stages A-Stable SDIRK scheme of order 4 has also been derived by Crouzeix. For $(s, p) = (3, 4)$ there is exactly one $A-$stable DIRK formula (see [1])

and it is given by (Crouzeix & Raviart [42])

SDIRK34:

$$\begin{array}{c|ccc} \gamma & \gamma & & \\ \frac{1}{2} & \frac{1}{2}-\gamma & \gamma & \\ 1-\gamma & 2\gamma & 1-4\gamma & \gamma \\ \hline & \delta & 1-2\delta & \delta \end{array}, \quad \gamma = \frac{1}{\sqrt{3}}\cos(\frac{\pi}{18}) + \frac{1}{2}, \quad \delta = \frac{1}{6(2\gamma-1)^2}. \tag{4.44}$$

The A-stable fourth order SDIRK method with five stages, proved to be the most accurate among the SDIRK methods in [71], is given by

S54b:

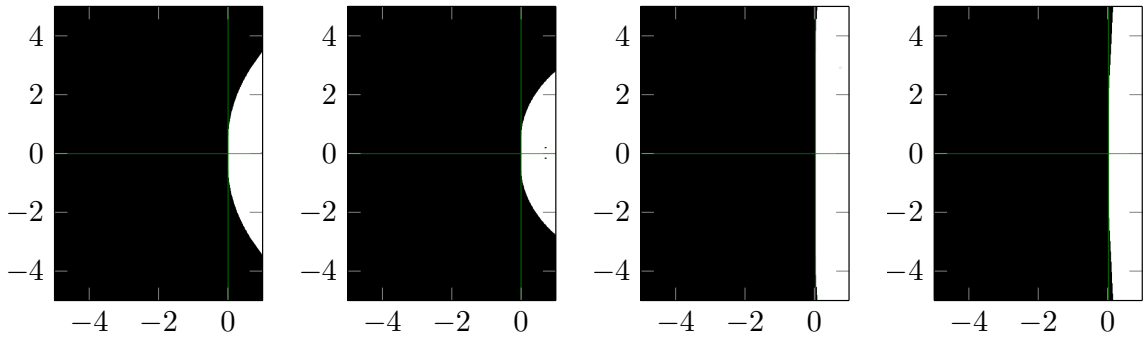
$$\begin{array}{c|ccccc} 1/4 & 1/4 & & & & \\ 0 & -1/4 & 1/4 & & & \\ 1/2 & 1/8 & 1/8 & 1/4 & & \\ 1 & -3/2 & 3/4 & 3/2 & 1/4 & \\ 1 & 0 & 1/6 & 2/3 & -1/12 & 1/4 \\ \hline & 0 & 1/6 & 2/3 & -1/12 & 1/4 \end{array}. \tag{4.45}$$

A $A-$stable fifth-order SDIRK method with 5 stages derived by Cooper & Sayfy [42] is given by

SDIRK55:

$$\begin{array}{c|ccccc} \frac{6-\sqrt{6}}{10} & \frac{6-\sqrt{6}}{10} & & & & \\ \frac{6+9\sqrt{6}}{35} & \frac{-6+5\sqrt{6}}{14} & \frac{6-\sqrt{6}}{10} & & & \\ 1 & \frac{888+607\sqrt{6}}{2850} & \frac{126-161\sqrt{6}}{1425} & \frac{6-\sqrt{6}}{10} & & \\ \frac{4-\sqrt{6}}{10} & \frac{3153-3082\sqrt{6}}{14250} & \frac{3213+1148\sqrt{6}}{28500} & \frac{-267+88\sqrt{6}}{500} & \frac{6-\sqrt{6}}{10} & \\ \frac{4+\sqrt{6}}{10} & \frac{-32583+14638\sqrt{6}}{71250} & \frac{-17199+364\sqrt{6}}{142500} & \frac{1329-544\sqrt{6}}{2500} & \frac{-96+131\sqrt{6}}{625} & \frac{6-\sqrt{6}}{10} \\ \hline 1 & 0 & 0 & \frac{1}{9} & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36} \end{array} \tag{4.46}$$

The stability regions of the different SDIRK schemes, are shown in the Figure 4.9. We can see that the regions of absolute stability are not bounded and cover the complex negative half plane as expected. Thus, we have no restriction on the time step regarding the stability.




(a) SDIRK23 (b) SDIRK34 (c) S54b (d) SDIRK55

Figure 4.9: Stability region of the SDIRK schemes of order 3, 4 and 5.

For the accuracy we need to account for the dissipation and the dispersion. In the Figures 4.10 and 4.11 we present the relative dissipation and dispersion errors using the test equation $y' = i\lambda y$. In the $x$-coordinate we have chosen to represent $\frac{z}{s}$, because $s$ represents the computational complexity of the $s$-stages SDIRK schemes. In the Figure 4.10, the dissipation curves of SDIR23 and SDIRK34 schemes are quite similar even if the SDIRK23 scheme seems more efficient. However, S54b and SDIRK55 schemes seem far away more accurate and more efficient compared to SDIRK23 and SDIRK34 schemes which makes sense since they have more stages. In any case, the Figure 4.10 points out the need of restricting the time step to limit dissipation errors that can polute the numerical solution.

Besides the dissipation errors we observe also in the dispersion curves presented in the Figure 4.11 that S54b and SDIRK55 schemes seem more accurate and more efficient compared to SDIRK23 and SDIRK34 schemes. To avoid the dispersion errors effect on the numerical solution, we must restrict the time step as well.

The results show that the two five-stages SDIRK (SDIRK55 and S54b) schemes are less dispersive and less dissipative compared to the two- and three-stages SDIRK (SDIRK23 and SDIRK34). In the next sections we will see if the numerical results will be affected.

### 4.5.2 *Optimized low-dispersive and low-dissipative Runge-Kutta method*

In previous sections, we analysed Runge-Kutta methods. We showed that the implicit Runge-Kutta schemes have great stability over the explicit ones. However, both implicit and explicit approaches suffer from dispersion and dissipation effects. As shown in [74] before any discretisation, acoustic waves are nei-

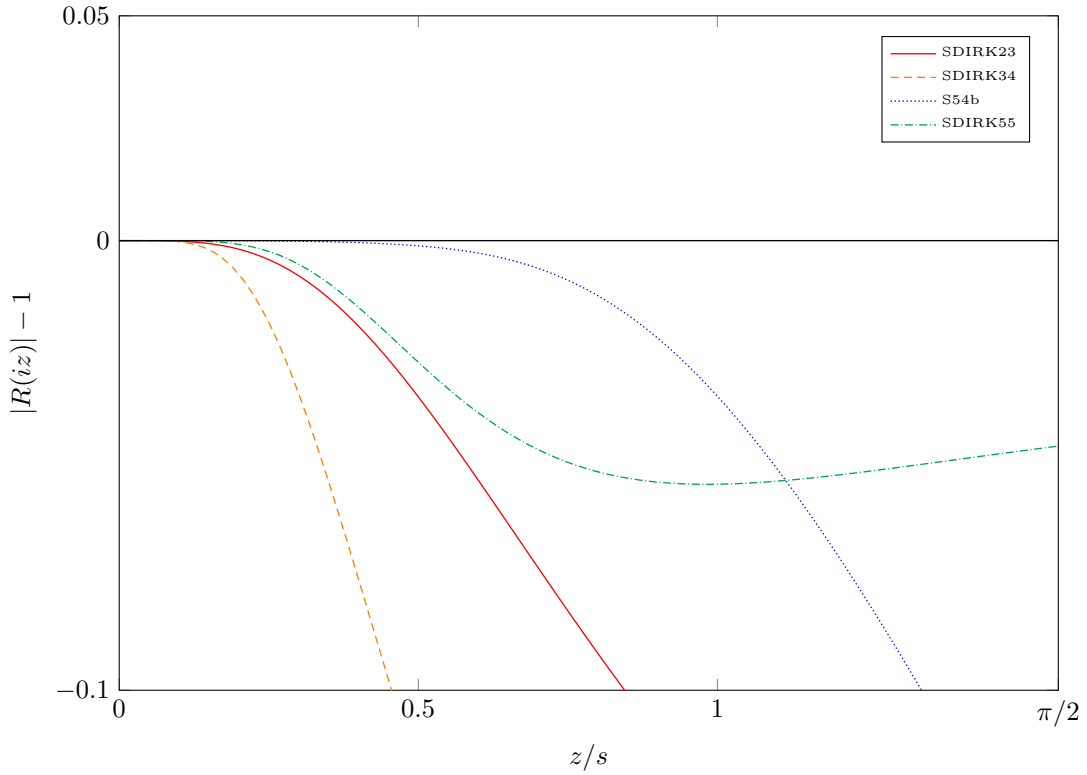



Figure 4.10: Dissipation of the SDIRK A-stbale schemes of order 3, 4 and 5.

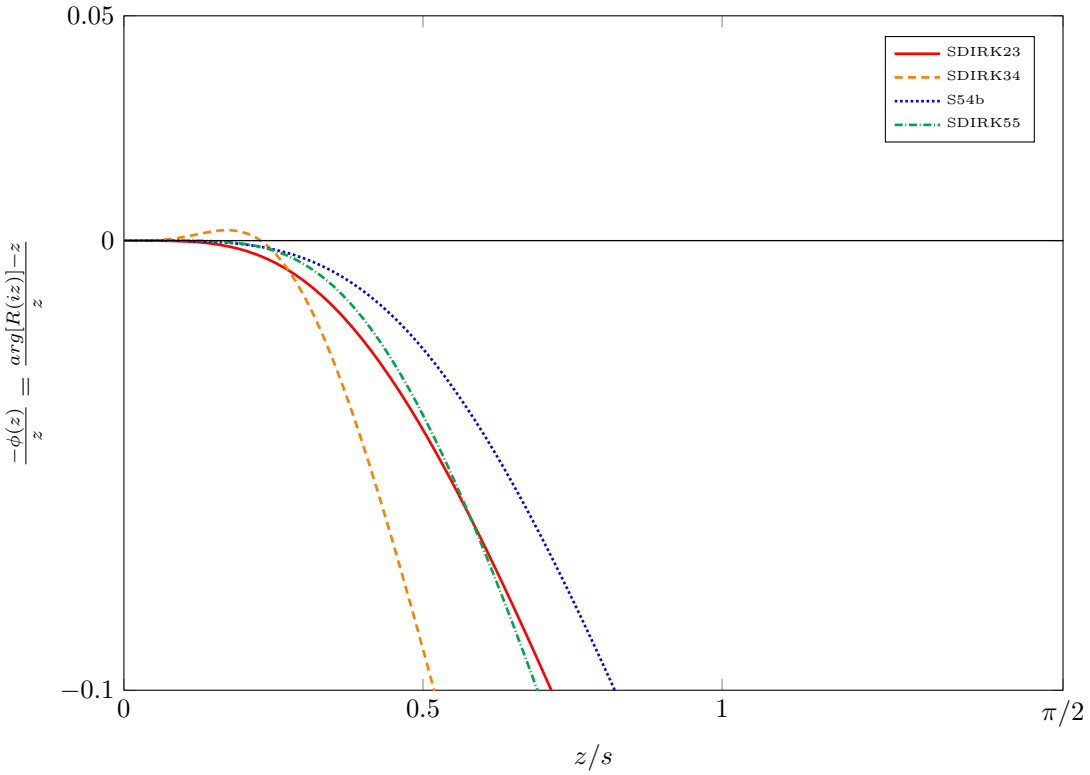



Figure 4.11: Dispersion of the A-stable SDIRK schemes of order 3, 4 and 5.

ther dispersive nor dissipative. To respect the physics of wave problems, there is thus a need of using schemes that generate the least dispersion and dissipation. Many classical Runge-Kutta methods are designed regarding the order conditions and the stability condition (the so-called CFL for explicit schemes) only. In those methods, the accuracy limit (regarding the dispersion and dissipation errors) is more stringent than the stability limit as shown in [45] for four-, five- and six-stages ERK schemes. So for ERK schemes which have a large CFL (or large stability domain), the accuracy error (phase and amplitude error) can be very large. In some implicit schemes, which are $A-$stable, there is no need of CFL. We are thus free of using a large time-step but we increase the chance of having large dispersion and dissipation errors. In those schemes, the accuracy limit must be taken into account and the step size must be chosen accordingly. In this section we seek to construct RK methods, which have large accuracy limit.

We have tried to construct A-stable SDIRK schemes by minimizing the dissipation and dispersion errors. From the works presented in [48] and the Matlab code RK-opt (a package of optimization of Runge-Kutta methods [49]) they provided, we have computed an optimal stability function that should provide low-dissipative and low-dispersive schemes. But we were not able to construct RK coefficients that satisfy all the order conditions [14]. In the Chapter 6, we provide new methods to construct high order A-stable SDIRK schemes for linear ODEs. The same method is used to construct high-order ERK schemes with optimal stability function in term of CFL in the Chapter 7 for linear ODEs.

An $A$-stable DIRK method, which minimizes the dispersion and dissipation errors, is presented in ([59]). The targeted order of accuracy $p = 4$ is maintained while designing the following three stages DIRK scheme

| | | | |
|---|---|---|---|
| 0.257820901066211 | 0.377847764031163 | | |
| 0.434296446908075 | 0.385232756462588 | 0.461548399939329 | |
| 0.758519768667167 | 0.675724855841358 | −0.061710969841169 | 0.241480233100410 |
| | 0.750869573741408 | −0.362218781852651 | 0.611349208111243 |

(4.47)

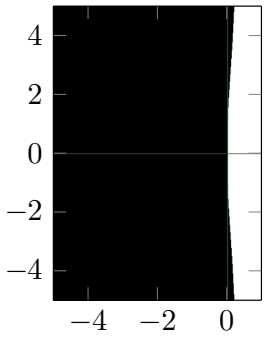
Actually this scheme doesn't verify linear relations:

$$c_i = \sum_{j=1}^{s} a_{ij}.$$

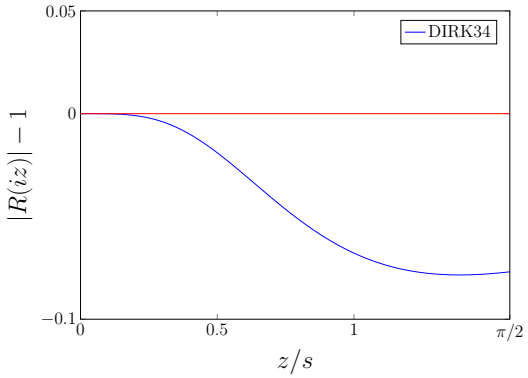
As a result, the scheme only satisfies these relations:

$$\sum_{i=1}^{s} b_i = 1, \quad \sum_{i,j=1}^{s} b_i a_{i,j} = \frac{1}{2}, \quad \sum_{i=1}^{s} b_i c_i = \frac{1}{2}$$

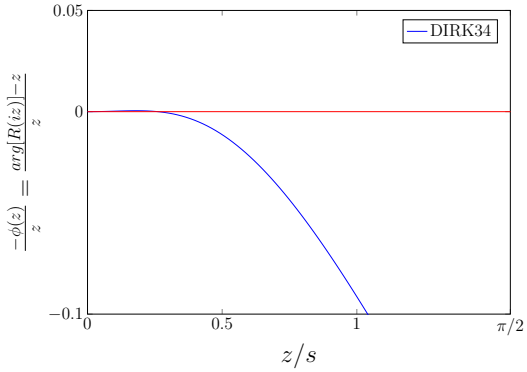
This scheme is a second-order scheme, and not a fourth-order scheme as claimed,




(a) Stability region




(b) Dissipation




(c) Dispersion

Figure 4.12: Stability region, dissipation and dispersion for the 3 stages Low-Dispersion and Low-Dissipation DIRK of order $p = 2$

since we have for instance

$$\sum_{i,j,k=1}^{s} b_i a_{i,j} a_{j,k} \neq \frac{1}{6}$$

The stability region of the scheme is shown in the Figure 4.12. The scheme (4.47) is compared to the only three-stages SDIRK, $A$-stable, fourth-order (4.44) and to the fourth order explicit RK method in [59]. It is pointed out that this scheme yields significantly reduced dispersion and dissipation errors compared to the SDIRK. As we can see in the dissipation and dispersion curves in the Figure 4.12 it seems to be more accurate than the ERK and the SDIRK methods. This makes sense since DIRK34, constructed to minimize the dissipation and dispersion errors, is a 3-stages scheme of order 2 only and not 4 according to the order conditions.

To summarise, we have seen that all ERK methods have a numerical stability limitation (CFL): the time-step $h$ must be chosen such that $z = h\lambda$ get into the stability region. Many works have been done on the construction of ERK schemes with maximal CFL (see [48] and [54]). Some works have focused on

constructing ERK schemes with low-dissipation and low-dispersion up to order 4. The reader can find more details on that in [3], [45], [10].

For some IRK methods, like those presented in this chapter, no time-step restriction is needed to ensure stability. They are unconditionally stable, because their stability region covers the negative half plane. But they all have a common drawback which is due to the dissipation and the dispersion errors.

We have investigated other methods to construct A-stable Runge-Kutta schemes by minimizing the dissipation and dispersion errors due the schemes. Using [48], we have not succeeded in the construction of the Runge-Kutta coefficients due to numerous order conditions plus the constraints related to the dissipation, the dispersion and the A-stability. In the next section, we provide numerical results on solving the acoustic wave equation in 1D and 2D. The different schemes presented in this chapter are compared and the performance assessments are given.

## 4.6 Applications

We have implemented all the IRK methods presented here in the high-order finite element code **Montjoie** [31] at any order. In this section we use the different RK-schemes for the solution of the semi-discrete acoustic wave equation in 1D and 2D.

First we recall the first acoustic wave equation we are solving. 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:

$$\begin{cases} \rho\,\partial_t u - \mathrm{div}\, v = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ \mu^{-1}\partial_t v - \nabla u = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ u(x,0) = \partial_t u(x,0) = 0, \quad \forall x \in \Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\ \mu \partial_n u = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu \partial_n u + \sqrt{\rho\mu}\,\partial_t u = f_A, \quad x \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{4.48}$$

where $\Omega$ is the computational domain. $\Gamma_D$, $\Gamma_N$ and $\Gamma_A$ are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. $n$ is the outgoing normal defined for the considered boundary, $\rho$ and $\mu$ are physical indexes, which are piecewise constant. $f_D$, $f_N$ and $f_A$ are given source functions. The details of the space discretization of this equation are presented in the Chapter 3.

The wave equation (4.48) is solved in a 1D homogeneous medium

$$\rho = \mu = 1$$

| Schemes | Time-step | Relative errors | Computatinal time (s) |
|---|---|---|---|
| ERK22 | 1e-04 | not stable | |
| ERK22 | 1e-05 | 1.96e-07 | 137 626.24 |
| GaussRK2 | 16e-04 | 2.41e-03 | 982.49 |
| GaussRK2 | 5e-04 | 2.45e-04 | 3 137.37 |
| GaussRK2 | 1e-05 | 9.81e-08 | 156 784.45 |
| ERK44 | 1e-03 | not stable | |
| ERK44 | 5e-04 | 3.28e-10 | 5 673.16 |
| SDIRK34 | 0.01 | 1.02e-03 | 2 729.17 |
| DIRK34 | 0.01 | 3.92e-03 | 2 973.46 |

Table 4.1: Illustration of the stability issues of ERK schemes (1-D case).

| Schemes | Number of stages | Time-steps | Relative errors |
|---|---|---|---|
| SDIRK34 order 4 | 3 | $\Delta t = 0.048$ | 0.3376 |
| S54b order 4 | 5 | $\Delta t = 0.08$ | 0.0216 |
| DIRK34 order 4 | 3 | $\Delta t = 0.048$ | 0.0965 |
| SDIRK55 order 5 | 5 | $\Delta t = 0.08$ | 0.0195 |

Table 4.2: Comparison of the efficiency of the DIRK and SDIRK schemes (1-D case).

in the computational domain $\Omega = [0, 20]$. A homogeneous Neumann condition is set on the left and right extremities. In space, mixed spectral elements of order 10 are used. The computational domain $\Omega$ is subdivided into 1000 regular subintervals. We choose $[0, 40]$ for the time interval. We first compare explicit and implicit methods to show the main drawback of the ERK schemes. In the Table 4.1 we present the comparative results obtained using the different schemes with different time-steps: as expected ERK-methods are not always stable, unless we take a very small time step which leads to exhaustive computational cost. We can also see that IRK-schemes presented here are stable and allows to choose a bigger time-steps. For large time-steps we have less computational times as expected, but the relative error is increased compared to the one corresponding to the explicit scheme. However with the same time step the implicit schemes (GaussRK2 here) are more accurate than the explicit schemes (ERK22 here) and also require more computational time as expected.

The second test we performed is to compare different high order implicit schemes. To be able to compare different schemes with different stages, we fixed $\Delta t = 0.8$ for $s = 5$ then for all $s \in \mathbb{N}$, we take $\Delta t_s = \dfrac{\Delta t}{5} \times s$. The aim of this comparison is mainly to see if the dispersion and dissipation analysis results are coherent with the numerical results presented in the Table 4.2.

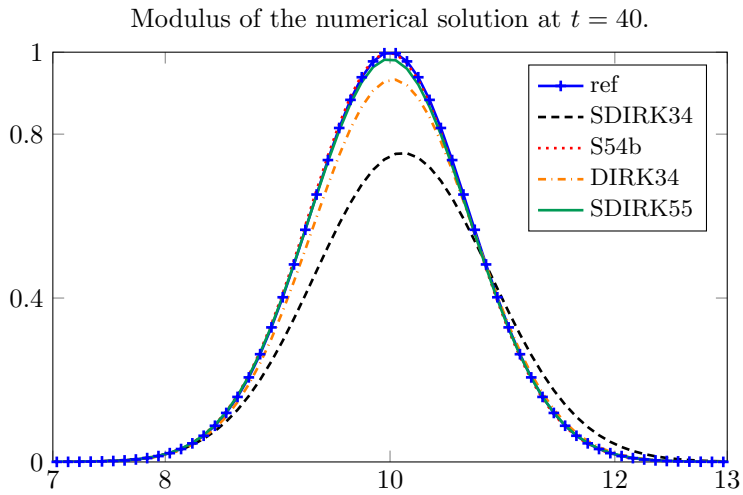



Figure 4.13: Modulus of the numerical solution at the final time $t = 40$ for the SDIRK schemes.

| Time-steps | Relative errors | Computational time (s) |
|---|---|---|
| 0.0016 | 0.0025 | 982.49 |
| 0.016 | 0.2495 | 98.72 |
| 0.032 | 0.9221 | 49.56 |

Table 4.3: Illustration of the phase shift of the A-stable GaussRK2 schemes (1-D case).

In these results S54b and SDIRK55 schemes have around $2\%$ of error. The DIRK34 error is about $10\%$ while it is about $34\%$ for SDIRK34. This confirms the efficiency of S54b and SDIRK55 in terms of dispersion and dissipation compared to the two others. This is illustrated in the Figure 4.13, where we can see both a large difference of amplitude and phase for SDIRK34 scheme (dashed line). The same thing is visible for DIRK24 (dash-dotted line).

To stress the dispersion error issues in wave propagation problems, we performed a third numerical test with the unconditionally stable scheme GaussRK2. In the Table 4.3, we present the numerical results for the 1D acoustic wave equation obtained when using GaussRK2 shceme for the time discretization. We present the results for different choice of time-step.

In the Table 4.3, we can see a small computational time for a large time step which was expected but we have $92\%$ errors for $t = 0.032$. By halving the time-step, the error goes down to $25\%$. We conclude that for large time-steps, the numerical solution is not clearly in phase with the exact solution. For wave propagation problems, this makes the wave propagate at a wrong phase velocity. The modulus of the numerical solution at the final time step $t = 40$ is ploted in the Figure 4.14. The solution is plotted on $[3, 15] \subset \Omega$.

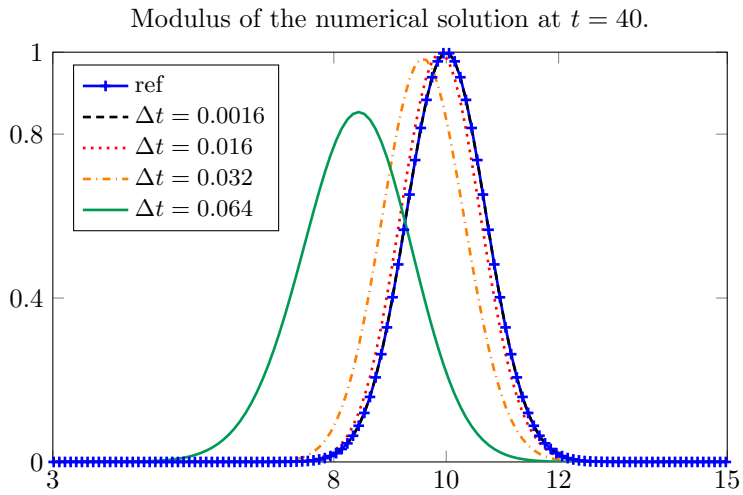



Figure 4.14: Illustration of the phase shift in the numerical solution of the 1D acoustic wave equation when using GaussRK2 scheme with different time-steps.

## 4.7 Conclusion

In this chapter we have presented classical Runge-Kutta schemes that are commonly used in the literature. We have presented explicit RK schemes up to order 8. We have shown that the fourth order schemes ERK44 are most efficient for hyperbolic equations including wave equations. Explicit time integration is stable under a CFL condition which may hamper the performances of the numerical method. We have thus considered implicit schemes that are unconditionally stable (A-stable). With such schemes we have no constraint on the choice of the time step regarding the stability. However, the dissipation and dispersion errors analysis showed that implicit unconditionally stable schemes have to be constrained by a given accuracy limit which leads to some limitation on the size of the time-step.

# Part II

# Construction of time integration schemes

Chapter 5

# A-stable high-order diagonal Padé Schemes for linear ODEs

In this chapter we address the problem of constructing high order A-stable time integration schemes based on Padé approximations of the exponential operator representing the solution. The developed schemes have the same stability function as the Gauss Runge-Kutta schemes but they work only for linear ODEs. We present the corresponding algorithm of implementation and in particular we detail a method to handle a source term. We provide numerical results when solving the acoustic wave equation in order to validate the Padé schemes we have developed and to assess their performance, we propose some comparisons with Runge-Kutta methods that are commonly used in the literature.

## Contents

## 5.1 Introduction

As we have introduced in the Chapter 2, the analytical solution of an ODE involves the exponential of a matrix. The exponential of a matrix can be computed in many ways [56]. Herein, we use a rational approximation of the exponential kwown as the Padé approximant. The motivation of this chapter is to construct a time integration scheme based on Padé approximations of exponential function. Introduced by Henri Padé (1863-1953), Padé approximations are known to provide accurate approximations of exponential functions by rational polynomials (see the Definition 4.4.1).

Herein, we mainly focus our study on a particular Padé approximant called diagonal Padé approximant as introduced in the Definition 4.4.2. The resulting schemes are called diagonal Padé schemes. We construct them at any order of accuracy.

### *5.1.1 General settings and stability function*

As previously, we consider the following ODE

$$\begin{cases} M_h \dfrac{dX(t)}{dt} + K_h X(t) = F(t) \quad t \in (0, T] \\ X(0) = X_0 \end{cases} \tag{5.1}$$

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 Chapter 3, we have detailed how this ODE is obtained in the case of the 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 $\Delta t$ is the time step. The analytical solution to (5.1) after one iteration of time is given by

$$X(t_{n+1}) = \mathrm{e}^{\Delta t A} \left( X(t_n) + \int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\, \mathrm{d}u \right), \tag{5.2}$$

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 chapter will

consist of computing a sequence $X_n$, which is an approximation of the analytical solution $X(t_n)$, applying the following numerical scheme:

$$X_{n+1} = R(\Delta t A)X_n + \widetilde{\phi}_n \tag{5.3}$$

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

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

According to the Definition 2.3.1 $R$ is the stability function of the corresponding scheme. Its stability region $\mathcal{S}$ is then defined as in the Definition 2.3.2.

Let us define the stability function of a Padé schemes as:

$$R_{r,s}(z) = \frac{N_{r,s}(z)}{D_{r,s}(z)} \tag{5.4}$$

where $N_{r,s}(z)$ and $D_{r,s}(z)$ are polynomials of $z$ defined by:

$$N_{r,s}(z) = \sum_{i=0}^{s} \frac{s!\ (r+s-i)!}{(r+s)!\ i!\ (s-i)!}(z)^i \quad \text{and} \quad D_{r,s}(z) = \sum_{i=0}^{r} \frac{r!\ (r+s-i)!}{(r+s)!\ i!\ (r-i)!}(-z)^i. \tag{5.5}$$

From the Definition 4.4.1, $R_{r,s}(z)$ is an approximation of order $(r+s)$ of $\mathrm{e}^z$.

As we have seen in the Chapter 2, if the degree $r$ of the denominator is greater or equal to one ($r \geq 1$) the corresponding scheme is implicit. Furtheremore, since the A-stable requirement excludes rational functions that tend towards infinity when z tends to infinity, the degree of $D_{r,s}$ must be greater or equal to the degree of $N_{r,s}$ ( i.e. $r \geq s$).

In [34] and [33], Ehle showed that in the cases $r = s$, $r = s+1$ and $r = s+2$ the schemes having $R_{r,s}$ for stability function will be $A$-stable.

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

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

Notice that the stability function for diagonal Padé schemes is the same as the stability function for Gauss-Runge-Kutta schemes (see the Theorem 4.4.3). 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. In fact, instead of solving a large system of $m \times L$ with Gauss RK schemes, we solve at most $m$ smaller linear systems of size $L$ with diagonal Padé schemes ($m$ is the degree of the denominator of the stability function and $L$ is the number of unknowns).

Introducing $C = \Delta t A$, we define $R_m(C) = [D_m(C)^{-1}]N_m(C)$ as an approximation of $\mathrm{e}^C$ of order $p$ ($\mathrm{e}^C = R_m(C) + O(\Delta t^{p+1})$), we assume that $D_m(C)^{-1}$ is well defined. The analytical solution (5.2) can then be written as

$$D_m(C)X(t_{n+1}) = N_m(C)X(t_n) + \phi, \tag{5.6}$$

where

$$\phi = N_m(C)\int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u + O(\Delta t^{p+1}). \tag{5.7}$$

For homogeneous ODEs ($F(t) = 0$), it follows from (5.7) that $\phi = O(\Delta t^{p+1})$. Then, the numerical solution (5.3) satisfies

$$D_m(C)X_{n+1} = N_m(C)X_n \tag{5.8}$$

For inhomogeneous ODEs, i.e. $F(t) \neq 0$, we need to compute the quantity $\phi$. However the integral in the expression (5.7) of $\phi$ is tedious to compute. We will rather compute the following equivalent quantity obtained from (5.6):

$$\phi = D_m(C)X(t_{n+1}) - N_m(C)X(t_n) \tag{5.9}$$

Finally, we propose the following numerical scheme:

$$\boxed{D_m(C)X_{n+1} = N_m(C)X_n + \phi_n} \tag{5.10}$$

where $\phi_n$ is an approximation of $\phi$ (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 $t_n + \dfrac{\Delta t}{2}$ and using derivatives of the equation (5.1), we write $\phi_n$ in the following form:

$$\phi_n = \sum_{r=1}^{m} A^{r-1}\Delta t^r \sum_{i=0}^{n_w-1} \omega_i^r\, F\left(t_n + \Delta t\, c_i\right) \tag{5.11}$$

where $m$ is the degree of the polynomial $N_m$ and $D_m$ and $n_w = m$ for the diagonal Padé schemes. We will go back on that process later on subsection 5.2.2.

### 5.1.2 Numerical stability, dissipation and dispersion

In this subsection we discuss the stability, dissipation and dispersion properties of the diagonal Padé schemes. First we recall that by definition of $R_m(z) = \frac{N_m(z)}{D_m(z)}$, we have $D_m(z) = N_m(-z)$. It follows from the Theorem $353A$ in [14] and the fact that $|N_m(ib)| = |N_m(-ib)|$, $\forall b \in \mathbb{R}$ that we have:

**Proposition 5.1.1** *The stability function of numerical schemes obtained using the diagonal Padé approximation satisfies:* $\forall z \in \mathbb{C}^-$

$$|R_m(z)| \leq 1, \quad \forall m \in \mathbb{N}. \tag{5.12}$$

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

$$|R_m(z)| = 1, \quad \forall m \in \mathbb{N}. \tag{5.13}$$

As a consequence,

- the diagonal Padé schemes are $A$-stable, according to the Definition 2.15,
- the diagonal Padé schemes when applied to the test equation $y'(t) = i\lambda y(t)$, $\lambda \in \mathbb{R}$, are not dissipative following the Definition 2.4.1.

For additional proof of the inequality (5.12) in Proposition 5.1.1 we present the following property of the diagonal Padé schemes:

**Proposition 5.1.2** *The stability functions $R_m(z) := \frac{N_m(z)}{D_m(z)}$ of the diagonal Padé schemes are holomorphic in the negative complex plane $\mathbb{C}^-$.*

**Proof** Let $\lambda_k$, $k = 1, \ldots, m$ be the roots of $D_m$, the denominator of the stability function $R_m$ of diagonal Padé scheme. To prove the Proposition 5.1.2 we just need to show that $\lambda_k$ satisfies

$$Re(\lambda_k) > 0, \forall \lambda_k, \; k = 1, \ldots, m,$$

since $D_m$ is a polynomial function. To reach that objective, we use the same approach as in [34] for $R_{m,m+1}(z)$. Since we have $D_m(z) = N_m(-z)$, showing that $D_m(z)$ never vanishes in the negative half plane ($R_e(z) < 0$) is equivalent to show that $N_m(z)$ never vanishes in the positive half plane ($Re(z) > 0$). We then notice that

$$N_m(z) = \frac{m!}{(2m)!} B_m^1(z),$$

where $B_m^\delta$ stands for the Bessel polynomials defined by

$$B_m^\delta(z) = \sum_{k=0}^{m} \binom{m}{k} (m+\delta)_k \, z^{m-k},$$

with

$$\begin{aligned} (m+\delta)_k &= (m+\delta)(m+\delta+1)\ldots(m+\delta+k-1), \\ (m+\delta)_0 &= 1. \end{aligned}$$

We know from [78] that all the zeros of the Bessel polynomials are in the negative half plane for $\delta > 0$ and $n \geq 1$ which concludes the proof.

This result means that $R_m(z)$ has no singularity for $Re(z) < 0$. It allows us to state that $R_m(z)$ is an holomorphic function as a product of an holomorphic function by the inverse of a holomorphic function that is nowhere zero in the domain $\mathbb{C}^-$. □

Having $R_m(z)$ holomorphic implies that the diagonal Padé scheme is A-stable since from the equation (5.13) of the Proposition 5.1.1 we can deduce that

$$|R_m(iy)| \leq 1, \; \forall y \in \mathbb{R}. \tag{5.14}$$

**Remark** The condition (5.14) states that we have stability on the imaginary axis. This condition is generally weaker than the A-stability condition. But in the particular case of diagonal Padé schemes, it is equivalent to the A-stability condition $|R_m(z)| \leq 1,\ \forall z \in \mathbb{C}^-$ presented in Proposition 5.1.1. In fact,

- Assume that $|R_m(iy)| \leq 1,\ \forall y \in \mathbb{R}$. We know from the Proposition 5.1.2 that $R_m(z),\ z \in \mathbb{C}^-$ is a holomorphic function. Then, by the maximum modulus principle, $|R_m(z)|$ has no local maximum in $\mathbb{C}^-$. That means

$$\max_{z\in\mathbb{C}^-} |R_m(z)| = \max_{y\in\mathbb{R}} |R_m(iy)| \leq 1,$$

  which gives the $A$-stability.

- Now let us suppose that the diagonal Padé scheme is $A$-stable. Then by definition of the A-stability, $|R_m(iy)| \leq 1,\ \forall y \in \mathbb{R}$.□

To recap, $R_m(z)$ verifies the A-stability condition and all its poles (zeros of $D_m$) are located on the positive half-plane. Furthermore, $D_m$ has at most one real root [63] 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 [69] in which the authors showed that all the poles of $R_m$ converge to a curved right-side section of Szegö's curve.

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

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

In Figure 5.1, we present the relative dispersion error of diagonal Padé schemes from order 2 to 10. As formerly done, we use as abscissa $\frac{z}{m}$, recalling that $m$ 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 assuming that the computational cost of a complex system matches the one of two real system. The case $m = 3$ corresponds to 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, and see clearly that the dispersion error is much smaller with higher order schemes.

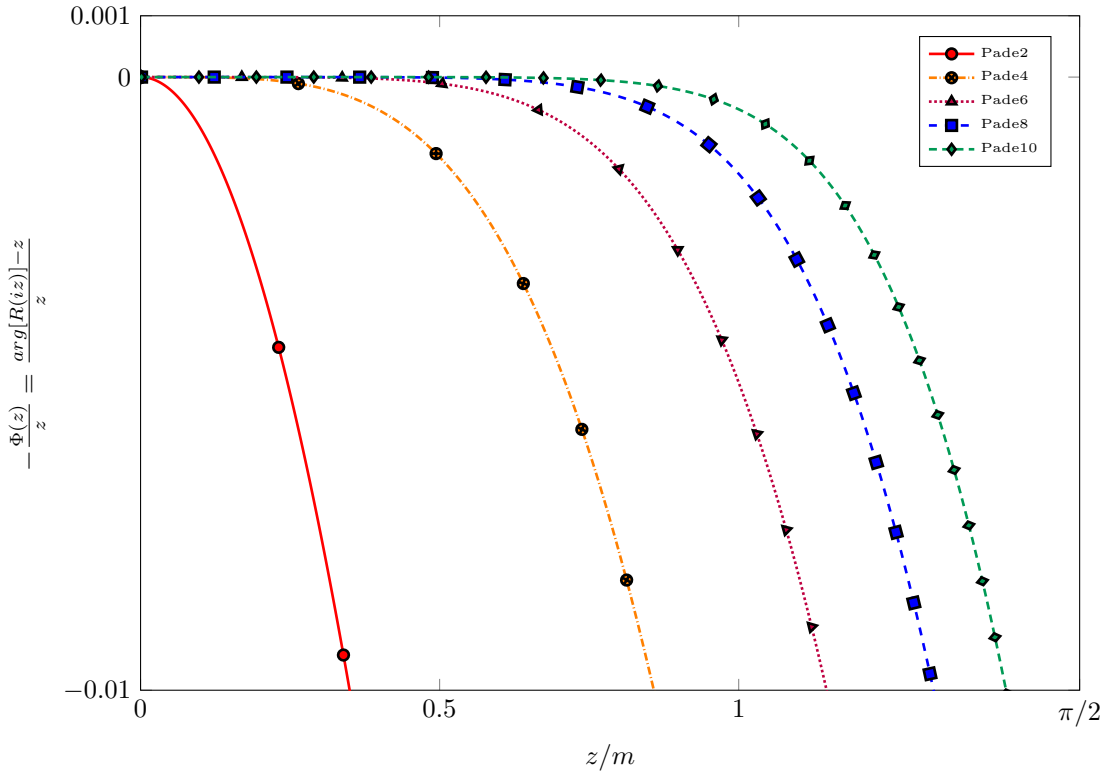



Figure 5.1: Dispersion of diagonal Padé schemes of order 2, 4, 6, 8 and 10 when applied to the test equation $y'(t) = i\lambda y(t)$.

## 5.2 Efficient implementation and computation of the source term

### *5.2.1 Homogeneous case*

In the homogeneous case, we consider (5.8) 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 terms. It is then easy to compute the right hand side $(N_m - D_m)(C)X_n$ using Hörner's algorithm. We note $G = (N_m - D_m)(C)X_n$ Now we have to solve the real linear system

$$D_m(C)(X_{n+1} - X_n) = G, \tag{5.15}$$

for each time step.

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

$$\left[\prod_{k=1}^{m}\left(I - \frac{C}{\lambda_k}\right)\right] Y_m = G. \tag{5.16}$$

Then we can solve successively the linear systems

$$\left(I - \frac{C}{\lambda_k}\right) Y_k = Y_{k-1} \quad k = 1, \ldots, m \tag{5.17}$$

$$Y_0 = G. \tag{5.18}$$

The final result $Y_m$ is the targeted 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 and that is convenient for parallelization [35].

Using the algebraic properties of $D_m$, we can optimize the computational 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 $D_m$:

$$\prod_{k=1}^{m/2}\left(I - \frac{C}{\lambda_k}\right)\left(I - \frac{C}{\overline{\lambda_k}}\right) Y_m = G. \tag{5.19}$$

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

$$\begin{aligned} &\left(I - \frac{C}{\lambda}\right) Y_0 = G, \\ &\prod_{k=1}^{(m-1)/2} \left(I - \frac{C}{\lambda_k}\right)\left(I - \frac{C}{\overline{\lambda}_k}\right)\; Y_m = Y_0. \end{aligned} \tag{5.20}$$

We solve the second degree equation using the following algorithm proposed in [4]: let $\lambda_k$ be a complex root of $D_m$, $a_k = -\dfrac{1}{\lambda_k}$ and $\overline{a}_k = -\dfrac{1}{\overline{\lambda}_k}$. 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 fractional decomposition of $P_2^{-1}(x) = \dfrac{1}{(1 + a_k x)(1 + \overline{a}_k x)}$ allows to write

$$P_2^{-1}(C) = b_k (I + a_k C)^{-1} + \bar{b}_k (I + \overline{a}_k C)^{-1}, \tag{5.21}$$

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

$$\begin{aligned} &(I + a_k C)\, u = Y_{k-1}, \\ &Y_k = b_k u + \bar{b}_k \overline{u} = 2\, Re(b_k u). \end{aligned} \tag{5.22}$$

The case where iterates $X_n$ are complex can be addressed by solving system (5.19) or (5.20) twice, each for the real and imaginary parts of $G$.

### 5.2.2 Computation of the right hand side (RHS) $\phi$

In the inhomogeneous case ($F(t) \neq 0$), we need to compute the coefficients $\omega_i^r$ involved in (5.11). This is done by using the Taylor expansion around $t_n + \dfrac{\Delta t}{2}$ of $\phi$ at order $p = 2m$. This expansion is completed with the expression (5.9) that we recall here:

$$\phi = D(C) X(t_{n+1}) - N(C) X(t_n) \tag{5.23}$$

We introduce the following notations

$$\rho_i^m = \frac{m!\,(2m-i)!}{(2m)!\; i!\; (m-i)!} = \binom{m}{i} \frac{(2m-i)!}{(2m)!} \quad \text{and} \quad C_k = \frac{1}{k!\; 2^{k-1}}. \tag{5.24}$$

Then, we have:

$$N_m(z) = \sum_{i=0}^{m} \rho_i^m z^i \qquad \text{and} \qquad D_m(z) = N_m(-z). \tag{5.25}$$

We perform a 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 $2m$

$$\begin{aligned} X(t_n + \Delta t) &= \sum_{k=0}^{2m} \frac{\Delta t^k}{2^k} \frac{1}{k!} X^{(k)} \left( t_n + \frac{\Delta t}{2} \right) + O(\Delta t^{2m+1}), \\ X(t_n) &= \sum_{k=0}^{2m} \frac{(-\Delta t)^k}{2^k} \frac{1}{k!} X^{(k)} \left( t_n + \frac{\Delta t}{2} \right) + O(\Delta t^{2m+2}). \end{aligned} \tag{5.26}$$

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 (5.23). After performing the Taylor expansion it gives

$$\begin{aligned} \phi = & \sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k X^{(k)} \\ & - \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k X^{(k)} + O(\Delta t^{2m+1}). \end{aligned} \tag{5.27}$$

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

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

to obtain:

$$X^{(k)} = \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + A^k X^{(0)}, \tag{5.28}$$

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 (5.28) in expression (5.27) gives:

$$\begin{aligned} \phi = & \sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} \\ & - \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} \\ & + S_m + O(\Delta t^{2m+1}), \end{aligned} \tag{5.29}$$

where $S_m$ is given by

$$\begin{aligned} S_m = & \sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k(\Delta t\, A)^{k+i} X^{(0)} \\ & - \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q,\, q\in\mathbb{N}}^{2m} C_k(\Delta t\, A)^{k+i} X^{(0)}. \end{aligned} \tag{5.30}$$

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 (5.8) (with $F = 0$) would be of order lower than $2m$ which is in contradiction with the properties of Padé approximants of the exponential. We first take $m = 1$ and we arrange the sum (5.30) with powers of $z = \Delta t A$. It gives

$$S_1 = (\rho_0^1 C_1 - \rho_1^1 C_0)\Delta t A X + O(\Delta t^3) = O(\Delta t^3),$$

since $\rho_0^1 C_1 - \rho_1^1 C_0 = 0$.

Using the same method for $m = 2$, we get

$$\begin{aligned} S_2 = & \rho_0^2(C_1 \Delta t A X + C_3 \Delta t^3 A^3 X) \\ & - \rho_1^2(C_0 \Delta t A X + C_2 \Delta t^3 A^3 X + \underbrace{C_4 \Delta t^5 A^5 X}_{O(\Delta t^5)}) \\ & + \rho_2^2(C_1 \Delta t^3 A^3 X + \underbrace{C_3 \Delta t^5 A^5 X}_{O(\Delta t^5)} \\ = & (\rho_0^2 C_1 - \rho_1^2 C_0)\Delta t A X \\ & + (\rho_0^2 C_3 - \rho_1^2 C_2 + \rho_2^2 C_1)\Delta t^3 A^3 X + O(\Delta t^5) \\ = & O(\Delta t^5). \end{aligned}$$

In fact, we have a simple computation shows that

$$\begin{cases} \rho_0^2 C_1 - \rho_1^2 C_0 = 0 \\ \rho_0^2 C_3 - \rho_1^2 C_2 + \rho_2^2 C_1 = 0. \end{cases}$$

Actually, this property can be proved $\forall m \in \mathbb{N}$ giving by this way a demonstration of the order of the scheme.

**Proposition 5.2.1** *For all* $m \in \mathbb{N}$,

$$S_m = \sum_{r=2p+1,\, p\in\mathbb{N}}^{2m-1} \zeta_r^m (\Delta t A)^r X^{(0)} + O(\Delta t^{2m+1}), \tag{5.31}$$

*with*

$$\zeta_r^m = \sum_{i=0}^{min(m,r)} (-1)^i \rho_i^m C_{r-i} \tag{5.32}$$

**Proof** Let $r$ be an integer defined by $r = k + i$. The result comes by developing the sum (5.30) and sorting by powers of $\Delta t A$. □

**Theorem 5.2.2** *Let $\zeta_r^m$ be defined in (5.32). Then we have*

$$\zeta_r^m = 0, \quad r = 2p - 1,\ 1 \leq p \leq m, \tag{5.33}$$

*which implies that*

$$S_m = O(\Delta t^{2m+1}). \tag{5.34}$$

The proof of the Theorem 5.2.2 involves the following lemma:

**Lemma 5.2.3** *Let $\Upsilon_{m,r}$ be the polynomial defined by*

$$\Upsilon_{m,r}(x) = \frac{(-1)^r}{(2m)!\ 2^{r-1}} x^m (x+2)^m. \tag{5.35}$$

*Then*

$$\frac{d^{2m-r}\Upsilon_{m,r}}{dx}(-1) = \zeta_r^m, \quad r = 2p - 1,\ 1 \leq p \leq m. \tag{5.36}$$

**Proof** We use Newton's binomial formula to develop (5.35) 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 is evaluated at $x = -1$; we get

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

**Proof** Theorem 5.2.2 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 $r = 2p - 1$. □

Applying Theorem 5.2.2 to (5.29) implies the following result.

**Corollary 5.2.4** *The simplified expression of $\phi$ defined in (5.29) is given by*

$$\begin{aligned}\phi = &\sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} \\ &- \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m (\Delta t A)^i \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m} C_k \Delta t^k \sum_{j=1}^{k} A^{k-j} F^{(j-1)} + O(\Delta t^{2m+1}).\end{aligned} \tag{5.37}$$

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

$$\begin{aligned}\phi = &\sum_{i=2p,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+1,\, q\in\mathbb{N}}^{2m-1-i} C_k \Delta t^{k+i} \sum_{j=1}^{k} A^{i+k-j} F^{(j-1)} \\ &- \sum_{i=2p+1,\, p\in\mathbb{N}}^{m} \rho_i^m \sum_{k=2q+2,\, q\in\mathbb{N}}^{2m-1-i} C_k \Delta t^{k+i} \sum_{j=1}^{k} A^{i+k-j} F^{(j-1)} \\ &+ O(\Delta t^{2m+1}).\end{aligned} \tag{5.38}$$

We change indexes in the sum by introducing

$$r = i + k - j + 1$$

We then obtain the following expression:

$$\begin{aligned}\phi = &\sum_{r=1}^{2m-1} \Delta t^r A^{r-1} \sum_{j=1, j-r=2q, q\in\mathbb{Z}}^{2m-r+1} \Delta t^{j-1} F^{(j-1)} \\ &\left( \sum_{i=2p, p\in\mathbb{N}}^{min(m,r-1)} \rho_i^m C_{r+j-i-1} - \sum_{i=2p+1, p\in\mathbb{N}}^{min(m,r-1)} \rho_i^m C_{r+j-i-1} \right) + O(\Delta t^{2m+1})\end{aligned} \tag{5.39}$$

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

$$\alpha_j^r = \begin{cases} \displaystyle\sum_{i=0}^{min(m,r-1)} (-1)^i \rho_i^m C_{r+j-i}, & \text{if } j-r \equiv 1[2] \\ 0 \quad \text{otherwise} \end{cases} \tag{5.40}$$

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, $\phi$ is written as:

$$\phi = \sum_{r=1}^{m} \Delta t^r A^{r-1} \sum_{j=1}^{2m-r+1} \alpha_{j-1}^r \Delta t^{j-1} F^{(j-1)} + O(\Delta t^{2m+1}) \tag{5.41}$$

To illustrate (5.41), we provide then the expression of $\phi$ for three particular values of $m$:

**Fourth-order diagonal Padé scheme** We take $m = 2$ in (5.38) and arrange with powers of $A$:

$$\begin{aligned} \phi = \Delta t \left(\rho_0^2 C_1 F + \rho_0^2 C_3 \Delta t^2 F^{(2)}\right) + A\Delta t^2 \left(\rho_0^2 C_3 \Delta t F^{(1)} - \rho_1^2 C_2 \Delta t F^{(1)}\right) \\ + A^2 \Delta t^3 \underbrace{\left(\rho_0^2 C_3 - \rho_1^2 C_2 + \rho_2^2 C_1\right)}_{=\zeta_3^2=0} F + O(\Delta t^5). \end{aligned}$$

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

$$\phi = \Delta t \left(F + \frac{\Delta t^2}{24} F^{(2)}\right) - A \frac{\Delta t^3}{12} F^{(1)} + O(\Delta t^5). \tag{5.42}$$

**Sixth-order diagonal Padé scheme** As for the fourth-order, we take $m = 3$ in (5.38) and arrange with powers of A. We obtain

$$\begin{aligned} \phi = &\Delta t \left(\rho_0^3 C_1 F + \rho_0^3 C_3 \Delta t^2 F^{(2)} + \rho_0^3 C_5 \Delta t^4 F^{(4)}\right) \\ &+ A\Delta t^2 \left(\rho_0^3 C_3 \Delta t F^{(1)} + \rho_0^3 C_5 \Delta t^3 F^{(3)} - \rho_1^3 C_2 \Delta t F^{(1)} - \rho_1^3 C_4 \Delta t^3 F^{(3)}\right) \\ &+ A^2 \Delta t^3 \left(\rho_0^3 C_3 F + \rho_0^3 C_5 \Delta t^2 F^{(2)} - \rho_1^3 C_2 F - \rho_1^3 C_4 \Delta t^2 F^{(2)} + \rho_2^3 C_1 F + \rho_2^3 C_3 \Delta t^2 F^{(2)}\right) \\ &+ A^3 \Delta t^4 \underbrace{\left(\rho_0^3 C_5 - \rho_1^3 C_4 + \rho_2^3 C_3 - \rho_3^3 C_2\right)}_{=\zeta_5^3=0} \Delta t F^{(1)} \\ &+ A^4 \Delta t^5 \underbrace{\left(\rho_0^3 C_5 - \rho_1^3 C_4 + \rho_2^3 C_3 - \rho_3^3 C_2\right)}_{=\zeta_5^3=0} F + O(\Delta t^7). \end{aligned}$$

The formula (5.32) gives $\zeta_r^m = 0$ and the source vector $\phi$ for the sixth-order diagonal Padé scheme is given by

$$\begin{aligned}\phi = \Delta t \left( F + \frac{\Delta t^2}{24} F^{(2)} + \frac{\Delta t^4}{1920} F^{(4)} \right) - A\Delta t^2 \left( \frac{\Delta t}{12} F^{(1)} + \frac{\Delta t^3}{480} F^{(3)} \right) \\ + A^2 \Delta t^3 \left( \frac{1}{60} F + \frac{\Delta t^2}{480} F^{(2)} \right) + O(\Delta t^7).\end{aligned} \tag{5.43}$$

**Eighth-order diagonal Padé scheme** The eighth-order source approximation for diagonal Padé scheme is obtained for $m = 4$ in (5.38). Then we arrange the sums with powers of $A$ to get:

$$\begin{aligned}\phi &= \Delta t \left( \rho_0^4 C_1 F + \rho_0^4 C_3 \Delta t^2 F^{(2)} + \rho_0^4 C_5 \Delta t^4 F^{(4)} + \rho_0^4 C_7 \Delta t^6 F^{(6)} \right) \\ &+ A\Delta t^2 \left\{ \left( \rho_0^4 C_3 - \rho_1^4 C_2 \right) \Delta t F^{(1)} + \left( \rho_0^4 C_5 - \rho_1^4 C_4 \right) \Delta t^3 F^{(3)} + \left( \rho_0^4 C_7 - \rho_1^4 C_6 \right) \Delta t^5 F^{(5)} \right\} \\ &+ A^2 \Delta t^3 \left\{ \begin{array}{r} \left( \rho_0^4 C_3 - \rho_1^4 C_2 + \rho_2^4 C_1 \right) F + \left( \rho_0^4 C_5 - \rho_1^4 C_4 + \rho_2^4 C_3 \right) \Delta t^2 F^{(2)} \\ + \left( \rho_0^4 C_7 - \rho_1^4 C_6 + \rho_2^4 C_5 \right) \Delta t^4 F^{(4)} \end{array} \right\} \\ &+ A^3 \Delta t^4 \left\{ \left( \rho_0^4 C_5 - \rho_1^4 C_4 + \rho_2^4 C_3 - \rho_3^4 C_2 \right) \Delta t F^{(1)} + \left( \rho_0^4 C_7 - \rho_1^4 C_6 + \rho_2^4 C_5 - \rho_3^4 C_4 \right) \Delta t^3 F^{(3)} \right\} \\ &+ A^4 \Delta t^5 \left\{ \begin{array}{l} \underbrace{\left( \rho_0^4 C_5 - \rho_1^4 C_4 + \rho_2^4 C_3 - \rho_3^4 C_2 + \rho_4^4 C_1 \right)}_{=\zeta_5^4 = 0} F \\ + \underbrace{\left( \rho_0^4 C_7 - \rho_1^4 C_6 + \rho_2^4 C_5 - \rho_3^4 C_4 + \rho_4^4 C_3 \right)}_{=\zeta_7^4 = 0} \Delta t^2 F^{(2)} \end{array} \right\} \\ &+ A^5 \Delta t^6 \underbrace{\left( \rho_0^4 C_7 - \rho_1^4 C_6 + \rho_2^4 C_5 - \rho_3^4 C_4 + \rho_4^4 C_3 \right)}_{=\zeta_7^4 = 0} \Delta t F^{(1)} \\ &+ A^6 \Delta t^7 \underbrace{\left( \rho_0^4 C_7 - \rho_1^4 C_6 + \rho_2^4 C_5 - \rho_3^4 C_4 + \rho_4^4 C_3 \right)}_{=\zeta_7^4 = 0} F + O(\Delta t^9).\end{aligned}$$

We finally have

$$\begin{aligned}\phi = \Delta t \left( F + \frac{1}{24} \Delta t^2 F^{(2)} + \frac{1}{1920} \Delta t^4 F^{(4)} + \frac{1}{322560} \Delta t^6 F^{(6)} \right) \\ - A\Delta t^2 \left( \frac{1}{12} \Delta t F^{(1)} + \frac{1}{480} \Delta t^3 F^{(3)} + \frac{1}{53760} \Delta t^5 F^{(5)} \right) \\ + A^2 \Delta t^3 \left( \frac{1}{36} F + \frac{1}{210} \Delta t^2 F^{(2)} + \frac{1}{26880} \Delta t^4 F^{(4)} \right) \\ - A^3 \Delta t^4 \left( \frac{1}{2940} \Delta t F^{(1)} + \frac{1}{40320} \Delta t^3 F^{(3)} \right) + O(\Delta t^9).\end{aligned} \tag{5.44}$$

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

### 5.2.3 *Numerical approximation of the RHS $\phi$*

We consider the following approximation:

$$\sum_{i=0}^{2m-r} \alpha_i^r \Delta t^i F^{(i)} \approx \sum_{i=0}^{n_w-1} \omega_i^r F\left(t_n + \Delta t\, c_i\right), \tag{5.45}$$

where $c_i$ are given points chosen in $[0,1]$ and $\omega_i^r$ 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

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

(5.45) can be written as follows

$$\sum_{i=0}^{2m-r} \alpha_i^r \Delta t^i F^{(i)} \approx \sum_{j=0}^{2m-1} \underbrace{\sum_{i=0}^{n_w-1} \omega_i^r \frac{\left(c_i - \frac{1}{2}\right)^j}{j!}}_{=\alpha_j^r} \Delta t^j F^{(j)}. \tag{5.46}$$

We identify $\Delta t^i F^{(i)}$ in (5.46) and deduce

$$\sum_{j=0}^{n_w-1} \omega_j^r \frac{\left(c_j - \frac{1}{2}\right)^i}{i!} = b_i = \begin{cases} \alpha_i^r, & \text{if } i \leq 2m-r \\ 0 & \text{otherwise} \end{cases} \tag{5.47}$$

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 $\omega_i^r$ by solving the linear system

$$\mathrm{VDM}\ \omega^r = b, \tag{5.48}$$

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

$$n_w = m$$

Finally we replace $c_i$ and $\omega_i^r$ in (5.45) to approximate $\phi$ in (5.41). $\phi_n$ 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 $\phi_n$ based on Gauss-Legendre points are strictly equivalent to Gauss-Runge-Kutta schemes (see the next section). 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** The presence of the source does not imply any additional matrix-vector product with the matrix $C = \Delta t A$. Indeed, the computation of $\phi_n$ 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.

## 5.3 Common features Gauss-RK and diagonal Padé schemes

As shown in the Chapter 4, the Gauss-RK method and the diagonal Padé schemes have the same stability function. We have observed that they are equivalent for linear systems if we use the Gauss quadrature node when we approximate the source function as mentioned in the remark of the previous section. To show this statement, let's consider the example of the fourth order Gauss-RK schemes:

$$\begin{array}{c|cc} c_1 & a_{11} & a_{12} \\ c_2 & a_{21} & a_{22} \\ \hline & b_1 & b_2 \end{array} = \begin{array}{c|cc} \frac{1}{2} - \frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}. \tag{5.49}$$

The solution of $y' = Ay + f$ at one step using (5.49) is computed as follows

$$k_1 = Ay_n + a_{11}\Delta t A k_1 + a_{12}\Delta t A k_2 + f_1 \tag{5.50}$$
$$k_2 = Ay_n + a_{21}\Delta t A k_1 + a_{22}\Delta t A k_2 + f_2 \tag{5.51}$$
$$y_{n+1} = y_n + \Delta t(b_1 k_1 + b_2 k_2). \tag{5.52}$$

We first need to find $k_1$ and $k_2$ by solving (5.50) and (5.51):

$$\underbrace{\begin{pmatrix} I - a_{11}\Delta t A & -a_{12}\Delta t A \\ -a_{21}\Delta t A & I - a_{22}\Delta t A \end{pmatrix}}_{:=B} \begin{pmatrix} k_1 \\ k_2 \end{pmatrix} = \begin{pmatrix} Ay_n + f_1 \\ Ay_n + f_2 \end{pmatrix} \tag{5.53}$$

In the general case (with $s$ steps), we have

$$\underbrace{\begin{pmatrix} I - a_{11}\Delta t A & -a_{12}\Delta t A & \cdots & -a_{1,m}\Delta t A \\ -a_{21}\Delta t A & I - a_{22}\Delta t A & \ddots & -a_{2,m}\Delta t A \\ \vdots & & \ddots & \vdots \\ -a_{m,1}\Delta t A & -a_{m,2}\Delta t A & \cdots & I - a_{m,m}\Delta t A \end{pmatrix}}_{:=B} \begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_m \end{pmatrix} = \begin{pmatrix} Ay_n + f_1 \\ Ay_n + f_2 \\ \vdots \\ Ay_n + f_m \end{pmatrix} = F \tag{5.54}$$

where $f_i = f(t_n + c_i\Delta t)$. The inverse of the matrix $B$ can be computed using the formula

$$B^{-1} = \frac{1}{\mathbf{det}(B)} \times (\mathbf{com}(B))^t.$$

The vectors $k_i$ are solutions to the following linear system

$$det(B)k_i = \left[(\mathbf{com}(B))^t F\right]_i$$

The vector $y_n$ is therefore computed as:

$$det(B)(y_{n+1} - y_n) = \sum_{i=1}^{m} b_i \left[(\mathbf{com}(B))^t F\right]_i$$

By expanding the right hand side of this equation, we will find the quantity $\phi_n$ we have introduced for Padé schemes if Gauss-Legendre points are chosen. The left hand side of this equation is a polynomial equation in $A$. Using the fourth order Gauss-RK for example, the determinant of the matrix $B$ is the following polynomial matrix

$$\begin{aligned} \mathbf{det}(B) &= I - (a_{11} + a_{22})\Delta t A + (a_{11}a_{22} - a_{12}a_{21})\Delta t^2 A^2 \\ &= I - \frac{1}{2}\Delta t A + \frac{1}{12}\Delta t^2 A^2 = D_m(\Delta t A), \text{ with } m = 2. \end{aligned} \tag{5.55}$$

We can see that we have the same matrix to invert for both the fourth order Gauss RK and the fourth order diagonal Padé scheme. This is also true for other values of $m$. For the right hand side, we have

$$com(B)^t = \begin{pmatrix} I - a_{22}\Delta t A & a_{12}\Delta t A \\ a_{21}\Delta t A & I - a_{11}\Delta t A \end{pmatrix}$$

We then find that

$$com(B)^t F = \begin{vmatrix} Ay_n + f_1 + (a_{12} - a_{22})\Delta t A^2 y_n + \Delta t A\left(a_{12}f_2 - a_{22}f_1\right) \\ Ay_n + f_2 + (a_{21} - a_{11})\Delta t A^2 y_n + \Delta t A\left(a_{21}f_1 - a_{11}f_2\right) \end{vmatrix}$$

We infer that

$$det(B)(y_{n+1} - y_n) = \Delta t A y_n + \frac{\Delta t}{2}(f_1 + f_2) + \frac{\sqrt{3}}{12}\Delta t^2 A\left(f_1 - f_2\right)$$

We have recovered the same expression as for Padé schemes:

$$D_m(\Delta t A)(y_{n+1} - y_n) = (N_m(\Delta t A) - D_m(\Delta t A))y_n + \phi_n$$

The procedure to find $\phi_n$ is easier to implement for Padé schemes since we provide the expressions of the right hand side $b$ and the VDM system to solve to obtain the coefficients $\omega_i^r$. In the case of Gauss Runge-Kutta schemes, the comatrix of B has to be computed and the terms have to be combined carefully to find the coefficients $\omega_i^r$. We have observed that Padé schemes and Gauss schemes are equivalent in the inhomogeneous case for any value of $m$. So, the two schemes are equivalent if and only if we use Gauss-Legendre points to approximate the source function in the diagonal Padé schemes.

## 5.4 Numerical results

This section provides numerical results in order to validate the diagonal Padé schemes implemented and to evaluate the efficiency of the schemes. The diagonal Padé schemes have been implemented in Montjoie, a $C$++ code.

We recall the first acoustic wave equation we are solving. 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:

$$\begin{cases} \rho\, \partial_t u - \operatorname{div} v = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ \mu^{-1} \partial_t v - \nabla u = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ u(x,0) = \partial_t u(x,0) = 0, \quad \forall x \in \Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\ \mu \partial_n u = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu \partial_n u + \sqrt{\rho\mu}\, \partial_t u = f_A, \quad x \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{5.56}$$

where $\Omega$ is the computational domain. $\Gamma_D$, $\Gamma_N$ and $\Gamma_A$ are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. $n$ is the outgoing normal vector defined for the considered boundary, $\rho$ and $\mu$ are physical indexes, which are piecewise constant. $f_D$, $f_N$ and $f_A$ are given source functions. The details of the space discretization of this equation are presented in the Chapter 3.

### 5.4.1 Convergence curves and numerical results in 1D

The wave equation (5.56) 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 this test case we use mixed spectral elements of order 16 for the space discretization. The computational domain $\Omega$ is subdivided into 500 regular sub-intervals. 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 5.2, we present the relative $L^2$ error between the exact solution and the numerical solution (obtained with diagonal Padé schemes of order 4, 6, 8 and 10) at $t = 200$. In the $x-$coordinate of the graph on the right we have chosen to represent $\frac{\Delta t}{m}$ where $\Delta t$ 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 5.2). 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 5.2) show that increasing the order makes the diagonal Padé schemes more efficient. These curves confirm the results we have obtained for the dispersion and dissipation curves as shown in the Figure 5.1.

Now we would like to evaluate the efficiency of the schemes regarding the computational times. We target 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 diagonal Padé of different order in Table 5.1 to reach this level of error. The results we obtained confirm also that by increasing the order the diagonal Padé schemes become more efficient in 1-D.

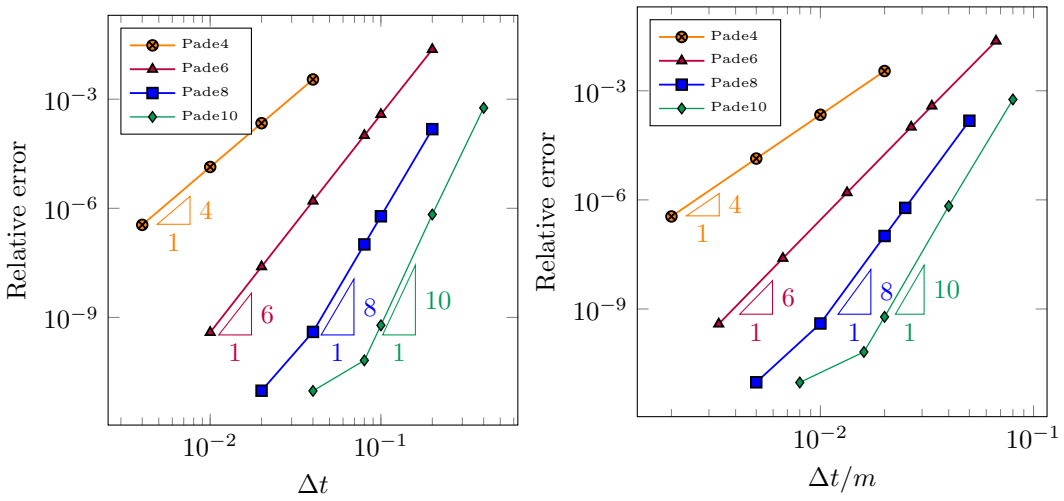



Figure 5.2: Relative $L^2$ error between numerical solution and exact solution for $t = 200$ versus the time step. Comparison of diagonal Padé schemes of order 4, 6, 8 and 10. On the right the time step is divided by the number of linear systems to be solved for each scheme.The space discretization error is about $10^{-12}$.

| | Pade4 | Pade6 | Pade8 | Pade10 |
|---|---|---|---|---|
| Number of time-steps | 33333 | 8360 | 3875 | 2326 |
| Computational time | 1mn36s | 37s | 24s | 17s |

| | DIRK34 | SDIRK3 | SDIRK55 |
|---|---|---|---|
| Number of time steps | 87000 | 110000 | 23760 |
| Computational time | 5mn08s | 6mn29s | 2mn23s |

Table 5.1: Computational time after imposing $1\%$ of relative errors (1-D case).

### *5.4.2 Computational results on a 2D regular square mesh*

In this part we present numerical results we have obtained when using Padé schemes to solve one simple 2D problem. We have used a tenth order finite elements method to approximate the acoustic wave equation (5.56) on a regular square mesh $\Omega = [-4, 4]^2$. The mesh is constructed by chosing 10 points along both the $x$ and $y$ axes. We have set a homogeneous Neumann conditions on the boundaries. In the midle of the domain, we have considered a volumetric source given by

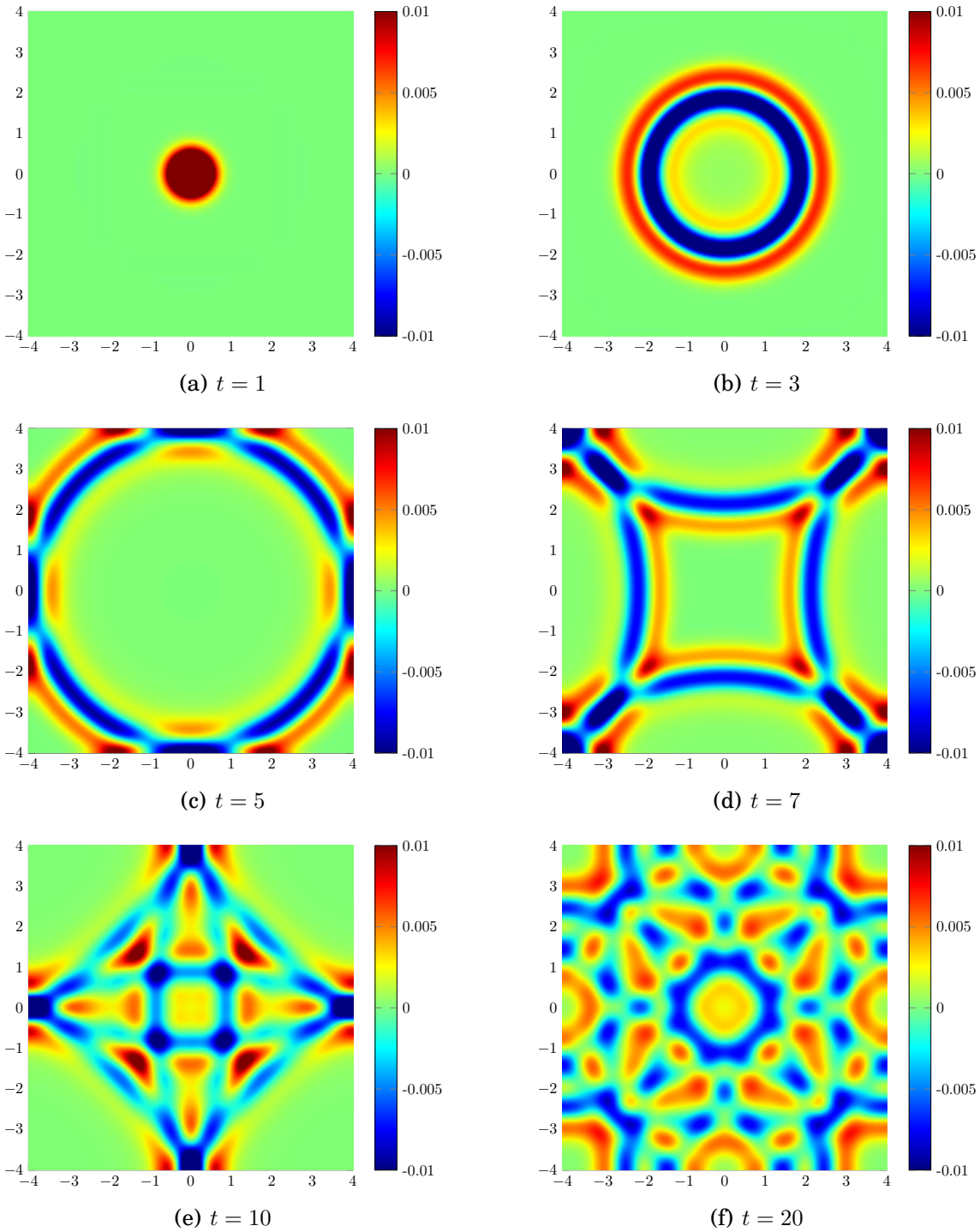
$$\int_{\Omega} f(x)\varphi_i dx,$$

where $f$ is the Gaussian function centered at the point $(0, 0)$ with the distribution radius equal to $1$.

For the time interval, we have chosen $[0, 20]$. The temporal source corresponds to the Ricker function (i.e. the second-derivative of the Gaussian function).

We have compared computational times of the two efficient SDIRK schemes (SDIRK34 and SDIRK55) with that of the diagonal Padé schemes of order 2, 4, 6 and 8, after imposing $0.1\%$ of errors. The reference solution has been computed using the eighth-order Padé scheme for $\Delta t = 0.01$ (2000 time-steps). The obtained results are presented in the Table 5.2. The column Memory (MiB) represents the memory consumption in megabytes during the overall simulation. For comparison, the number of time-steps are also chosen such that the solution can be written at exactly $t = 20$. For this reason, we observe in the Table 5.2 that the relative error is less than $0.1\%$ for some schemes. Although this fact impacts the computational time, we can observe that Padé schemes are more efficient compared to the SDIRK. But we observe also that Padé schemes require much more memory compared to the SDIRK schemes. This is due to the fact SDIRK schemes requires the solution of a linear system which is factorized once for all iterations (see the Chapter 4). Only the right hand side changes at each iteration. On the contrary for Padé schemes, their $m$ different system to be factorized. In the Figure 5.3, we show the evolution of the numerical solution at different time-steps.

These preliminaries results demonstrate a clear advantage in terms of computational time for Diagonal Padé schemes compared to RK Schemes even though they require more memory. To confirm the efficiency obtained, we perform further comparisation of simulations for 2D and 3D cases in the next chapters. Furthermore, we have seen in the Chapter 4 that classical Runge-Kutta schemes can be used to solve non linear problems which is not the case for the Padé schemes developed in this document. To have a fair comparison, we design in the Chapter 6 Linear-SDIRK time schemes having the same properties as SDIRK schemes but for linear ODE only.




(a) $t = 1$ (b) $t = 3$

(c) $t = 5$ (d) $t = 7$

(e) $t = 10$ (f) $t = 20$

Figure 5.3: Solution obtained for the scattering in a square mesh at $t = 1, 3, 5, 7, 10$ and the final time $t = 20$.

| Schemes | Number of time steps | Relative errors | Computational time | Memory (MiB) |
|---|---|---|---|---|
| Pade2 | 10840 | 0.001034 | 123.68s | 29.6 |
| Pade4 | 480 | 0.000954 | 11.49s | 45.86 |
| Pade6 | 160 | 0.000704 | 8.21s | 62.99 |
| Pade8 | 80 | 0.001031 | 6.71s | 79.26 |
| Pade10 | 60 | 0.000326 | 6.74s | 96.39 |
| SDIRK34 | 1560 | 0.000997 | 47.92s | 28.28 |
| SDIRK55 | 380 | 0.001079 | 20.75s | 28.81 |

Table 5.2: Computational time after imposing $0.1\%$ of relative errors at the final time $t = 20$, Padé versus SDIRK34 and SDIRK55 (2D case).

Chapter 6

# A-stable high-order Singly Diagonally Runge-Kutta (SDIRK) schemes for linear ODEs

In this chapter we construct a new familly of A-stable SDIRK schemes. The developed schemes are called Linear-SDIRK since they share the same properties as classical SDIRK but for linear ODEs only. We explain how to implement them and compare their performances with diagonal Padé schemes. The assessement is done regarding dissipation, dispersion and computational time.

## Contents

## 6.1 Introduction

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 development of implicit Runge-Kutta schemes using first the Gauss quadrature formula to get order $2s$ when using scheme of $s$ stages. In the Chapter 4 we have seen that this kind of scheme requires the solution of a large linear system of size $sN$ ($s$ being the number of stages and $N$ being the number of unknowns) at each time step which makes this approach quite inefficient.

To reduce the computational burden, the idea is to construct implicit Runge-Kutta schemes in which we will have to solve the same linear system of size $N$ with different right hand sides at each time step. As introduced in the Chapter 4, this kind of method is called Singly Diagonally Implicit Runge-Kutta (SDIRK) method. From (4.42) we have seen that the stability function of SDIRK schemes has only one pole. But due to non-linear order conditions and the A-stability requirement, SDIRK schemes are not common in the litterature beyond order 4 (see the Section 4.5). Therefore, the main motivation of this chapter is to construct high-order time integration schemes for linear ODEs for which the stability function has only one pole. These new schemes that can be developed at any order of accuracy will be called Linear-SDIRK schemes. Contrary to Padé schemes that require the solution of different real/complex systems as detailed in see the Chapter 5, we shall see that Linear-SDIRK schemes require to solve a unique real linear system several times at each time step.

Unsurprisingly, in the reminder of this chapter, we consider the ODE

$$\begin{cases} M_h \dfrac{dX(t)}{dt} + K_h X(t) = F(t) \quad t \in (0, T] \\ X(0) = X_0 \end{cases} \tag{6.1}$$

obtained after spatial discretization, where $M_h$ is the mass matrix, $K_h$ is the stiffness matrix and $h$ 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. The analytical solution to (6.1) after one step $\Delta t = t_{n+1} - t_n$ is given by

$$X(t_{n+1}) = \mathrm{e}^{\Delta t A} \left( X(t_n) + \int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\, \mathrm{d}u \right), \tag{6.2}$$

where $A = -M_h^{-1} K_h$. The construction of the Linear-SDIRK schemes follows the same rules as in the Chapter 5. We consider the stability function in the form

$$R(z) = \frac{N(z)}{D(z)} \approx \mathrm{e}^z,$$

where $N(z)$ and $D(z)$ are polynomials of $z \in \mathbb{C}$. We note

$$\phi = N(\Delta t A) \int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u + O(\Delta t^{p+1}), \tag{6.3}$$

and the equation (6.2) becomes

$$D(\Delta t A) X(t_{n+1}) = N(\Delta t A) X(t_n) + \phi. \tag{6.4}$$

The numerical solution $X_{n+1}$, that approximates $X(t_{n+1})$, is obtained by solving the following linear system:

$$\boxed{D(\Delta t A) X_{n+1} = N(\Delta t A) X_n + \phi_n} \tag{6.5}$$

where $\phi_n$ is given by the following formula

$$\phi_n = \sum_{r=1}^{m} A^{r-1} \Delta t^r \sum_{i=0}^{n_w - 1} \omega_i^r \, F\left(t_n + \Delta t \, c_i\right). \tag{6.6}$$

In the equation (6.6) $m$ is the degree of the polynomial $N$ or $D$ and $n_w$ is a number that depends on the scheme. Typically, $n_w$ is written as a function of the order of convergence of the scheme.

## 6.2 Stability functions of Linear-SDIRK schemes

In this section we will consider a rational polynomial function

$$R(z) = \frac{N(z)}{D(z)},$$

which approximates the exponential function and minimizes the error. In order to have only one pole, the denominator of $R(z)$ is given by

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

where $\gamma$, the only pole of $R$, 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 $\gamma$ satisfying the following requirements

- the method is A-stable (see the Definition 2.15),
- the method 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 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 [14], one can try to reconstruct Runge-Kutta coefficients. For higher orders, there are too many 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

$$R_s^l(z) = 1 + z + \frac{z}{2!} + \cdots + \frac{z^s}{s!} + \frac{z^{s+1}}{(s+1)!} + \frac{\alpha_1 z^{s+2} + \cdots + \alpha_{s+l} z^{2s+l+1}}{(1-\gamma z)^{s+l}}. \tag{6.7}$$

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 (6.12), (6.14), (6.16) and (6.18) for respectively $l = 0$, $l = 1$, $l = 2$ and $l = 3$. In the following, we use $D_s^l$ as

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

### 6.2.1 *Linear-SDIRK methods with $s$ 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 obtained 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:

$$\gamma = \frac{1}{2}$$

$$\alpha_0 = \frac{\gamma}{2}.$$

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 $\alpha_1$, $\alpha_2$ and $\gamma$ 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 $\alpha_1$, $\alpha_2$ and $\gamma$ for the third order approximation. We get

$$\alpha_2 + \frac{\gamma^2}{6} = 0, \tag{6.8}$$

$$\alpha_1 - \frac{\gamma}{3} + \frac{\gamma^2}{2} = 0, \tag{6.9}$$

$$\frac{1}{6} - \gamma + \gamma^2 = 0. \tag{6.10}$$

We compute $\gamma$ as a solution to (6.10), and $\alpha_1$ and $\alpha_2$ are deduced from relations (6.8) and (6.9). 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 $\gamma$ are $\frac{1}{2} - \frac{1}{2\sqrt{3}}$ and $\frac{1}{2} + \frac{1}{2\sqrt{3}}$ (the roots of (6.10)). The one that leads to $A$-stable scheme is $\gamma = \frac{1}{2} + \frac{1}{2\sqrt{3}}$. In fact with this choice of $\gamma$ 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 [1]).

### Order 4

To construct a fourth order Linear-SDIRK scheme with $s = 3$ and $l = 0$, we must find $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\gamma$ 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 $\alpha_1$, $\alpha_2$, $\alpha_3$ from $\gamma$. The parameter $\gamma$ 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

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

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 $\alpha_1$, $\alpha_2$ and $\alpha_3$ are set to cancel the higher order terms):

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

We then have

$$R_3^0(z) = \frac{a_0(\gamma) + a_1(\gamma)z + a_2(\gamma)z^2 + a_3(\gamma)z^3}{(1-\gamma z)^3}.$$

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

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

This scheme has the same stability function as the SDIRK scheme of order 4 obtained by Crouzeix (see [1]).

### 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:

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

The constants $\alpha_i$ 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

$$a_{s+1}(\gamma) = \sum_{i=0}^{s} \frac{(-\gamma)^i \binom{s}{i}}{(s+1-i)!} = 0. \tag{6.11}$$

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

$$N_s^0(z) = a_0(\gamma) + a_1(\gamma) z + a_2(\gamma) z^2 + \cdots + a_s(\gamma) z^s. \tag{6.12}$$

We have

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

To ensure the A-stability condition we choose $\gamma$ as follows: for each $\gamma$ root of (6.11), 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:

$$|N(i\sqrt{z})|^2 = |D(i\sqrt{z})|^2. \tag{6.13}$$

We choose $\sqrt{z}$ instead of $z$ to obtain a polynomial equation of lower degree and easier to solve. Since $\gamma > 0$, $R_s^0$ is holomorphic in the negative half plane ($\mathbb{C}^-$). Then based on the maximum principle, $R_s^0$ reaches its maximal value on the imaginary axis or when $|z|$ tends to infinity. For these reasons it suffices to consider only the imaginary axis. Hence, if the polynomial equation (6.13) has no real roots except zero, then the scheme is A-stable, otherwise the scheme is not A-stable. We present in Table 6.1, the A-stable schemes we have obtained.

As presented in Table 6.1, 6 is the maximal order that we can achieve. In fact, for $s \geq 6$, it can be conjectured that there is no root $\gamma$ that leads to a A-stable scheme. Without extra stage ($l = 0$), we have retrieved the Linear-SDIRK schemes presented by Burrage [13]. To get higher-order schemes, we need to increase the number of stages. This will be addressed in the next subsections.

| s | $\gamma_{opt}$ | comment |
|---|---|---|
| 1 | 0.5 | - |
| 2 | 0.788675134594813 | - |
| 3 | 1.068579021301629 | - |
| 4 | x | No A-stable schemes |
| 5 | 0.473268391258295 | - |
| 6 | x | No A-stable schemes |
| $r \geq 8$ | x | No A-stable schemes |

Table 6.1: Minimal stage Linear-SDIRK of order **(s+1)** and associated value of $\gamma$.

### 6.2.2 *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, $\gamma$ is a free parameter. We note $P$ the polynomial that is involved while reducing to the common denominator:

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

The constants $\alpha_i$ 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

$$N_s^1(z) = a_0(\gamma) + a_1(\gamma) z + a_2(\gamma) z^2 + \cdots + a_{s+1}(\gamma) z^{s+1} \tag{6.14}$$

and we thus have

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

The optimization is done by minimizing $|\eta(\gamma)| = |a_{s+2}(\gamma) + \frac{1}{(s+2)!}|$ under the A-stability constraint:

$$\underset{\gamma, |R_s^1(z)| \leq 1, z \in \mathbb{C}^-}{\arg\min} |\eta(\gamma)|. \tag{6.15}$$

Since $\gamma$ is the only free parameter, this constraint imposes that $\gamma$ belongs to an interval or a set of intervals. The admissible intervals for $\gamma$, for which the

| s | Interval for $\gamma$ | $\gamma_{opt}$ |
|---|---|---|
| 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 |

Table 6.2: Linear-SDIRK of order **(s+1)** with one additional stage.

schemes are A-stable, are represented in Table 6.2. The optimal value of $\gamma$ that minimizes the error term $|\eta(\gamma)|$ is also provided for each $s$. This optimization has been perfomed using the simplex algorithm. We can observe that the optimum coincides with the left extremity of the stability interval for $\gamma$. We have obtained the same admissible intervals as Burrage (see [13]).

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.

### 6.2.3 *Linear-SDIRK methods $(s+2)$-stages of order $(s+1)$*

To obtain higher-order Linear-SDIRK schemes 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 $\gamma$ and $\alpha_1$. Now let $P$ be the polynomial that appears while reducing to the common denominator:

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

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

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

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

$$N_s^2(z) = a_0(\gamma) + a_1(\gamma) z + a_2(\gamma) z^2 + \cdots + a_{s+1}(\gamma) z^{s+1} + (a_{s+2}(\gamma) + \alpha_1) z^{s+2}. \quad (6.16)$$

| s | $\gamma_{opt}$ | $\alpha_{1_{opt}}$ | comment |
|---|---|---|---|
| $1 \leq s \leq 4$ | - | - | No uniqueness |
| 5 | 0.204071 | $1.9839430662 \cdot 10^{-4}$ | - |
| 6 | - | - | leads to eighth order |
| 7 | 0.166890 | $2.9259251764 \cdot 10^{-6}$ | - |
| 8 | x | x | No A-stable schemes |
| 9 | 0.141940 | $2.2982637210 \cdot 10^{-8}$ | - |
| $r \geq 10$ | x | x | No A-stable schemes |

Table 6.3: Linear-SDIRK of order **(s+1)** with two additional stages

We have

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

The optimization is done by minimizing $|\eta(\gamma)| = |\alpha_1 - \frac{1}{(s+2)!}|$ under the A-stability constraint:

$$\underset{\gamma,\, \alpha_1, |R_s^2(z)| \leq 1,\, z \in \mathbb{C}^-}{\arg\min} |\eta(\gamma)| \tag{6.17}$$

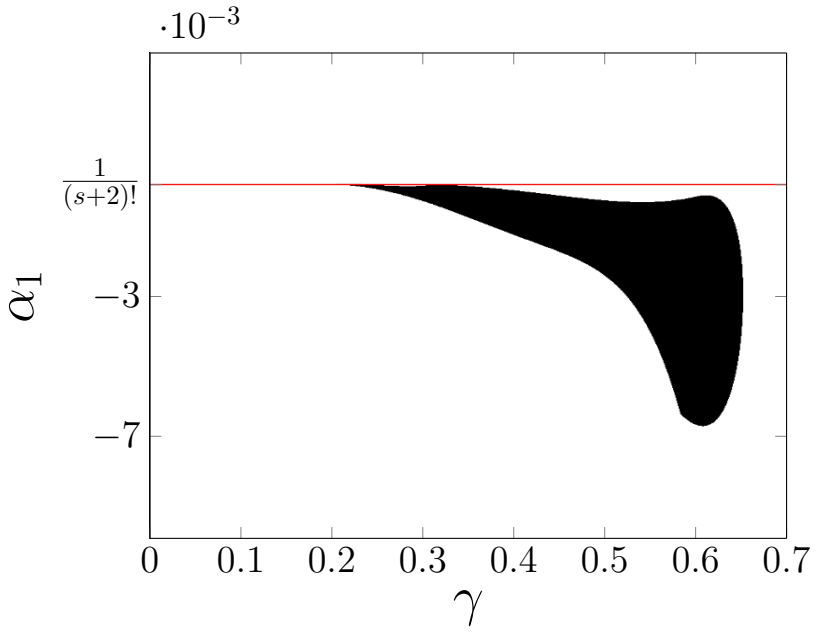
Since we have two parameters, this constraint will imposes that $\gamma$ and $\alpha_1$ belong to 2-D regions. We have first identified the 2-D admissible regions for ($\gamma$, $\alpha_1$) where the schemes are A-stable. Then, we have zoomed on the minimum value. The stability region for the two parameters is colored in black in the Figures 6.1, 6.2 and 6.3. The optimal value that minimizes the error $\eta$ is found when $\alpha_1$ is closer to the line $y = \frac{1}{(s+2)!}$, $s = 5, 6, 7$ in the different figures.

The values we have chosen for $\gamma$ and $\alpha_1$ are presented in Table 6.3. The presented values are not exactly the optimal values, since the optimal values lie in the boundary of the admissible regions. In order to have robust methods, we prefered to choose nearly optimal values that are strictly inside the admissible regions. 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.

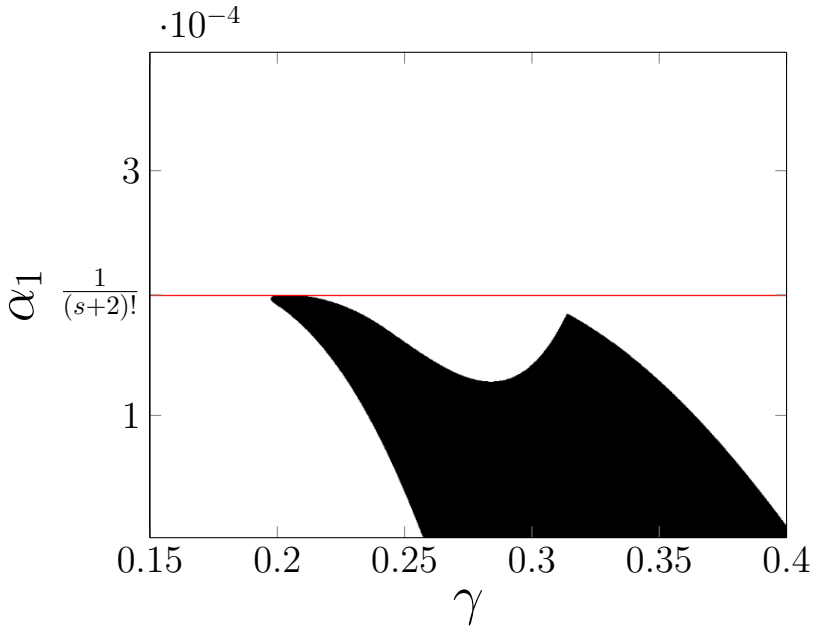
### 6.2.4 *Linear-SDIRK methods $(s+3)$-stages of order $(s+1)$*

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:

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




(a) Domain for the two parameters




(b) Zoom next to optimum

Figure 6.1: Admissible stability region for $\gamma$ and $\alpha_1$ in the construction of Linear-SDIRK schemes of order $s+1=6$ with 2 additional stages.

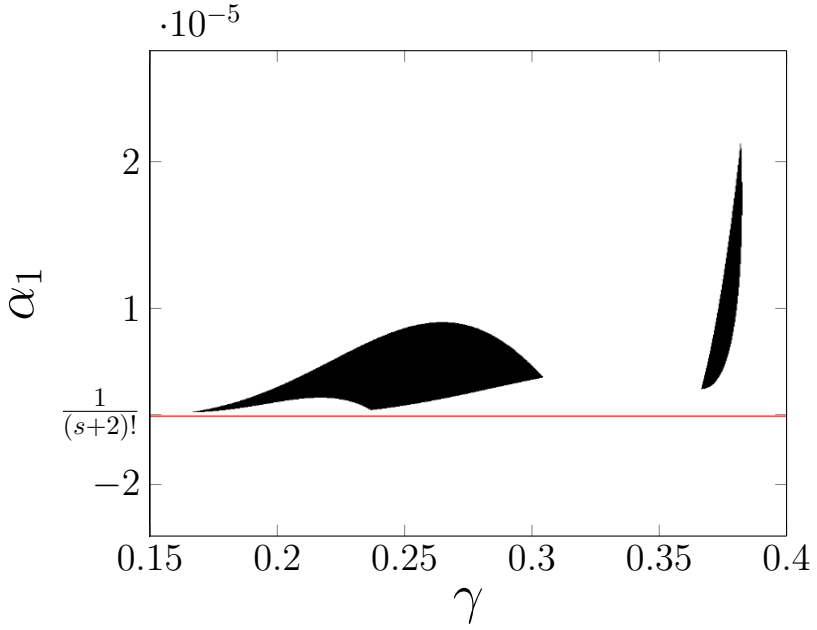



Figure 6.2: Admissible stability region for $\gamma$ and $\alpha_1$ in the construction of Linear-SDIRK schemes of order $s+1=8$ with 2 additional stages.

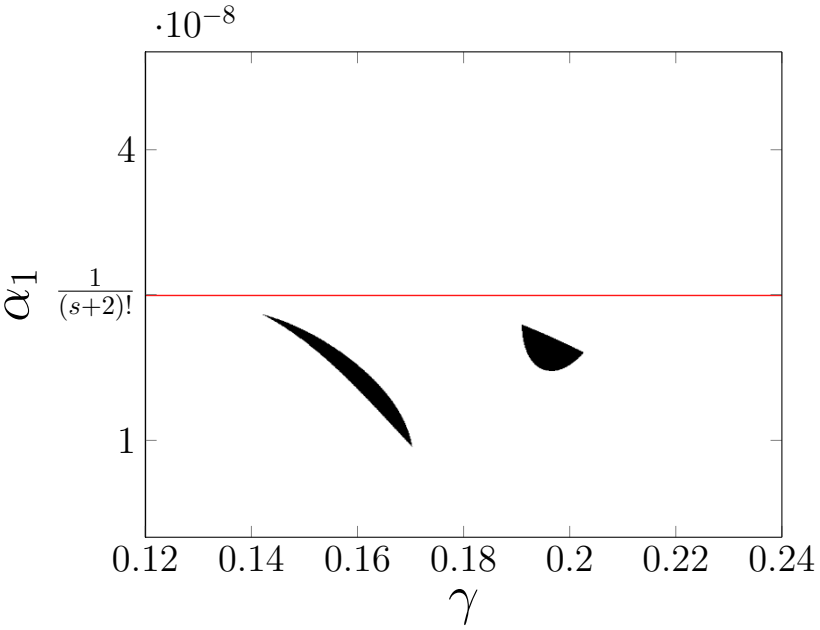



Figure 6.3: Admissible stability region for $\gamma$ and $\alpha_1$ in the construction of Linear-SDIRK schemes of order $s+1=10$ with 2 additional stages.

| s | $\gamma_{opt}$ | $\alpha_{1_{opt}}$ | $\alpha_{2_{opt}}$ |
|---|---|---|---|
| $1 \leq s \leq 6$ | - | - | No uniqueness |
| 7 | 0.136339 | $2.767416226 \cdot 10^{-06}$ | $-3.464398093 \cdot 10^{-06}$ |
| 8 | - | - | leads to tenth order |
| 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 |

Table 6.4: Linear-SDIRK of order **(s+1)** with three additional stages

We have three free parameters $\gamma$, $\alpha_1$ and $\alpha_2$. Like in previous subsections, we note $P$ the polynomial that appears while reducing to the common denominator:

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

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

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

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

$$\begin{aligned} N_s^3(z) = a_0(\gamma) + a_1(\gamma)z + a_2(\gamma)z^2 + \cdots + a_{s+1}(\gamma)z^{s+1} \\ + (a_{s+2}(\gamma) + \alpha_1)z^{s+2} + (a_{s+3}(\gamma) + \alpha_2)z^{s+3}. \end{aligned} \tag{6.18}$$

The stability function reads:

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

The optimization is done by minimizing $|\eta(\gamma)| = |\alpha_1 - \frac{1}{(s+2)!}|$ under the A-stability constraint:

$$\underset{\gamma, \alpha_1, \alpha_2, |R_s^3(z)| \leq 1, z \in \mathbb{C}^-}{\arg\min} |\eta(\gamma)|. \tag{6.19}$$

With three parameters, the constraint imposes that $\gamma$, $\alpha_1$ and $\alpha_2$ belong to a 3-D region. Like in the previous subsection with two parameters, we have proceeded by identifying the region for the free parameters where the schemes is A-stable. This is done by finding the region for $\gamma$ first then searching for a 2D region for $\alpha_1$ and $\alpha_2$. In the Figures 6.4, 6.5 and 6.6 we show the shape of the stability region for the free parameters. Knowing the stability region for the three parameter, we have chosen the coefficients that minimize the error term $\eta(\gamma)$. The optimal

Figure 6.4: Admissible stability region for $\gamma$, $\alpha_1$ and $\alpha_2$ in the construction of Linear-SDIRK schemes of order $s+1=8$ with 3 additional stages.

Figure 6.5: Admissible stability region for $\gamma$, $\alpha_1$ and $\alpha_2$ in the construction of Linear-SDIRK schemes of order $s+1=10$ with 3 additional stages.

values obtained for $\gamma$, $\alpha_1$ and $\alpha_2$ are presented in Table 6.4. Here we have been able to obtain an A-stable scheme of order twelve (versus 10 in the previous sub-section). For $1 \leq s \leq 6$, we did not find a unique optimal choice for the three free parameters.

**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 is no 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 for 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 6.3 and 6.4.

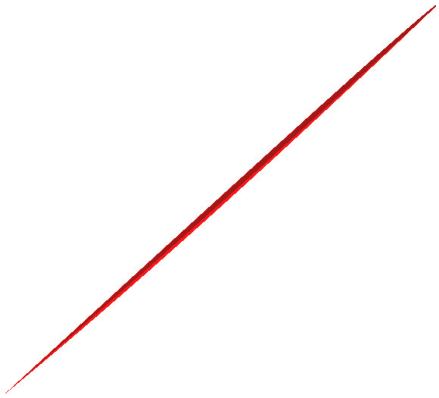


Figure 6.6: Admissible stability region for $\gamma$, $\alpha_1$ and $\alpha_2$ in the construction of Linear-SDIRK schemes of order $s+1=12$ with 3 additional stages.

To summarize, we have provided optimal coefficients to construct the stability function of A-stable Linear-SDIRK schemes up to order 12. These stability functions are obtained by replacing the different coefficients into the equation (6.7). We note that the values obtained for $\gamma$ are rather small. For this reason the matrix $(I-\gamma\Delta tA)$, that wil be involved in the linear system to be solved (see 6.4.2), will be well-conditioned.

## 6.3 Numerical stability, dissipation and dispersion

We recall the test equation we use to analyse the numerical stability, dissipation and dispersion:

$$y' = i\lambda y, \quad y(t_0) = y_0 \text{ and } \lambda \in \mathbb{R}. \tag{6.20}$$

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 (6.20) and have the same dispersion error. In Figures 6.7, 6.8, 6.9 and 6.10, 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. The abscissa is $\frac{z}{m}$ where $m$ represents the computational complexity of the scheme (see the Section 5.2 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.

**Remark** In the figure 6.7, the diagonal Padé scheme of order 4 (Pade4), which

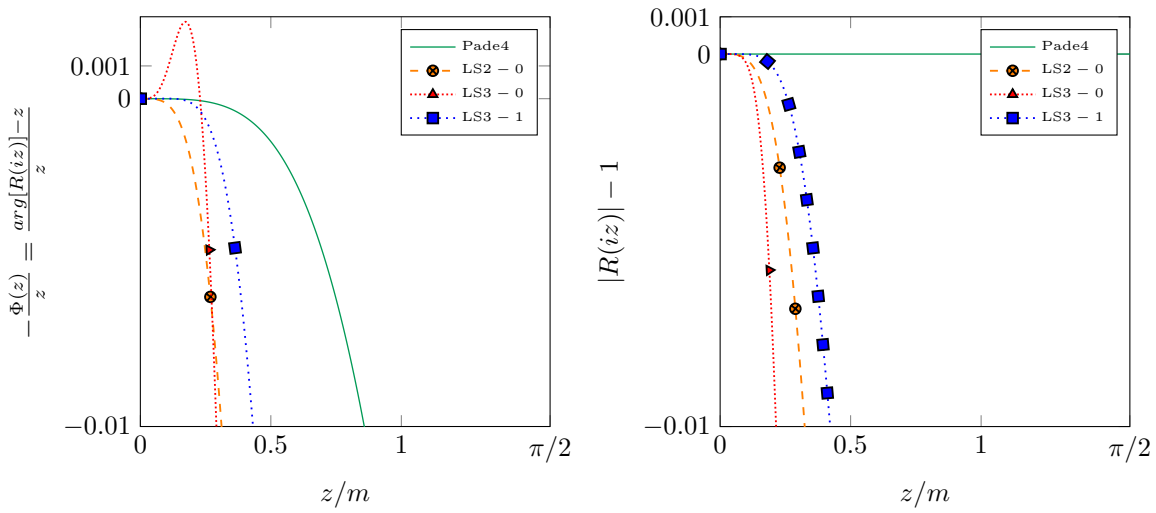



Figure 6.7: Dispersion and dissipation curves of diagonal Padé schemes of order 4 compared to that of the Linear-SDIRK, when applied to the test equation (6.20). LS$2-l$ and LS$3-l$ represents the $s=2$ and $s=3$ plus $l$ additional stages Linear-SDIRK of order 3 and 4.

is not dissipative, is less dispersive than the Linear-SDIRK schemes. Among the Linear-SDIRK schemes of order 4, the less dispersive and the less dissipative scheme is the one obtained for $s=3$ and $l=1$ (LS$3-1$).

**Remark** In the Figure 6.8, the diagonal Padé scheme of order 6 (Pade6), which is not dissipative, is less dispersive compared to the Linear-SDIRK schemes of the same order. Among the Linear-SDIRK schemes of order 6, the less dispersive and the less dissipative scheme is the one obtained for $s=5$ and $l=2$ (LS$5-2$).

**Remark** In Figure 6.9, the diagonal Padé scheme of order 8 (Pade8), which is not dissipative, is less dispersive than any of the Linear-SDIRK schemes of the same order. Among the Linear-SDIRK schemes of order 8, the less dispersive and the less dissipative scheme is the one obtained for $s=7$ and $l=3$ (LS$7-3$).

## 6.4 Computation of the RHS term and algorithm

### *6.4.1 Computation of the right hand side (RHS)*

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

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

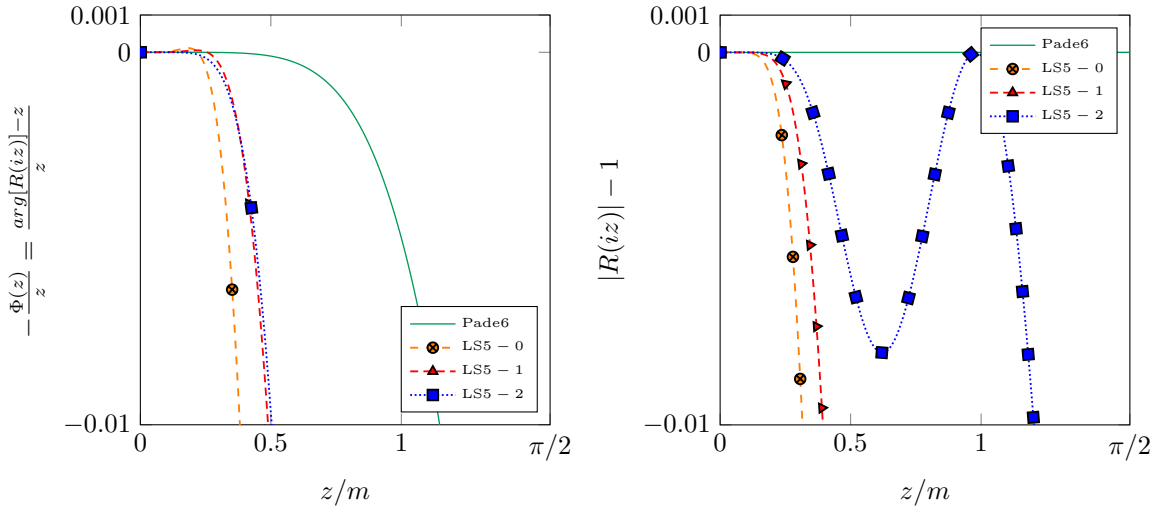



Figure 6.8: Dispersion and dissipation curves of diagonal Padé schemes of order 6 compared to that of the Linear-SDIRK, when applied to the test equation (6.20). LS$5-l$ represents the $s=5$ plus $l$ additional stages Linear-SDIRK of order 6.

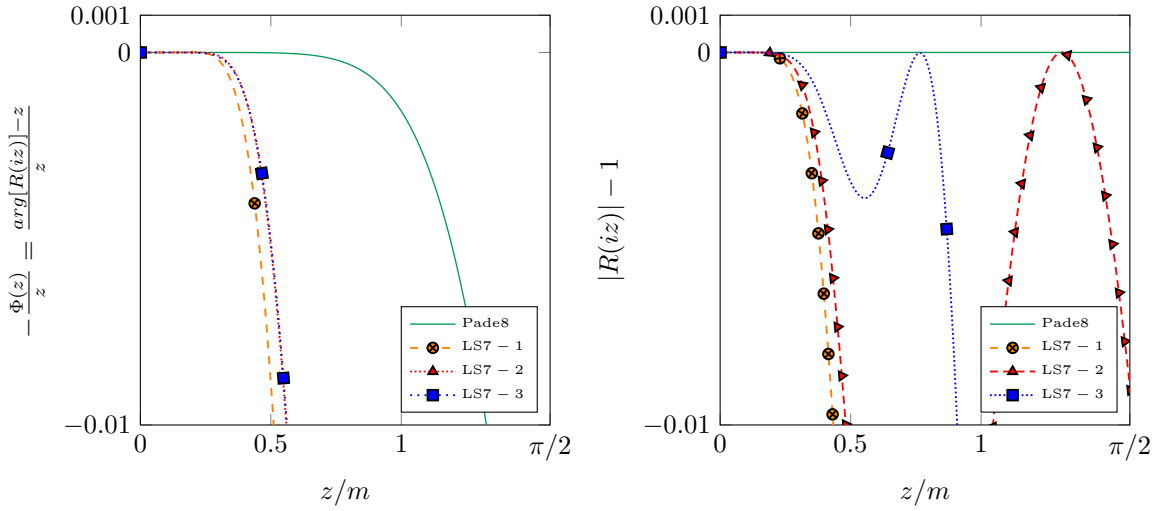



Figure 6.9: Dispersion and dissipation curves of diagonal Padé schemes of order 8 compared to that of the Linear-SDIRK, when applied to the test equation (6.20). LS$7-l$ represents the $s=7$ plus $l$ additional stages Linear-SDIRK of order 8.

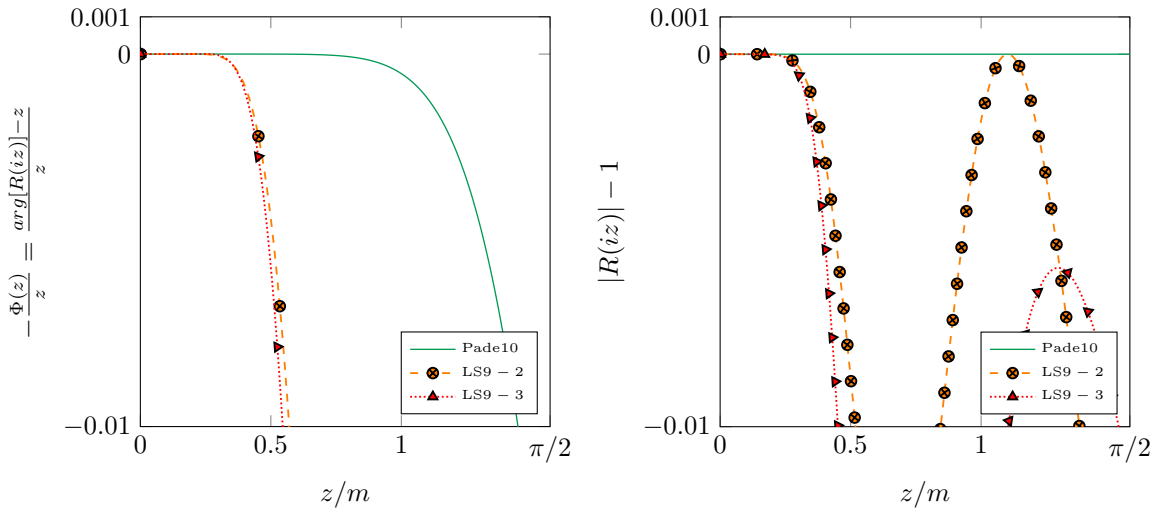



Figure 6.10: Dispersion and dissipation curves of diagonal Padé schemes of order 10 compared to that of the Linear-SDIRK, when applied to the test equation (6.20). LS$9-l$ represents the $s = 9$ plus $l$ additional stages Linear-SDIRK of order 10.

with $N_s^l(z)$ and $D_s^l(z)$ defined in the previous subsections. We derive the expression of $\phi$ from the equation (6.5) to obtain

$$\phi = D_s^l(\Delta tA)X(t_{n+1}) - N_s^l(\Delta tA)X(t_n) + O(\Delta t^{s+2}). \tag{6.21}$$

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

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

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

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

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

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

where

$$\alpha_{j-1}^{r,l} = \sum_{i=0}^{\min(r-1,s+l)}\frac{1}{2^{r+j-i-1}(r+j-i-1)!}\left(D_i-(-1)^{r+j-i-1}N_i\right). \tag{6.22}$$

We can notice that

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

because $\gamma$ solves Equation (6.11). 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

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

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 of taking the nominal values for $(\alpha_i^{r,l})_{i>s+1-r}$ (as defined by equation (6.22)) 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 5.2.3 for Padé schemes. $\phi_n$ is therefore computed by using the formula

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

### 6.4.2 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 $\Delta tA$ 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. This instability is due to round-off errors and the Hörner's algorithm when we deal with polynomials of higher degrees.

When solving a homogeneous ODE (there is no source term), a stable algorithm consists of 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 an accurate solution. We have detailed in

**Algorithm 4** Stable algorithm without source term for Linear-SDIRK schemes

$y = X_n$
**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**
$X_{n+1} = y$

Algorithm 4 the stable algorithm 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 the source term is added, we rewrite $\phi_n$ in the following form:

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

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

$$Q_0(z) = \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_2 z+a_2 z^2\right) z$$

$$Q_1(z) = \left(1-\frac{z}{\lambda_{n_r}}\right)\left(1-\frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_2 z+a_2 z^2\right)$$

$$Q_2(z) = \left(1-\frac{z}{\lambda_{n_r}}\right)\cdots\left(1-\frac{z}{\lambda_{n_1}}\right)\left(1-b_{n_c}z+a_{n_c}z^2\right)\cdots\left(1-b_3 z+a_3 z^2\right)(1-\gamma z)^2 z$$

$$Q_3(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right) \cdots \left(1 - \frac{z}{\lambda_{n_1}}\right) \left(1 - b_{n_c} z + a_{n_c} z^2\right) \cdots \left(1 - b_3 z + a_3 z^2\right) (1 - \gamma z)^2$$

$$\cdots$$

$$Q_{s+l-2}(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right) (1 - \gamma z)^{s+l-1}$$

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

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 5. Note that we have proposed an efficient algorithm to solve a second order polynomial having two complex-conjugate roots in the Section 5.2 of the Chapter 5.

**Algorithm 5** Stable algorithm with source term for Linear-SDIRK schemes

$y = X_n$
**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**
$X_{n+1} = y$

We recall that the values we have obtained for $\gamma$ in the Section 6.2 are small ($\gamma < 1$). As a result the matrix $(I - \gamma \Delta t A)$, involved in the linear system to be solved at each time step (see the Algorithms 4 and 5), is well-conditioned.

The Algortims 4 and 5 can be applied to the Padé scheme developed in the Chapter 5 as well. In fact for Padé schemes of order beyond 12, we have observed the same instability.

## 6.5 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:

$$\begin{cases} \rho\,\partial_t u - \mathbf{div}\, v = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ \mu^{-1}\partial_t v - \nabla u = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ u(x,0) = \partial_t u(x,0) = 0, \quad \forall x \in \Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\ \mu \partial_n u = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu \partial_n u + \sqrt{\rho\mu}\;\partial_t u = f_A, \quad x \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{6.23}$$

where $\Omega$ is the computational domain. $\Gamma_D$, $\Gamma_N$ and $\Gamma_A$ are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. $n$ is the outgoing normal to the considered boundary, $\rho$ and $\mu$ are physical indexes, which are piecewise constant. $f_D$, $f_N$ and $f_A$ are given source functions.

### *6.5.1 Convergence curves and numerical results in 1-D*

The wave equation (6.23) 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 $\Omega$ is subdivided into 500 regular sub-intervals. 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)

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

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

| | Pade4 | LS3−0 | LS3−1 | Pade6 | LS5−0 | LS5−1 | LS5−2 |
|---|---|---|---|---|---|---|---|
| Number of time-steps | 33333 | 110000 | 25960 | 8360 | 18835 | 11540 | 7355 |
| Computational time | 1mn36s | 5mn23s | 1mn48s | 37s | 1mn31s | 1mn09s | 53s |

| | Pade8 | LS7−1 | LS7−2 | LS7−3 | Pade10 | LS9−2 | LS9−3 |
|---|---|---|---|---|---|---|---|
| Number of time-steps | 3875 | 5830 | 4600 | 3700 | 2326 | 3106 | 2845 |
| Computational time | 24s | 46s | 43s | 38s | 17s | 34s | 34s |

Table 6.5: Computational time after imposing $1\%$ of relative errors (1-D case).

In Figure 6.11, 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 $t = 200$. The abscissa is $\frac{\Delta t}{m}$ where $\Delta t$ is the time step and $m$ is the number of linear systems we need to solve at each time step (see the Section 5.2 of the Chapter 5 for Padé schemes and the Section 6.2 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 6.11) 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 6.7, 6.8 and 6.9. 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.

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. Our target is 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 6.5 to reach this error. The results we have obtained confirm that the diagonal Padé schemes are more efficient than the Linear-SDIRK schemes in 1D.

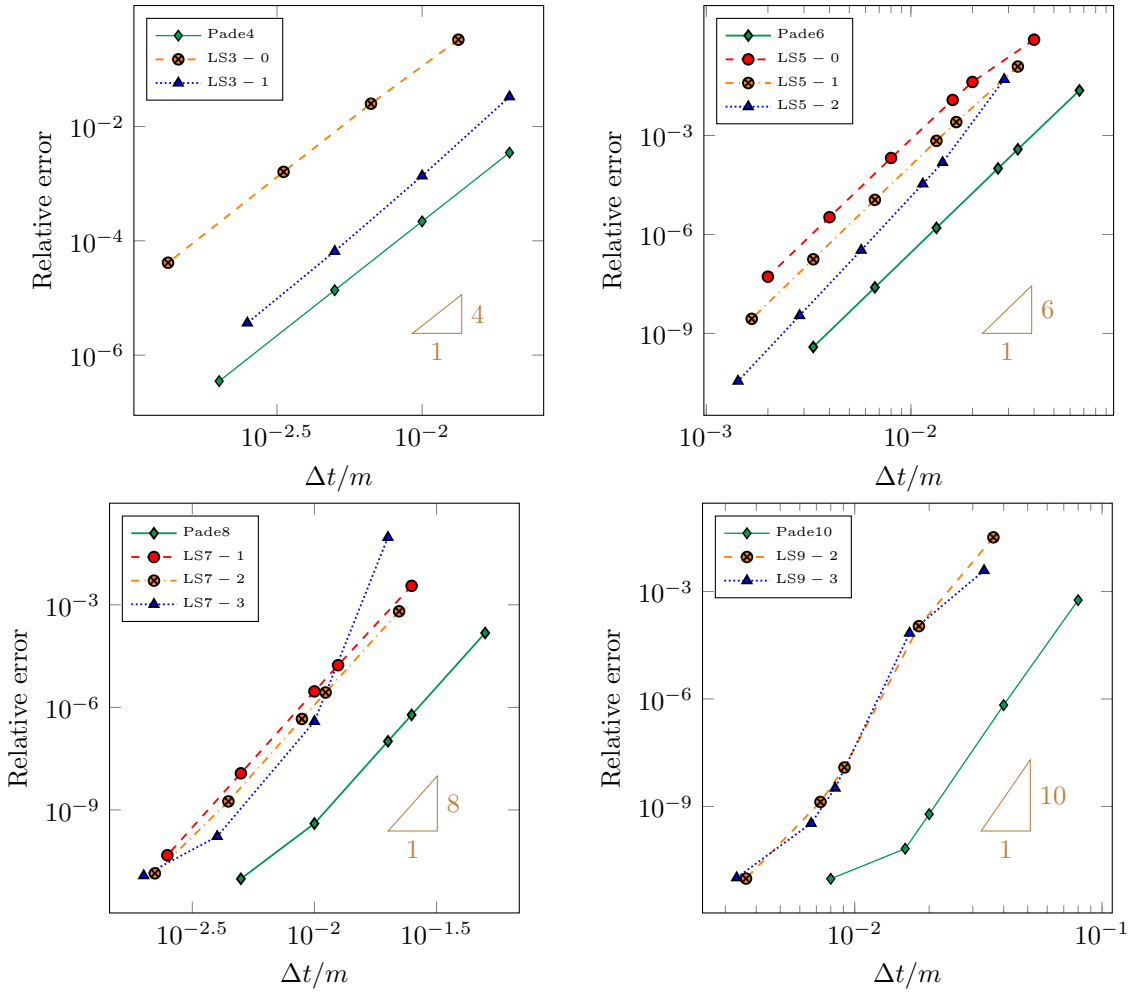



Figure 6.11: Relative $L^2$ error between numerical solution and exact solution for $t = 200$ versus the time step. Comparison of diagonal Padé and Linear-SDIRK of order 4, 6 and 8. LS$s - l$ represent the $s$ plus $l$ additional stages Linear-SDIRK of order s+1. The space discretization error is about $10^{-12}$.

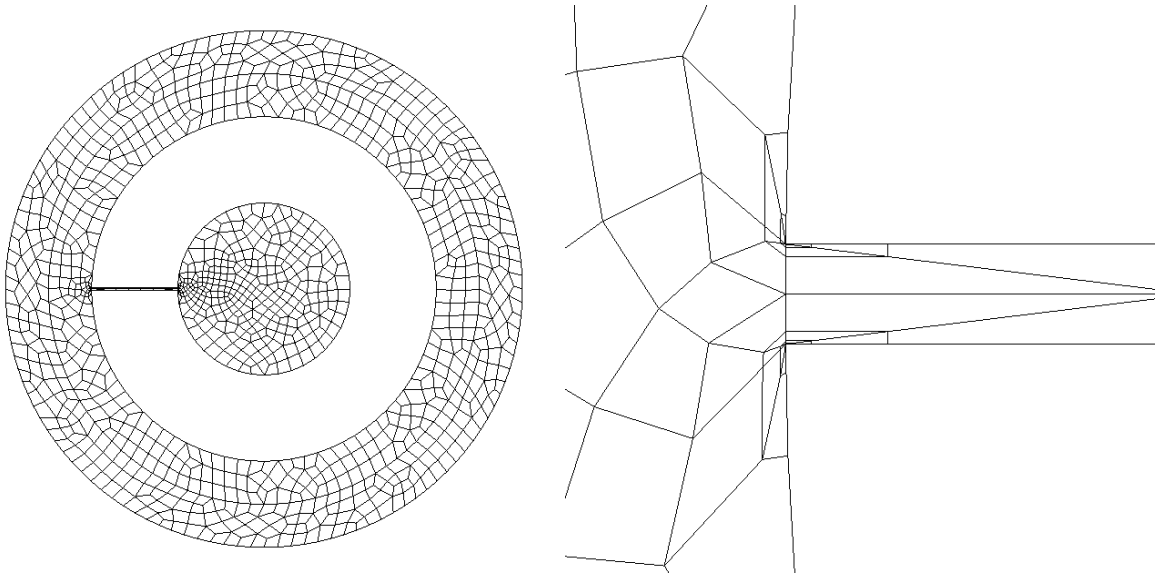


Figure 6.12: Mesh used for the resonant cavity. On the right, detail of the mesh close to the entry of the cavity.

### 6.5.2 Numerical results in 2D

**Results with an absorbing boundary condition**

In this paragraph, we consider the scattering of a resonant cavity. The computational domain $\Omega$ is meshed with quadrilaterals (see Figure 6.12). The external boundary is a circle of radius 1.5. The 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

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

An 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

$$h(t) = \sin(\omega t) e^{-b(t-T)^2}$$

| | Pade4 | LS$3-0$ | LS$3-1$ | Pade6 | LS$5-0$ | LS$5-1$ | LS$5-2$ |
|---|---|---|---|---|---|---|---|
| Number of time-steps | 2370 | 7745 | 1955 | 623 | 1393 | 856 | 823 |
| Computational time | 5mn25s | 21mn17s | 7mn27s | 2mn16s | 6mn27s | 4mn49s | 5mn26s |
| Memory (MiB) | 362.96 | 233.99 | 237.17 | 486.78 | 240.36 | 243.54 | 246.73 |

| | Pade8 | LS$7-1$ | LS$7-2$ | LS$7-3$ | Pade10 | LS$9-2$ | LS$9-3$ |
|---|---|---|---|---|---|---|---|
| Number of time-steps | 297 | 449 | 370 | 911 | 181 | 478 | 481 |
| Computational time | 1mn33s | 3mn23s | 3mn13s | 8mn40s | 1mn19s | 5mn05s | 5mn37s |
| Memory (MiB) | 606.21 | 249.91 | 253.1 | 256.29 | 730.03 | 259.48 | 262.66 |

Table 6.6: Computational time after imposing $0.1\%$ of $L^2$ relative error for the scattering of a resonant cavity.

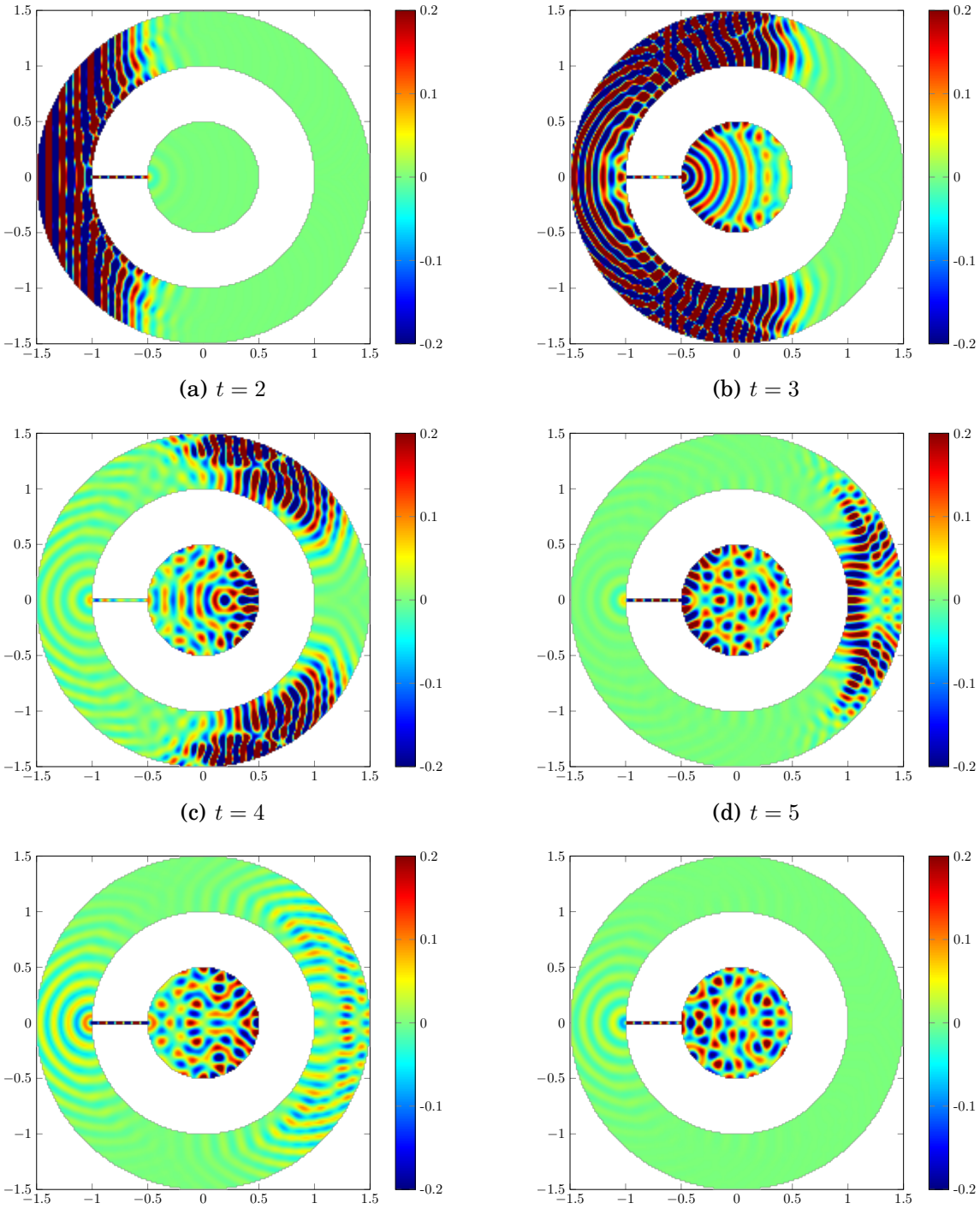
with

$$\omega = 16\pi,\ b = 4,\ T = \sqrt{\log(10^6)}$$

The solution is computed with real numbers (contrary to 1-D results), for a time interval $[0, 10]$. The solution is displayed in Figure 6.13 for $t = 2, 3, 4, 5, 6$ and at the final time $t = 10$.

The mesh is locally refined close to the points where the solution possesses a singularity (see Figure 6.12) 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 $\Delta t$ (the mesh is always the mesh of Figure 6.12), 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 6.6, the number of time iterations and the computational time needed to reach this error are given.

Padé schemes are clearly more efficient for this case. We have observed that LS$7-1$ is more efficient than LS$7-3$. This is due to the fact that the source term is not treated optimally. However, we think that by choosing appropriately the coefficients $\alpha_i^{r,l}$ and the points $c_i$ (free parameters introduced in the Section 6.4.1), it might be possible to recover a good behavior. To confirm this observation, we have set an homogeneous absorbing boundary condition and the



(a) $t = 2$ (b) $t = 3$

(c) $t = 4$ (d) $t = 5$

(e) $t = 6$ (f) $t = 10$

Figure 6.13: Solution obtained for the scattering of a resonant cavity from $t = 2, 3, 4, 5, 6$ and $10$.

| | Pade4 | LS$3-1$ | Pade6 | LS$5-0$ | LS$5-1$ | LS$5-2$ |
|---|---|---|---|---|---|---|
| Number of time-steps | 2500 | 2059 | 738 | 1642 | 1007 | 679 |
| Computational time | 5mn31s | 7mn30s | 2mn37s | 7mn38s | 5mn41s | 4mn28s |

| | Pade8 | LS$7-1$ | LS$7-2$ | LS$7-3$ | Pade10 | LS$9-2$ | LS$9-3$ |
|---|---|---|---|---|---|---|---|
| Number of time-steps | 381 | 560 | 442 | 365 | 248 | 316 | 298 |
| Computational time | 1mn53s | 4mn18s | 3mn49s | 3mn29s | 1mn36s | 3mn10s | 3mn17s |

Table 6.7: Computational time after imposing $0.1\%$ of $L^2$ relative error with an initial condition.

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 LS$7-3$ is more efficient than LS$7-1$ as expected. The comparison results of Linear-SDIRK schemes and Padé schemes are represented in Table 6.7. We see that Linear-SDIRK schemes perform well, but they are less efficient than Padé schemes.

### 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 6.14) is considered. Two consecutive inclusions are separated by 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

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

where

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

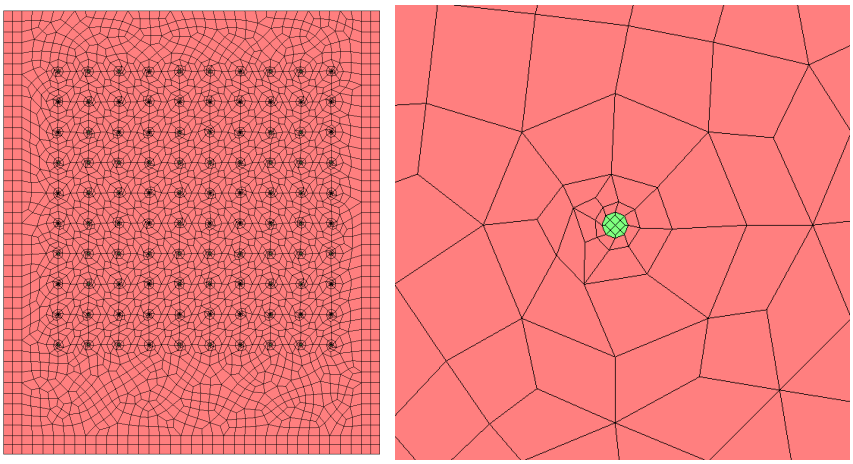
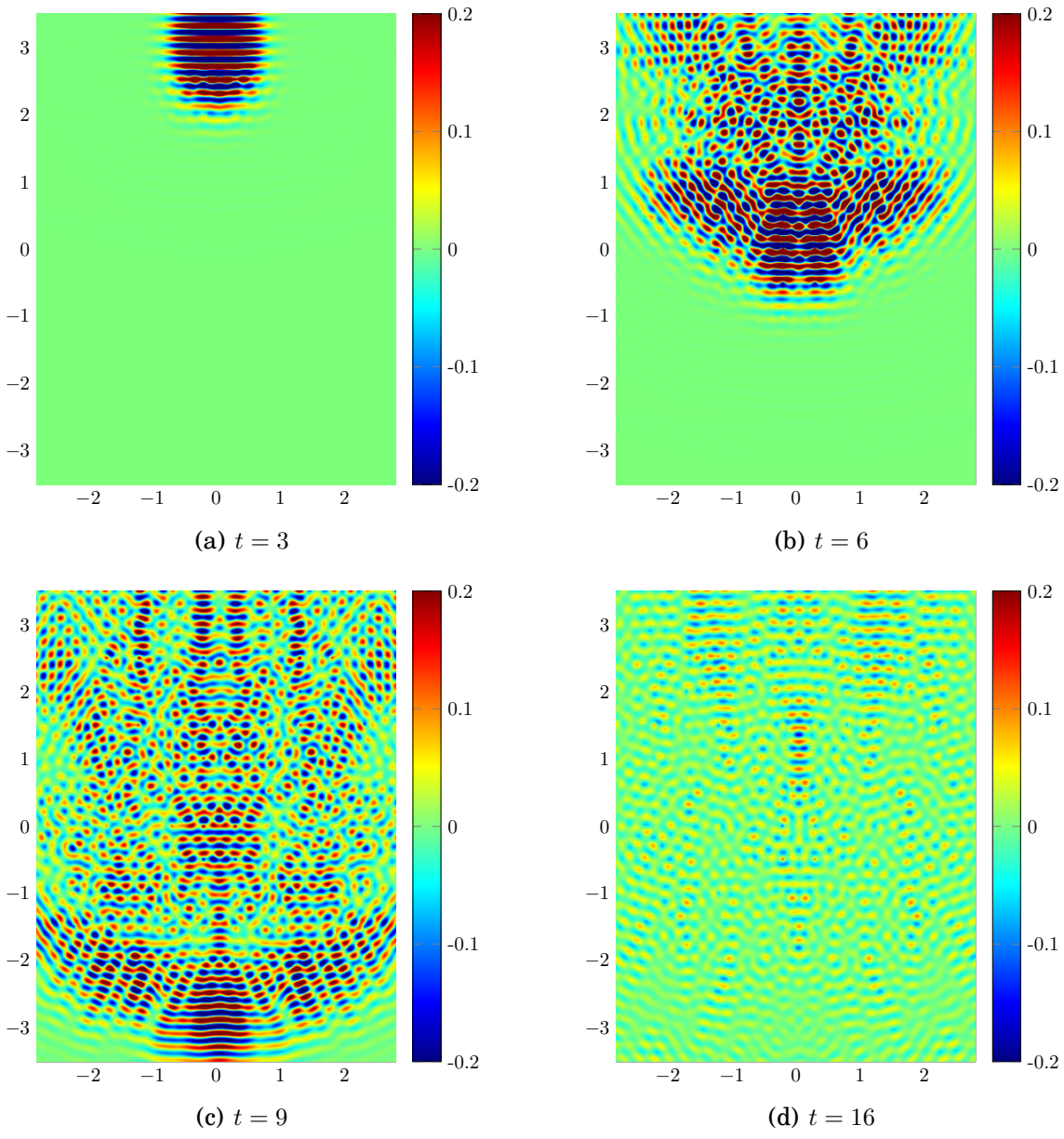


Figure 6.14: Mesh used for the array of inclusions. At right, detail of the mesh close to an inclusion (in green).

The solution for $t = 3, 6, 9$ and the final time $t = 16$ are plotted in the Figure 6.15.

We are using $\mathbb{Q}_{10}$ finite elements (as detailed previously) on the mesh of the figure 6.14 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 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 6.8, the number of time iterations and the computational time needed to reach this error are given.

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 observe that it involves high-frequency modes whereas the numerical error for Padé schemes involves the "usual" modes.




(a) $t = 3$ (b) $t = 6$

(c) $t = 9$ (d) $t = 16$

Figure 6.15: Solution obtained for the scattering of circular inclusions for $t = 3, 6, 9$ and the final time $t = 16$.

| | 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 | 16mn32s | 28mn51s | 10mn47s | 30mn53s | 20mn30s | 20mn57s |
| Memory (GiB) | 1.78 | 1.14 | 2.4 | 1.16 | 1.17 | 1.19 |

| | 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 | 7mn18s | 15mn20s | 14mn38s | 25mn24s | 4mn59s | 15mn21s | 14mn34s |
| Memory (GiB) | 3 | 1.2 | 1.22 | 1.23 | 3.62 | 1.25 | 1.26 |

Table 6.8: Computational time after imposing $1\%$ of relative $L^2$ error for the scattering of inclusions.

## 6.6 Concluding remarks for Linear-SDIRK and Padé schemes

In the Chapter 5 and the current chapter, we have investigated two different classes of one-step schemes satisfying the A-stability property the so-called 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 $\gamma$ and $\alpha_i$ until order 12. They have been implemented in the C++ code Montjoie. An implementation in Python of these schemes is also 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. But Linear-SDIRK schemes are less memory consuming compared to Padé schemes.

Concerning Linear-SDIRK, extra-stages are usually beneficial. For example, 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. [71, 44, 59]). 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 high order schemes (such as tenth order schemes), the optimization should take into account the set of coefficients $\alpha_i$ and not only $\alpha_1$ to recover the advantage of extra-stages. 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.

Chapter 7

# Higher-order optimized Explicit Runge-Kutta schemes for linear ODEs

In this chapter we present a construction of optimized explicit Runge-Kutta schemes for linear ODEs that we called Linear-ERK. The schemes are constructed using polynomial stability functions which are obtained by maximizing the CFL number. Then we provide algorithms to implement these schemes and numerical results to compare them.

## Contents

## 7.1 Introduction

As we have seen in the Chapter 4, explicit sechemes generate algorithms both cheap in memory and highly scalable. However, for stability purposes the time-step is restricted by the CFL condition which is highly related to the size of the smallest element in the mesh and by the degree of the polynomials used in the FEM or HDG methods. As a consequence, even few small elements can make the maximal value of the time-step so small that the computational cost becomes prohibitive.

Many works have been done on the construction of higher-order optimal CFL number explicit schemes. In [36, 46], the authors propose the modified equation technique to obtain high-order time schemes with an optimal CFL number for second order hyperbolic problems. In the proceeding [32], we propose optimized high-order explicit Runge-Kutta Nyström methods for second order problems. Other works has been been focussed on the optimization of the stability domain of ERK schemes [42, 48, 54, 64]. All those schemes are constructed for both linear and non-linear ODEs. In our case, the non-linear conditions [14] are too strong, and there is no need to take them into acount.

In this chapter, using the method described in [48], we compute optimal stability functions, in terms of CFL number, with respect to a typical spectrum. The obtained stability functions, are then used to construct optimized explicit Runge-Kutta schemes for linear ODEs.

## 7.2 Explicit schemes for linear ODEs

The construction of the explicit time integration schemes for linear ODE follows the rule detailed in the previous Chapters 5 and 6. We consider the same ODE

$$\begin{cases} M_h \dfrac{dX(t)}{dt} + K_h X(t) = F(t) \quad t \in (0,T] \\ X(0) = X_0 \end{cases} \tag{7.1}$$

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. 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 $\Delta t$ is the time step. The analytical solution to (7.1) after one-step is given by

$$X(t_{n+1}) = \mathrm{e}^{\Delta t A} X(t_n) + \int_0^{\Delta t} \mathrm{e}^{(\Delta t - u)A} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u, \tag{7.2}$$

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

The numerical solution is constructed by approximating the exponential with a polynomial function $R$ such that

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

The numerical schemes consist of computing a sequence $X_n$, which is an approximation of the analytical solution $X(t_n)$, with the following numerical scheme:

$$X_{n+1} = R(\Delta t A) X_n + \widetilde{\phi}_n \tag{7.3}$$

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

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

Following the Definition 2.3.1, the function $R$ is the stability function of the corresponding numerical scheme.

We assume $R(\Delta t A)$ to be an approximation of order $p$ of $\mathrm{e}^{\Delta t A}$ ($\mathrm{e}^{\Delta t A} = R(\Delta t A) + O(\Delta t^{p+1})$). The analytical solution (7.2) is then written as

$$X(t_{n+1}) = R(\Delta t A) X(t_n) + \phi, \tag{7.4}$$

where

$$\phi = \int_0^{\Delta t} \mathrm{e}^{(\Delta t - u)A} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u + O(\Delta t^{p+1}). \tag{7.5}$$

For homogeneous ODEs ($F(t) = 0$), it follows from (7.5) that $\phi = O(\Delta t^{p+1})$, we take $\phi_n = 0$ and the numerical solution (7.3) satisfies

$$X_{n+1} = R(\Delta t A) X_n \tag{7.6}$$

For inhomogeneous ODEs, i.e. $F(t) \neq 0$, to compute the quantity $\phi$, we will rather compute the following equivalent quantity obtained from (7.4):

$$\phi = X(t_{n+1}) - R(\Delta t A) X(t_n) \tag{7.7}$$

Finally the linear explicit RK scheme we propose is:

$$\boxed{X_{n+1} = R(\Delta t A) X_n + \phi_n} \tag{7.8}$$

where $\phi_n$ is an approximation of $\phi$ (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 (7.1), like in the previous chapter, we compute $\phi_n$ in the following form:

$$\phi_n = \sum_{r=1}^{m} A^{r-1} \Delta t^r \sum_{i=0}^{n_w - 1} \omega_i^r \, F\left(t_n + \Delta t\, c_i\right) \tag{7.9}$$

where $m$ is the degree of the polynomial $R$ and $n_w$ is a number that depends on the scheme.

## 7.3 Linear-ERK methods with $(s+l)$-stages of order $s$

In this section we consider a polynomial function $R(z)$ which approximates the exponential function and optimizes the CFL number. We propose to find the stability function $R(z)$ such that the CFL number is maximal.

### *7.3.1 Stability function based on Taylor's series*

In Chapter 4, Theorem 4.2.2 showed that the stability function of a RK scheme of order $p$ is given by the Taylor expansion of the exponential function at order $p$:

$$R(z) = 1 + z + \frac{z^2}{2} + \cdots + \frac{z^p}{p!} + O(z^{p+1}). \tag{7.10}$$

It seems natural to consider schemes based on the following stability function

$$R(z) = 1 + z + \frac{z^2}{2} + \cdots + \frac{z^p}{p!}. \tag{7.11}$$

Using (7.11), we can solve a linear ODE with the numerical scheme (7.8) as described above. The corresponding scheme will be naturally of order $p$ and with only $p$-stages. These schemes based on the Taylor series are equivalent to the classical ERK schemes up to order $p = 4$. In fact, from order $p = 5$, the number of stages of the classical ERK schemes are strictly greater than the order of accuracy [14]. The numerical schemes for linear ODE obtained using the Taylor series represent the explicit schemes with the less stage number for a given order $p$. In practice, these schemes will be limited by their CFL number on the imaginary axis defined as:

$$\text{CFL number on imaginary axis} = \max\{z \in \mathbb{R} \text{ such that } |R(iz)| \leq 1\}$$

For $k \in \mathbb{N}$, we have the following results:

- if $p = 4k + 1$ or $p = 4k + 2$, the CFL number is zero on the imaginary axis and the scheme is unstable,
- if $p = 4k + 3$, the CFL number is strictly lower than $\pi$ and tends toward $\frac{\pi}{2}$ when $p$ goes to infinity.
- if $p = 4k$, the CFL number is strictly below $\frac{3\pi}{2}$ and tends toward $\pi$ when $p$ goes to infinity.

These results have been observed in practice. As a consequence when using higher-order schemes, there is a need to optimize in order to get a better CFL number. This is the main goal of this chapter.

### 7.3.2 Methodology to construct an optimal stability function

We denote by $R_s^l(z)$, $s, l \in \mathbb{N}$ the stability function that corresponds to the $m = s + l$ stages Linear-ERK schemes of order $s$. By construction, it reads:

$$R_s^l(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^s}{s!} + \alpha_1 z^{s+1} + \cdots + \alpha_{s+l} z^{s+l}, \tag{7.12}$$

such that the scheme has a correct order $s$. The coefficients $\alpha_i$, $i = 1, \cdots, s+l$ are found by maximizing the CFL number with the methodology described in [48].

The first step of this methodology consists in providing the spectrum of the linear operator $A = -M_h^{-1} K_h$ of the considered ODE (7.1) (see the Chapter 3). The linear operator depends on the solved equation (acoustic or electromagnetic wave), the mesh, the chosen formulation (continuous or discontinuous), the physical coefficients $\varepsilon, \mu$... Herein, we have chosen to conduct the optimization for the HDG formulation which is more versatile. In the Figure 7.1, we have displayed the obtained spectrum for a 2D regular mesh made of quadrilateral elements covering $[-4, 4]^2$ and $\mathbb{Q}_3$ approximation. This is typical for mesh containing elements of the same size. From this spectrum, we have defined a set noted $Cabane$, that contains this typical spectrum like in the Figure 7.1.

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

$$Cabane^+ := \{it,\ t \in [0,1]\} \oplus \{i - t,\ t \in [0,1]\} \oplus \left\{ t - 2 + i\frac{t}{10}(14 - 4t),\ t \in [0,1] \right\} \tag{7.13}$$

The lower part of $Cabane$ is obtained by symmetry of this profile with respect to the real axis.

The optimization problem then reads

$$\underset{\alpha_1, \alpha_2, \cdots, \alpha_l}{\arg\max}\ \mathbf{cfl}(\alpha_1, \alpha_2, \cdots, \alpha_l) \tag{7.14}$$

where cfl is the CFL number defined as

$$\mathbf{cfl}(\alpha_1, \alpha_2, \cdots, \alpha_l) = \max\{\Delta t \text{ such that } \Delta t \times Cabane \subset \mathcal{S}(\alpha_1, \alpha_2, \cdots, \alpha_l)\} \tag{7.15}$$

where $\mathcal{S}(\alpha_1, \alpha_2, \cdots, \alpha_l)$ is the stability region of the corresponding Linear-ERK scheme given by:

$$\mathcal{S}(\alpha_1, \alpha_2, \cdots, \alpha_l) = \{z \in \mathbb{C},\ |R_s^l(z)| \leq 1\}.$$

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

$$\Delta t \leq \frac{\mathbf{cfl}(\alpha_1, \alpha_2, \cdots, \alpha_l)}{\max_{\lambda \in sp(A)} |Im(\lambda)|} \tag{7.16}$$

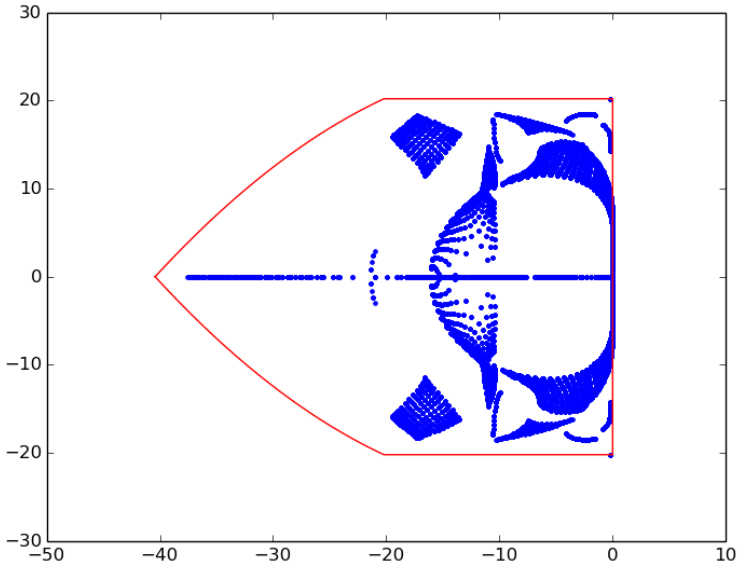



Figure 7.1: Spectrum of $A$ (blue points are eigenvalues of $A$) in the complex plane included in a set $\alpha \times Cabane$ (inside the red polygon) obtained when solving the acoustic wave equation with HDG formulation and $\mathbb{Q}_3$ polynomials. The computational domain is a square $[-4,4]^2$ of $15 \times 15$ elements. Neumann condition is set on the boundary of the square.

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

$$\Delta t = \max \{\Delta t \text{ such that } \Delta t \lambda \in \mathcal{S}(\alpha_1, \alpha_2, \cdots, \alpha_l) \; \forall \lambda \in sp(A)\}. \tag{7.17}$$

Note that in [48] the authors use the true spectrum of the operator and look for the maximal time-step $h$ that satisfies the stability constraint. After optimization the obtained value $h$ represents the maximal time step such that $z = h\lambda \in \mathcal{S}$ for all $\lambda \in sp(A)$, $sp(A)$ representing the spectrum of $A$. Here, we consider a normalized envelope Cabane, and for the operator $A$ the maximal time-step is obtained from (7.17).

To compare the different explicit schemes, we introduce the following definition:

**Definition 7.3.1** *Let $R_s^l(z)$ be the stability function of a Linear-ERK scheme of order $s$ with $s+l$ stages. Assume the CFL number of this scheme is given by (7.15). Then, the efficiency of the corresponding scheme is defined by*

$$\textit{Efficiency} = \frac{\textit{CFL number}}{s+l}. \tag{7.18}$$

## 7.4 Computation of the RHS term and algorithm

### 7.4.1 Handling a right hand side term

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

$$R_s^l(z) = N_s^l(z).$$

with $N_s^l(z)$ defined in the previous subsections. The equation (7.7) becomes then

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

Let us denote the coefficients of polynomial $N_s^l$:

$$N_s^l(z) = \sum_{i=0}^{s+l} N_i z^i$$

Like in the Chapters 5 and 6, 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 $s$. 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 $t_n + \dfrac{\Delta t}{2}$. We obtain:

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

where

$$\begin{aligned} \delta_{i,0} &= 1, \text{ if } i = 0; \\ \delta_{i,0} &= 0, \text{ otherwise} \end{aligned}$$

Then, we use the relation

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

derived from the equation

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

to obtain the following expression

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

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

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

where

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

As it has been noticed for Linear-SDIRK, the sum in $r$ and $j$ can go until $s + l$ instead of $s, s + 1 - r$ without changing the order of the method. When $l$ is large, this substitution should improve the solution in the inhomogeneous case. Finally we consider the approximation

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

We choose $s+1$ Gauss-Legendre points for $c_i$ in the interval $[0, 1]$ ($n_w = s+1$), and the weights $\omega_i^{r,l}$ are found by solving a Vandermonde system as detailed before in the Chapter 5. $\phi_n$ is therefore computed by using the formula

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

### *7.4.2 Algorithm based on the roots of the stability function*

To prevent from the eventual risk of instability (when using Hörner's algorithm for the polynomial of higher degrees) as we have observed for the Linear-SDIRK schemes in the Chapter 6, we propose also an algorithm based on the roots of the stability function for the Linear-ERK schemes as well. When no source is present, this algorithm consists in factorizing the stability function $R_s^l(z)$ as follows

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

and of using this factorization to write the algorithm given in the Algorithm 6. We have detailed the algorithm 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 the source term is added, we rewrite $\phi_n$ in the following form:

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

**Algorithm 6** Stable algorithm without source term for Linear-ERK schemes

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

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

$$Q_0(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right)\left(1 - \frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1 - \frac{z}{\lambda_{n_1}}\right)\left(1 - b_{n_c} z + a_{n_c} z^2\right)\cdots\left(1 - b_2 z + a_2 z^2\right) z$$

$$Q_1(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right)\left(1 - \frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1 - \frac{z}{\lambda_{n_1}}\right)\left(1 - b_{n_c} z + a_{n_c} z^2\right)\cdots\left(1 - b_2 z + a_2 z^2\right)$$

$$Q_2(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right)\left(1 - \frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1 - \frac{z}{\lambda_{n_1}}\right)\left(1 - b_{n_c} z + a_{n_c} z^2\right)\cdots\left(1 - b_3 z + a_3 z^2\right) z$$

$$Q_3(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right)\left(1 - \frac{z}{\lambda_{n_r-1}}\right)\cdots\left(1 - \frac{z}{\lambda_{n_1}}\right)\left(1 - b_{n_c} z + a_{n_c} z^2\right)\cdots\left(1 - b_3 z + a_3 z^2\right)$$

$$\cdots$$

$$Q_{s+l-2}(z) = \left(1 - \frac{z}{\lambda_{n_r}}\right)$$

$$Q_{s+l-1}(z) = 1$$

Let us denote:

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

The stable algorithm is described in the Algorithm 7.

## 7.5 Optimal coefficients and CFL number for the schemes from order 2 to 8

In this section we provide the optimal coefficients we have obtained for the Linear-ERK schemes of order $s$ with $l \geq 0$ additional stages noted Linear-ERK$s - l$. To solve the problem (7.14) we have used RK-opt, a package of optimization of RK methods ([49], [48]). We have run this code with the shape of the

**Algorithm 7** Stable algorithm with source term for Linear-ERK schemes

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

$Cabane$ that represents the typical spectrum of our problems. After optimization of the stability function, the maximal CFL number obtained and the stability region of each scheme are given. We provide even-order schemes only to match with even-order implicit schemes we have provided in the previous Chapters 5 and 6. Both explicit and implicit schemes are then used in the Chapter 8 to construct locally implicit schemes.

### 7.5.1 *Linear-ERK*$2 - l$ *schemes of order 2*

The coefficients obtained in the construction of Linear-ERK schemes of order 2 are provided in this part. The stability function of these schemes is obtained for $s = 2$ in the equation (7.12) and takes the form

$$R_2^l(z) = 1 + z + \frac{z^2}{2} + \alpha_1 z^{s+1} + \cdots + \alpha_{s+l} z^{s+l}. \tag{7.19}$$

where the optimal coefficients $\alpha_i$, $i = 1, \cdots, l$ are given in Table 7.1. The maximal CFL number (7.15) and efficiency according to the Definition 7.3.1 are given in Table 7.2 for each scheme of order 2. We observe that the CFL number in the typical set $Cabane$ is zero for the scheme without additional stage and it increases with the number of stages. The efficiency also increases.

We have plotted in the Figure 7.2 the CFL number and the efficiency with respect to the number of stages.

The stability domains of the different schemes of order 2 are represented in the Figure 7.3. In the Figure 7.3(b), we show the stability region taking into account the number of stages for each scheme. That is why we have chosen Real(z)/(s+l) and Imag(z)/(s+l) on the real and imaginary axes respectively. We observe that by increasing the number of additional stages $l$, the stability region tends to the desired shape $Cabane$ and includes more and more points of the imaginary axis.

| $l$ | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ |
|---|---|---|---|
| 1 | 1.451277982649155e-1 | | |
| 2 | 1.665532314108146e-1 | 2.327815361933148e-2 | |
| 3 | 1.618342913053687e-1 | 3.289792611743811e-2 | 2.839528016518102e-3 |
| 4 | 1.642981320398038e-1 | 3.657769285804588e-2 | 5.035250867609586e-3 |
| 5 | 1.626462249413356e-1 | 3.762678272315501e-2 | 5.996644250417070e-3 |
| 6 | 1.627509585676844e-1 | 3.773348832445807e-2 | 6.387803046851333e-3 |
| 7 | 1.640094942014296e-1 | 3.840429977823329e-2 | 6.724597512047917e-3 |
| 8 | 1.649990588856614e-1 | 3.927394350377206e-2 | 7.055384479248899e-3 |

| | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ |
|---|---|---|---|
| 4 | 3.001880509358407e-4 | | |
| 5 | 5.826143210213330e-4 | 2.487327304531716e-5 | |
| 6 | 7.489561665296774e-4 | 5.356270766078865e-5 | 1.713109940102836e-6 |
| 7 | 8.718626803227696e-4 | 7.857554562878064e-5 | 4.327975378833797e-6 |
| 8 | 9.695797812914759e-4 | 9.943224646288322e-5 | 7.129812259258231e-6 |

| | $\alpha_7$ | $\alpha_8$ | |
|---|---|---|---|
| 7 | 1.072985856243921e-7 | | |
| 8 | 3.148056880771953e-7 | 6.324920988294407e-9 | |

Table 7.1: Coefficients for the Linear-ERK schemes of order **2** with $l$ additional stages

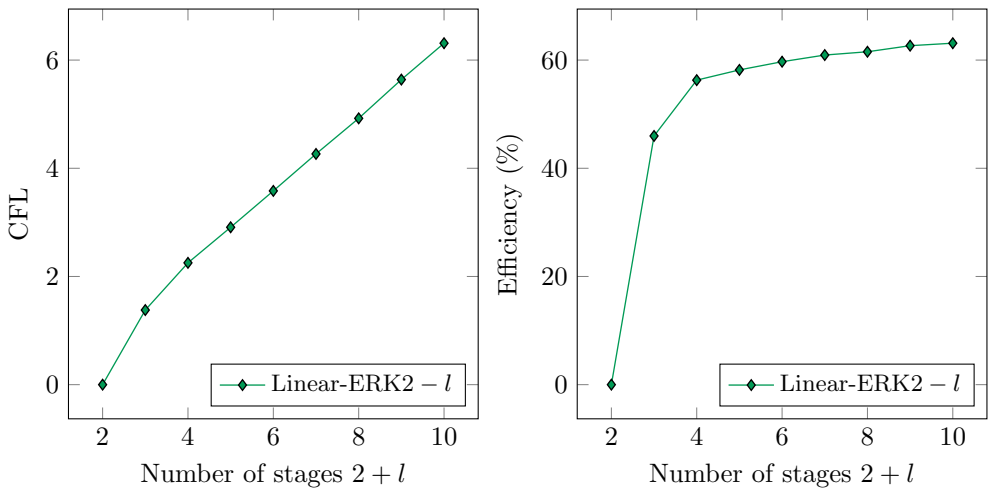
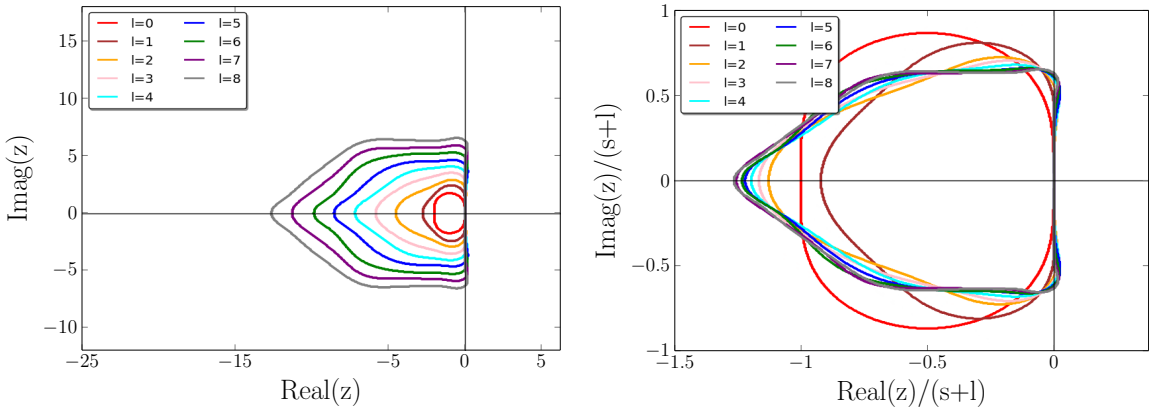



Figure 7.2: CFL number and efficiency of the Linear-ERK schemes of order 2 with respect to the number of stages $2+l$, $l=0,\cdots,8$.

| *Schemes* | Linear-ERK$2-0$ | Linear-ERK$2-1$ | Linear-ERK$2-2$ |
|---|---|---|---|
| CFL | 0 | 1.379212 | 2.251664 |
| Efficiency | 0% | 45.9 % | 56.2% |
| | | | |
| *Schemes* | Linear-ERK$2-3$ | Linear-ERK$2-4$ | Linear-ERK$2-5$ |
| CFL | 2.909154 | 3.581817 | 4.265085 |
| Efficiency | 58.1% | 59.6% | 60.9 % |
| | | | |
| *Schemes* | Linear-ERK$2-6$ | Linear-ERK$2-7$ | Linear-ERK$2-8$ |
| CFL | 4.922950 | 5.639401 | 6.311962 |
| Efficiency | 61.5% | 62.6% | 63.1% |

Table 7.2: CFL number in the typical profile $Cabane$ for the Linear-ERK$2-l$ schemes of order **2** with $2+l$ stages.




(a) Stability region

(b) Stability region with the complexity

Figure 7.3: Stability region of the Linear-ERK schemes of order 2.

| $l$ | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ |
|---|---|---|---|
| 1 | 4.730163010446185e-3 | | |
| 2 | 6.541349497416528e-3 | 4.395282130923843e-4 | |
| 3 | 7.241999849787970e-3 | 7.614940065988191e-4 | 3.521874589831831e-5 |
| 4 | 7.603292194142675e-3 | 9.535828377031919e-4 | 7.298469178025099e-5 |
| 5 | 7.817918289656257e-3 | 1.075759999127459e-3 | 1.026588721744709e-4 |
| 6 | 7.992535147077134e-3 | 1.180030987873825e-3 | 1.307878349087823e-4 |
| 7 | 9.619397138072583e-3 | 3.970757223041604e-3 | 1.979923031733034e-3 |
| 8 | 8.105487675563905e-3 | 1.249316412377197e-3 | 1.531845812394507e-4 |
| | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ |
| 4 | 2.500124976522895e-6 | | |
| 5 | 6.038353896295552e-6 | 1.628169027707504e-7 | |
| 6 | 1.020785594818226e-5 | 4.943966219870204e-7 | 1.097077616437946e-8 |
| 7 | 6.726632799312973e-4 | 1.385778310637994e-4 | 1.585824201586086e-5 |
| 8 | 1.473468121845849e-5 | 1.071860716775002e-6 | 5.510748021396615e-8 |
| | $\alpha_7$ | $\alpha_8$ | |
| 7 | 7.742514686545619e-7 | | |
| 8 | 1.766727504578043e-9 | 2.623218531216638e-11 | |

Table 7.3: Coefficients for the Linear-ERK schemes of order **4** with $l$ additional stages.

### 7.5.2 *Linear-ERK*$4-l$ *schemes of order 4*

The fourth order Linear-ERK schemes have been constructed using the following stability function:

$$R_s^l(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^4}{4!} + \alpha_1 z^{s+1} + \cdots + \alpha_{s+l} z^{s+l}. \tag{7.20}$$

Its corresponds to $s = 4$ in the equation (7.12). The optimal coefficients obtained for $\alpha_i$, $i = 1, \cdots, l$ are given in Table 7.3. The corresponding CFL number (7.15) and the efficiency following Definition 7.3.1 are provided in Table 7.4.

In the Figure 7.4 we show the variation of the CFL number obtained with respect to number of stages. The stability domains of the different schemes of order 4 are shown in the Figure 7.5. Again, we show in the Figure 7.5(b) 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 $y$

| *Schemes* | $LinearERK4-0$ | $LinearERK4-1$ | $LinearERK4-2$ |
|---|---|---|---|
| CFL | 1.392646 | 2.483669 | 3.129610 |
| Efficiency | 34.8% | 49.6% | 52.1% |
| | | | |
| *Schemes* | $LinearERK4-3$ | $LinearERK4-4$ | $LinearERK4-5$ |
| CFL | 3.961619 | 4.577616 | 5.044231 |
| Efficiency | 56.5% | 57.2% | 56% |
| | | | |
| *Schemes* | $LinearERK4-6$ | $LinearERK4-7$ | $LinearERK4-8$ |
| CFL | 5.744698 | 2.947906 | 7.146060 |
| Efficiency | 57.4% | 26.7% | 59.5% |

Table 7.4: CFL in the typical profile $Cabane$ for the Linear-ERK$4-l$ schemes of order **4** with $4+l$ stages.

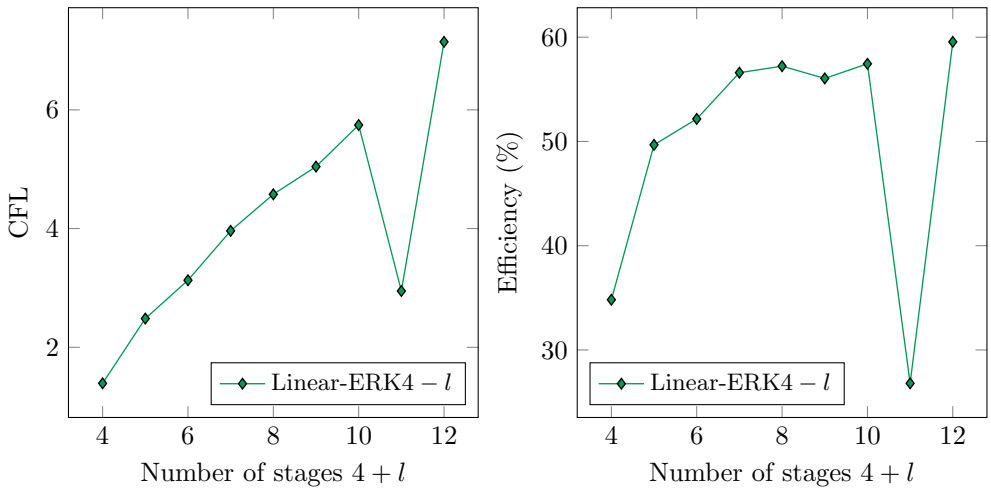



Figure 7.4: CFL number and efficiency of the Linear-ERK schemes of order $4$ with respect to the number of stage $4+l$, $l=0,\cdots,8$.

| *Schemes* | $LinearERK4-0$ | $LinearERK4-1$ | $LinearERK4-2$ |
|---|---|---|---|
| CFL | 2.828427 | 3.309192 | 3.748643 |
| Efficiency | 70.7% | 66.1% | 62.4% |
| | | | |
| *Schemes* | $LinearERK4-3$ | $LinearERK4-4$ | $LinearERK4-5$ |
| CFL | 4.168552 | 4.594556 | 5.044231 |
| Efficiency | 59.5% | 57.4% | 56% |
| | | | |
| *Schemes* | $LinearERK4-6$ | $LinearERK4-7$ | $LinearERK4-8$ |
| CFL | 5.744698 | 2.996975 | 7.146060 |
| Efficiency | 57.4% | 27.2% | 59.5% |

Table 7.5: CFL number on the imaginary axis for the Linear-ERK$4-l$ schemes of order **4** with $4+l$ stages.

coordinate respectively. We observe that by increasing the number of additional stages $l$, the stability region tends to the desired shape $Cabane$. We also notice that the stability region of these ERK schemes includes more and more points of the real axis compared to classical ERK schemes and as shown in the Table 7.4 the CFL number and the efficiency on the $Cabane$ profile increase. However, we loose in terms of efficiency on the imaginary axis. In fact, in the second plot of the Figure 7.5 we observe that the ERK schemes with no additional stage include more points on the imaginary axis than the others. Table 7.5, in which we present the CFL number on the imaginary axis, displays the efficiency decreasing while the number of additional stages increases.

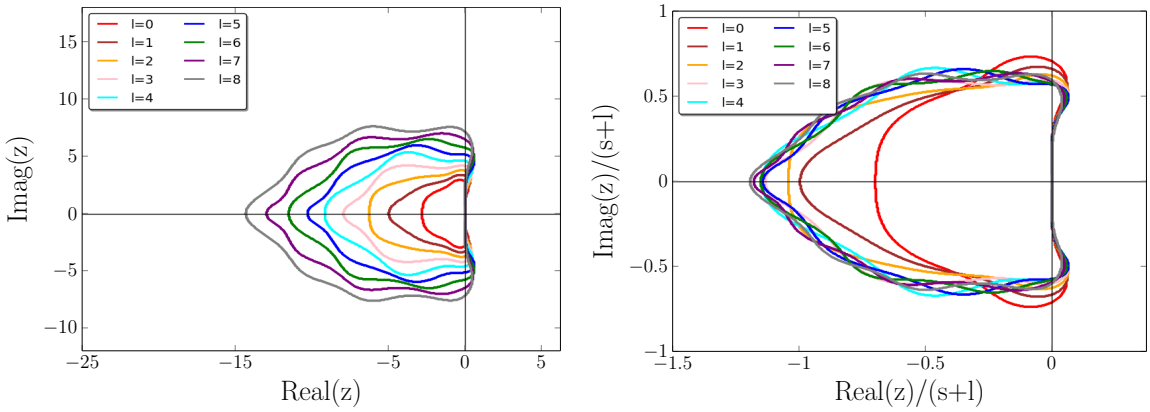
### 7.5.3 *Linear-ERK*$6-l$ *schemes of order 6*

The Linear-ERK schemes are found for $s=6$ in the equation (7.12). The stability functions are then given by

$$R_s^l(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^6}{6!} + \alpha_1 z^{s+1} + \cdots + \alpha_{s+l} z^{s+l}. \tag{7.21}$$

The optimal value of the $l$ free parameters $\alpha_i$, $i = 1, \cdots, l$ are provided in Table 7.6. The optimal CFL (7.15) obtained and the efficiency regarding Definition 7.3.1 are given for each scheme of order 6 in Table 7.7. It is worth noting that the CFL number in the typical profile $Cabane$ is also zero for the scheme with no additional stage (Linear-ERK$6-0$). This scheme is unstable on the imaginary axis. The CFL number and the efficiency increase with additional stages.

Like the second and fourth order Linear-ERK schemes, we have plotted the variation of the optimal time-step $\Delta t$ obtained with respect to the number of




(a) Stability region

(b) Stability region with the complexity

Figure 7.5: Stability region of the Linear-ERK schemes of order 4.

| $l$ | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ |
|---|---|---|---|
| 1 | 2.070461615593214e-4 | | |
| 2 | 2.204061707466545e-4 | 1.942982735313673e-5 | |
| 3 | 2.073919102492977e-4 | 2.499262304459253e-5 | 1.453234258464881e-6 |
| 4 | 2.358338644436141e-4 | 4.056334413908446e-5 | 4.775871882059528e-6 |
| | $\alpha_4$ | - | - |
| 4 | 2.442645091656458e-07 | - | - |

Table 7.6: Coefficients for the Linear-ERK schemes of order **6** with $l$ additional stages.

| *Schemes* | $LinearERK6-0$ | $LinearERK6-1$ | $LinearERK6-2$ |
|---|---|---|---|
| CFL | 0 | 1.946294 | 2.893398 |
| Efficiency | 0% | 27.8% | 36.1% |
| | | | |
| *Schemes* | $LinearERK6-3$ | $LinearERK6-4$ | |
| CFL | 3.555059 | 3.566593 | - |
| Efficiency | 39.5% | 35.6% | - |

Table 7.7: CFL number and efficiency on the typical profile *Cabane* for the Linear-ERK$6-l$ schemes of order **6** with $6+l$ stages.

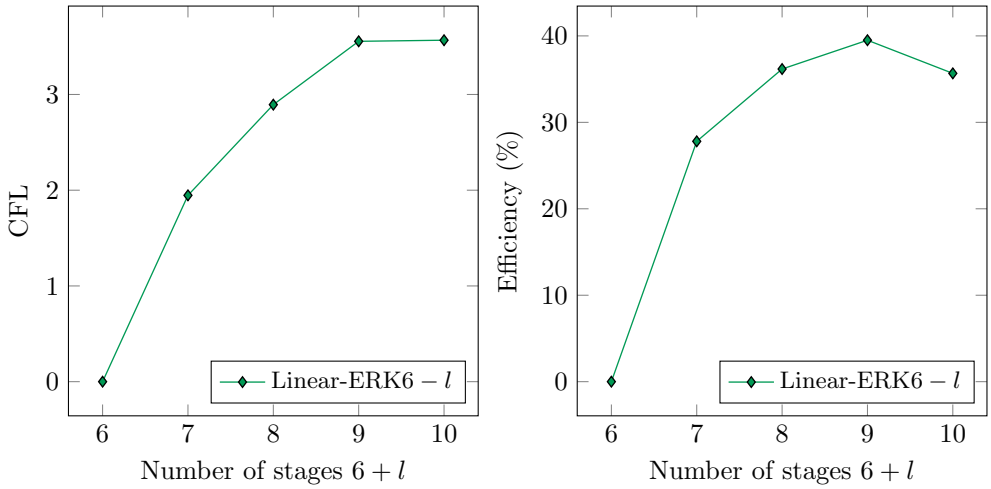



Figure 7.6: CFL number and efficiency of the Linear-ERK schemes of order $6$ with respect to the number of stages $6+l$, $l=0,\cdots,6$.

stages in the Figure 7.6.

In the Figure 7.7, we represent the stability region of the Linear-ERK schemes of order 6. In the Figure 7.5(b) the stability region is displayed using Real(z)/(s+l) and Imag(z)/(s+l) in the real and imaginary axes respectively. We observe that by increasing the number of additional stages $l$, the stability region tends to the wanted shape $Cabane$ and includes more and more points of the imaginary axis.

### 7.5.4 *Linear-ERK*$8-l$ *schemes of order 8*

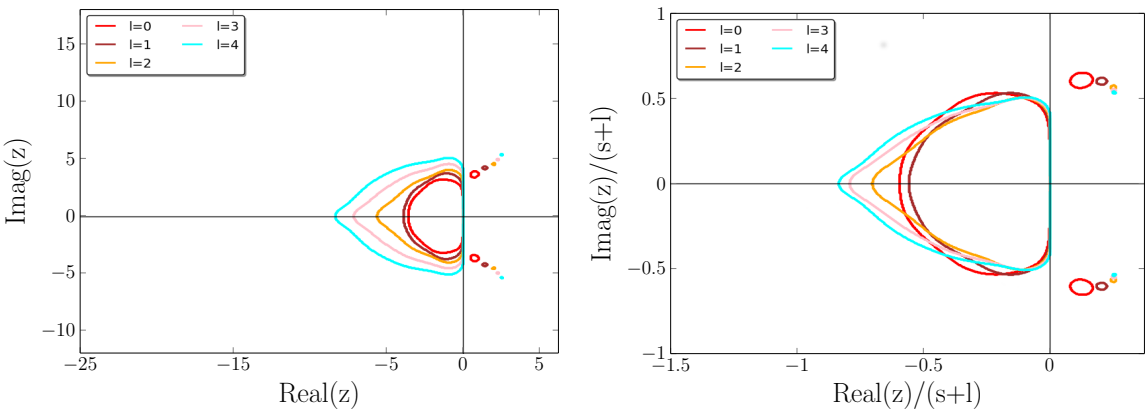
The eighth order Linear-ERK schemes is constructed by taking $s=8$ in the equation (7.12) which corresponds to the following stability function:

$$R_s^l(z)=1+z+\frac{z^2}{2!}+\cdots+\frac{z^8}{8!}+\alpha_1 z^{s+1}+\cdots+\alpha_{s+l}z^{s+l}. \tag{7.22}$$

The optimal values obtained for $\alpha_i$, $i=1,\cdots,l$ are given in Table 7.8. The maximal CFLs and the efficiency obtained for each scheme of order 8, are provided in Table 7.9.

In the Figure 7.8, we have plotted the CFL number and the efficiency obtained after optimization, with respect to the number of stages.

The CFLs on the imaginary axis are also given in Table 7.10. They are given for schemes that have a different CFL number (on the imaginary axis) from their CFL number on the profile $Cabane$.




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

Figure 7.7: Stability region of the Linear-ERK schemes of order 6.

| $l$ | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ |
|---|---|---|---|
| 1 | 1.684112035592431e-6 | | |
| 2 | 2.288709306973234e-6 | 9.960040692054680e-8 | |
| 3 | 2.528206540248994e-6 | 1.724423811134767e-7 | 5.449535772542617e-9 |
| 4 | 2.638893313733145e-6 | 2.150620166601062e-7 | 1.123553506837818e-8 |
| 5 | 2.703333893632985e-6 | 2.435581983430564e-7 | 1.631043038503232e-8 |
| 6 | 2.711246141311401e-6 | 2.500568374959440e-7 | 1.817647917892119e-8 |

| | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ |
|---|---|---|---|
| 4 | 2.690758844819519e-10 | | |
| 5 | 6.905312067380033e-10 | 1.342332862257654e-11 | |
| 6 | 9.481642471601341e-10 | 3.089127728872379e-11 | 4.655664953646905e-13 |

Table 7.8: Coefficients for the Linear-ERK schemes of order **8** with $l$ additional stages.

| *Schemes* | $LinearERK8-0$ | $LinearERK8-1$ | $LinearERK8-2$ |
|---|---|---|---|
| CFL | 2.1568136 | 3.274393 | 3.978773 |
| Efficiency | 26.9% | 36.3% | 39.7% |
| | | | |
| *Schemes* | $LinearERK8-3$ | $LinearERK8-4$ | $LinearERK8-5$ |
| CFL | 4.654201 | 5.419076 | 6.007948 |
| Efficiency | 42.3% | 45.1% | 46.2% |
| | | | |
| *Schemes* | $LinearERK8-6$ | - | - |
| CFL | 6.178560 | - | - |
| Efficiency | 44.1% | - | - |

Table 7.9: CFL number and efficiency on the typical profile $Cabane$ for the Linear-ERK$8-l$ schemes of order **8** with $8+l$ stages.

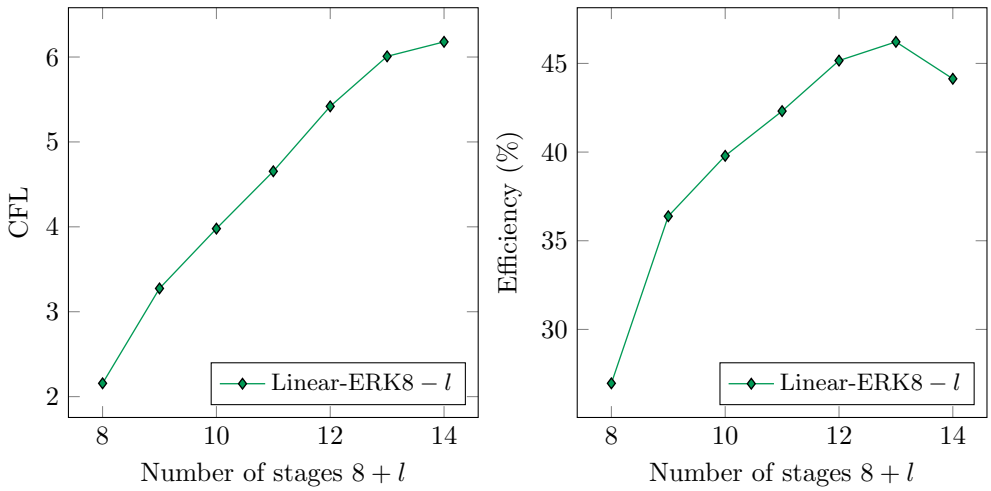
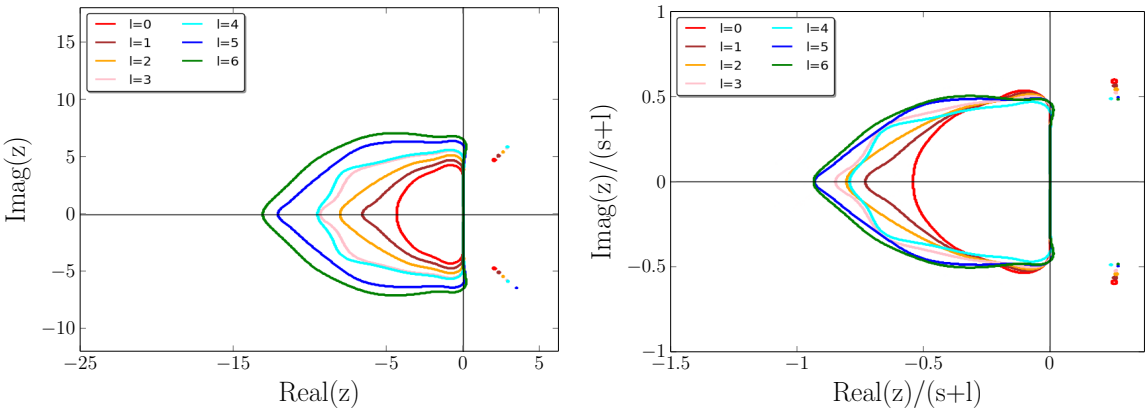



Figure 7.8: CFL number and efficiency on the typical profile $Cabane$ of the Linear-ERK schemes of order $8$ with respect to the number of stage $8+l$, $l=0,\cdots,6$.




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

Figure 7.9: Stability region of the Linear-ERK schemes of order 8.

| *Schemes* | $LinearERK8-0$ | $LinearERK8-1$ | $LinearERK8-2$ |
|---|---|---|---|
| CFL | 3.395140 | 3.935957 | 4.452846 |
| Efficiency | 42.4% | 43.7% | 44.5% |
| | | | |
| *Schemes* | $LinearERK8-3$ | - | - |
| CFL | 4.938094 | - | - |
| Efficiency | 44.8% | - | - |

Table 7.10: CFL number and efficiency on the imaginary axis for the Linear-ERK$8-l$ schemes of order **8** with $8+l$ stages.

## 7.6 Dispersion of the Linear-ERK$s-l$ schemes

To measure the dispersion error we consider the test equation $y' = i\lambda y$ and use the formula $\Phi(z) = z - arg[R(iz)]$. As detailed in the Chapter 2 $z = \Delta t\lambda$ where $\Delta t$ is the time-step.
In the Figure 7.10 we show the relative dispersion errors of the Linear-ERK$s-l$ schemes of order 2, 4, 6 and 8. In the $x$-axis we have chosen $\frac{z}{m}$ where $m = s + l$ is the number of stages of the scheme. $m$ represents by this way the number of evaluations of the ODE, i.e. the number of matrix-vector products with $\Delta tA$ while computing the numerical solution.

Clearly the Figure 7.10 shows that high-order schemes are less dispersive compared to low-order schemes. We can also note that among the schemes with same order, the dispersion error curves are similar. However, in the group of second order schemes (see the Figure 7.10(a)), the one with two additional stages (Linear ERK$2-2$ in the legend) seems more accurate compared to the others.

## 7.7 Numerical comparison results for the solution of the acoustic wave equation

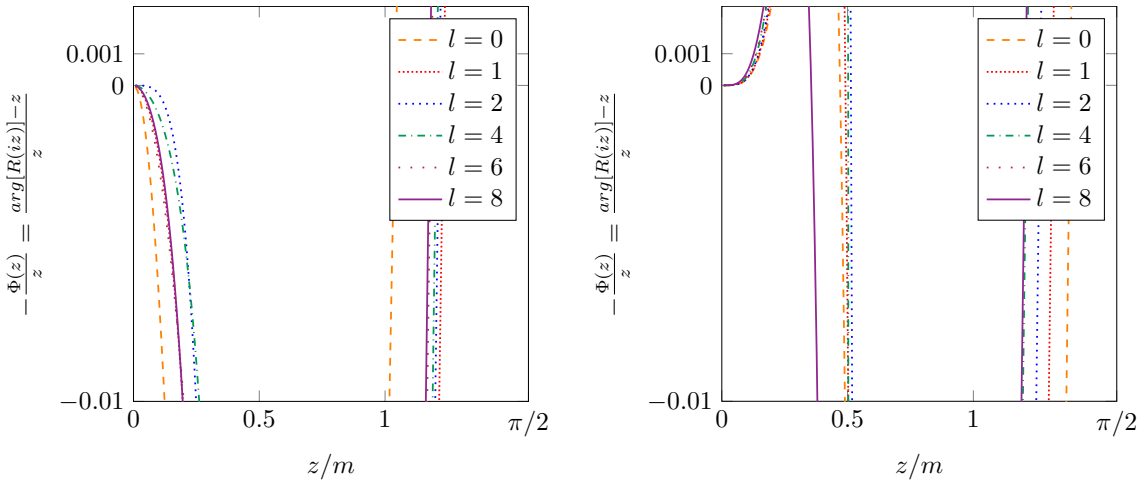
The numerical simulations in this section are conducted using the first order formulation of the acoustic wave equation as presented in the Chapter 3 with HDG formulation. First we present some validation tests of the Linear-ERK schemes we have developed. Then we present numerical results obtained for the scattering of a dielectric disk.

### *7.7.1 Numerical results and validation on a regular square mesh*

In this subsection the space domain is a 2D square mesh $[-4, 4]^2$ in which we use the $10^{th}$ order HDG method implemented in the C++ code Montjoie [31]. The mesh is constructed by choosing 10 points along the $x$ and $y$ coordinate. The homogeneous Neumann condition has been set on the boundaries and we have chosen a volumetric source using a Gaussian function centered at $(0, 0)$ with the distribution radius equal to 1. The frequency is also equal to 1.

The time interval is $[0, 200]$. The temporal source corresponds to the second derivative of the Gaussian function. The numerical reference solution has been computed using the eighth order implicit Padé scheme with $\Delta t = 0.01$. Then we have computed the relative error of the different Linear-ERK schemes at the final time $t = 200$.

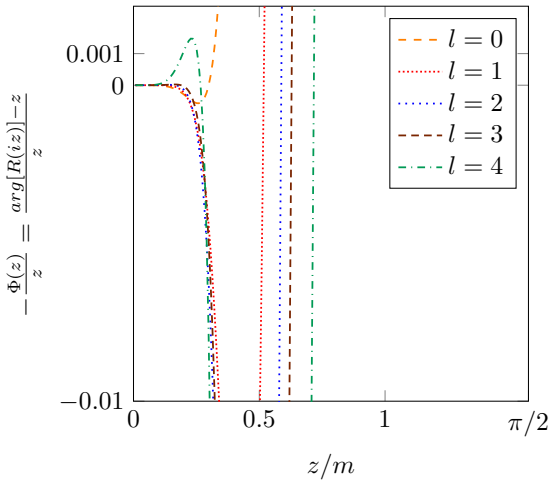
We present two comparison results in this section. First we present a result obtained for each scheme handled with the limit value of the CFL. Then we consider the case where the time-step takes into account the complexity of each scheme.



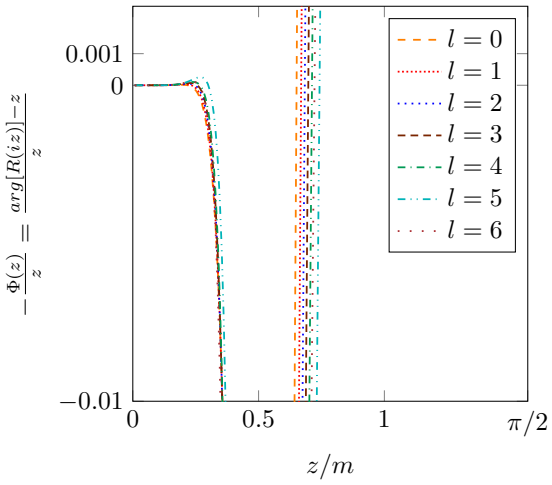

(a) Linear-ERK$2 - l$




(b) Linear-ERK$4 - l$




(c) Linear-ERK$6 - l$




(d) Linear-ERK$8 - l$

Figure 7.10: Relative dispersion errors of Linear-ERK$s - l$ schemes of order s equals 2, 4, 6 and 8 with $s + l$ stages.

**Comparison results near the CFL number**

In this case, we have chosen the maximal time step satisfying the stability condition. In Table 7.11 we present the comparison results of the optimized Linear-ERK$s-l$ schemes. First we observe that increasing the number of stages is beneficial for the stability. It allows to take larger and larger time-steps (and thus less number of time-steps) since the stability region is bigger and bigger. Second we notice that low-order schemes are more stable than high-order schemes. In fact, they requires less evaluations to obtain a stable solution compared to high-order schemes. But they have a large relative error when the time step is chosen close to the CFL number. Therefore they are less accurate compared to high-order schemes as expected. This points out the need to increase the order of accuracy. Finally, among the schemes of order 2, the one with 8 additional stages is the most stable but at the CFL level it is not accurate enough. For the schemes of order 4, we observed that adding one stage provides a large saving in terms of computational cost, and the most stable scheme appears to be the one with 8 additional stages. For schemes of order 6 and 8 the most stable schemes correspond to the schemes with 9 and 14 number of stages respectively. They require less evaluations compared to the other schemes of the same order.

**Comparison results with the same complexity**

To compare the different Linear-ERK$s-l$ schemes, we have chosen a reference time step $\Delta t_{ref} = 0.0005$. Then we have computed the time step $\Delta t_{sl}$ corresponding to the time step to be chosen for Linear-ERK$s-l$ as follows:

$$\Delta t_{sl} = (s+l) \times \Delta t_{ref}.$$

By this way the chosen time-step includes the complexity of each scheme since $s+l$ represents the total number of stages of the Linear-ERK$s-l$ scheme. So, the number of evaluations ($s+l$ multiplied by the number of time-steps) is approximatively the same for all schemes. In the Table 7.12 we present the obtained results.

First we observe that high-order schemes are more accurate than low-order schemes as expected. Among the schemes of order 2, the Linear-ERK$2-2$ is the most accurate. It has the lowest relative error compared to the other schemes of order 2. For the schemes of order 4, 6 and 8 the most accurate ones are respectively Linear-ERK$4-2$, Linear-ERK$6-1$ and Linear-ERK$8-1$.

To recapitulate, low-order schemes have a larger CFL number as compared to high-order schemes having the same number of stages. As an example the tenth-stages second order Linear-ERK$2-8$ provides a stable solution with only $3575$ time-steps while $3910$ time-steps are needed for the tenth-stages fourth order Linear-ERK$4-6$ (see Table 7.11). However the fourth order scheme is more accurate as expected. When the same number of time-steps is used for schemes

| Schemes | Number of time-steps | Number of evaluation | Relative error |
|---|---|---|---|
| Linear-ERK2-0 | 22450 | 44900 | 0.291 |
| Linear-ERK2-1 | 16275 | 48825 | 0.070 |
| Linear-ERK2-2 | 9975 | 39900 | 0.026 |
| Linear-ERK2-4 | 6275 | 37650 | 0.054 |
| Linear-ERK2-6 | 4575 | 36600 | 0.161 |
| Linear-ERK2-8 | 3575 | 35750 | 0.116 |
| Linear-ERK4-0 | 16120 | 64480 | 2.457e-04 |
| Linear-ERK4-1 | 9040 | 45200 | 1.073e-03 |
| Linear-ERK4-2 | 7175 | 43050 | 1.346e-03 |
| Linear-ERK4-4 | 4905 | 39240 | 2.501e-03 |
| Linear-ERK4-6 | 3910 | 39100 | 2.879e-03 |
| Linear-ERK4-8 | 3145 | 37740 | 4.582e-03 |
| Linear-ERK6-0 | 12635 | 75810 | 1.434e-05 |
| Linear-ERK6-1 | 11535 | 80745 | 8.320e-07 |
| Linear-ERK6-2 | 7760 | 62080 | 3.720e-05 |
| Linear-ERK6-3 | 6316 | 56844 | 1.021e-06 |
| Linear-ERK6-4 | 6296 | 62960 | 4.265e-06 |
| Linear-ERK8-0 | 10409 | 83272 | 1.153e-07 |
| Linear-ERK8-1 | 6856 | 61722 | 7.529e-09 |
| Linear-ERK8-2 | 5643 | 56430 | 9.919e-07 |
| Linear-ERK8-4 | 4144 | 49728 | 4.636e-08 |
| Linear-ERK8-6 | 3417 | 47838 | 8.256e-08 |
| Linear-ERK10-0 | 8858 | 88580 | 1.662e-10 |

Table 7.11: Numerical results for the $s + l$-stages Linear-ERK schemes of order **s** near the CFL number (2D case).

| Schemes | Number of time-steps | Relative error |
|---|---|---|
| Linear-ERK2-0 | 200000 | 0.424e-03 |
| Linear-ERK2-1 | 133333 | 9.977e-04 |
| Linear-ERK2-2 | 100000 | 3.032e-05 |
| Linear-ERK2-4 | 66666 | 4.405e-04 |
| Linear-ERK2-6 | 50000 | 1.289e-03 |
| Linear-ERK2-8 | 40000 | 8.608e-04 |
| Linear-ERK4-0 | 100000 | 1.670e-07 |
| Linear-ERK4-1 | 80000 | 1.763e-07 |
| Linear-ERK4-2 | 66666 | 1.819e-07 |
| Linear-ERK4-4 | 50000 | 2.341e-07 |
| Linear-ERK4-6 | 40000 | 2.667e-07 |
| Linear-ERK4-8 | 33333 | 3.696e-07 |
| Linear-ERK6-0 | 66666 | 1.607e-11 |
| Linear-ERK6-1 | 57142 | 2.284e-12 |
| Linear-ERK6-2 | 50000 | 1.003e-11 |
| Linear-ERK6-3 | 44444 | 7.921e-12 |
| Linear-ERK6-4 | 40000 | 6.462e-11 |
| Linear-ERK8-0 | 50000 | 3.456e-13 |
| Linear-ERK8-1 | 44444 | 1.302e-13 |
| Linear-ERK8-2 | 40000 | 1.313e-12 |
| Linear-ERK8-4 | 33333 | 9.011e-13 |
| Linear-ERK8-6 | 28571 | 5.933e-13 |
| Linear-ERK10-0 | 40000 | 9.389e-14 |

Table 7.12: Numerical results for the $s+l$-stages Linear-ERK schemes of order **s** with time-step integrating the complexity (2D case).

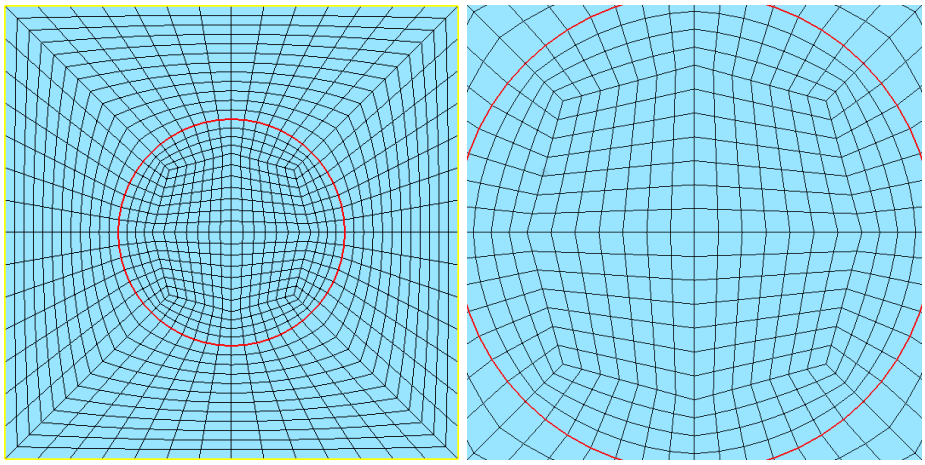


Figure 7.11: Mesh used for the dielectric disk. At right zoom of the mesh in disk.

having a common number of stages, high order schemes are more accurate. For example, in Table 7.12 we can also observe that with the same number of time-steps ($40000$), Linear-ERK$4-6$ has a low relative error ($2.667e-7$) compared to Linear-ERK$2-8$ (only $8.608e-4$). We finally conclude that low order schemes are efficient in terms of stability but they are limited by the accuracy while high order schemes provide accurate results but they are limited by the CFL.

### *7.7.2 Numericals results on a dielectric disk*

Here, the space domain $\Omega$ is represented by a 2D dielectric disk in the middle of a square as shown in the Figure 7.11. The disk delimited by the red circle of radius 2 is an homogeneous media with $\rho = 4,\ \mu = 1$ while the exterior domain is homogeneous with $\rho = \mu = 1$

We use the $10^{th}$ order finite element method as detailed in the chapter 3. We used the first order absorbing boundary condition on around the domain and a Gaussian source function centered at $(-3, 0)$ with the distribution radius equal to 1. The frequency is also equal to 1. The mesh contains 1092 elements with 24 332 degrees of freedom.

The time interval is $I = [0, 40]$. The temporal source corresponds to a Ricker (i.e. the second derivative of a Gaussian function) of central frequency 1. The numerical reference solution has been computed using the eighth order implicit Padé scheme with $\Delta t = 0.001$ (40 000 time-steps). Then we have computed the relative errors for the different schemes at the final time $t = 40$. The time-steps for each scheme are chosen such that the relative error is below $0.01\%$.

The results for the fourth order implicit and explicit schemes are given in the Table 7.13. We have chosen the most efficient Linear-SDIRK scheme of order 4 (see the Chapter 6). First we observe that implicit schemes require more memory compared to explicit schemes, in particular the fourth order Padé

| Schemes | Number of time-steps | Relative error | Memory (GiB) | Computational times |
|---|---|---|---|---|
| Linear-ERK4-0 | 12560 | 3.0241e-07 | 0.046 | 12mn09s |
| Linear-ERK4-1 | 7005 | 1.3515e-06 | 0.046 | 8mn35s |
| Linear-ERK4-2 | 5560 | 1.6936e-06 | 0.046 | 8mn10s |
| Linear-ERK4-4 | 3800 | 3.1624e-06 | 0.046 | 7mn25s |
| Linear-ERK4-6 | 3030 | 3.6537e-06 | 0.046 | 7mn36s |
| Linear-ERK4-8 | 2435 | 5.8630e-06 | 0.046 | 7mn14s |
| LinearSDIRK3-1 | 1500 | 8.9051e-05 | 0.707 | 17mn17s |
| Pade4 | 1920 | 9.2117e-05 | 1.36 | 11mn06s |

Table 7.13: Numerical results for a dielectric disk obtained using explicit and implicit schemes of order 4. The time-steps are chosen such that the relative error is below $0.01\%$ at $t = 40$.

| Schemes | Number of time-steps | Relative error | Memory (GiB) | Computational times |
|---|---|---|---|---|
| Linear-ERK6-0 | 9790 | 2.7717e-09 | 0.052 | 14mn20s |
| Linear-ERK6-1 | 8937 | 1.9552e-08 | 0.052 | 15mn25s |
| Linear-ERK6-2 | 6015 | 5.5684e-11 | 0.052 | 12mn04s |
| Linear-ERK6-3 | 4895 | 7.8021e-11 | 0.052 | 10mn59s |
| Linear-ERK6-4 | 4880 | 3.3101e-10 | 0.052 | 12mn08s |
| LinearSDIRK5-1 | 670 | 8.7647e-05 | 0.713 | 11mn43s |
| Pade6 | 480 | 9.4341e-05 | 2.01 | 4mn50s |

Table 7.14: Numerical results for a dielectric disk obtained using explicit and implicit schemes of order 6. The time steps are chosen such that the relative error is below $0.01\%$ at $t = 40$.

scheme (Pade4) consumes 1.36 gigabyte (GiB) while an explicit scheme requires only 0.046 GiB. Regarding the computational times, we observe that implicit schemes take more time to reach the targeted accuracy compared to optimized explicit schemes.

In Table 7.14, we sketch the performance of the schemes of order 6. We see that the sixth order Padé scheme (Pade6) is the most memory consuming compared to Linear-SDIRk6-1 and the Linear-ERK schemes. However it takes less computational to achieve the targeted accuracy, only 4mn50 against 10mn59s for the best Linear-ERK scheme and 11mn43s for the Linear-SDIRK scheme of order 6.

For the schemes of order eight and ten, we still observe that implicit schemes (Padé and Linear-SDIRK) consume more memory than explicit schemes (Linear-ERK). We also note that implicit schemes, in particular Padé schemes, require less computational time to achieve the desired accuracy (see Tables 7.15 and

| Schemes | Number of time-steps | Relative error | Memory (GiB) | Computational times |
|---|---|---|---|---|
| Linear-ERK8-0 | 8065 | 2.0924e-09 | 0.058 | 16mn03s |
| Linear-ERK8-1 | 5315 | 3.9061e-12 | 0.058 | 12mn00s |
| Linear-ERK8-2 | 4375 | 4.3705e-12 | 0.058 | 10mn40s |
| Linear-ERK8-4 | 3212 | 5.0255e-12 | 0.058 | 9mn30s |
| Linear-ERK8-6 | 2655 | 3.8671e-12 | 0.058 | 9mn09s |
| LinearSDIRK7-2 | 280 | 7.1795e-05 | 0.722 | 7mn58s |
| Pade8 | 230 | 9.1128e-05 | 2.66 | 3mn25s |

Table 7.15: Numerical results for a dielectric disk obtained using explicit and implicit schemes of order 8. The time-steps are chosen such that the relative error is below $0.01\%$ at $t = 40$.

| Schemes | Number of time-steps | Relative error | Memory (GiB) | Computational times |
|---|---|---|---|---|
| Linear-ERK10-0 | 6865 | 3.8989e-12 | 0.064 | 16mn38s |
| LinearSDIRK9-2 | 200 | 4.8549e-05 | 0.728 | 6mn54s |
| Pade10 | 145 | 7.8015e-05 | 3.31 | 3mn11s |

Table 7.16: Numerical results for a dielectric disk obtained using explicit and implicit schemes of order 10. The time-steps are chosen such that the relative error is below $0.01\%$ at $t = 40$.

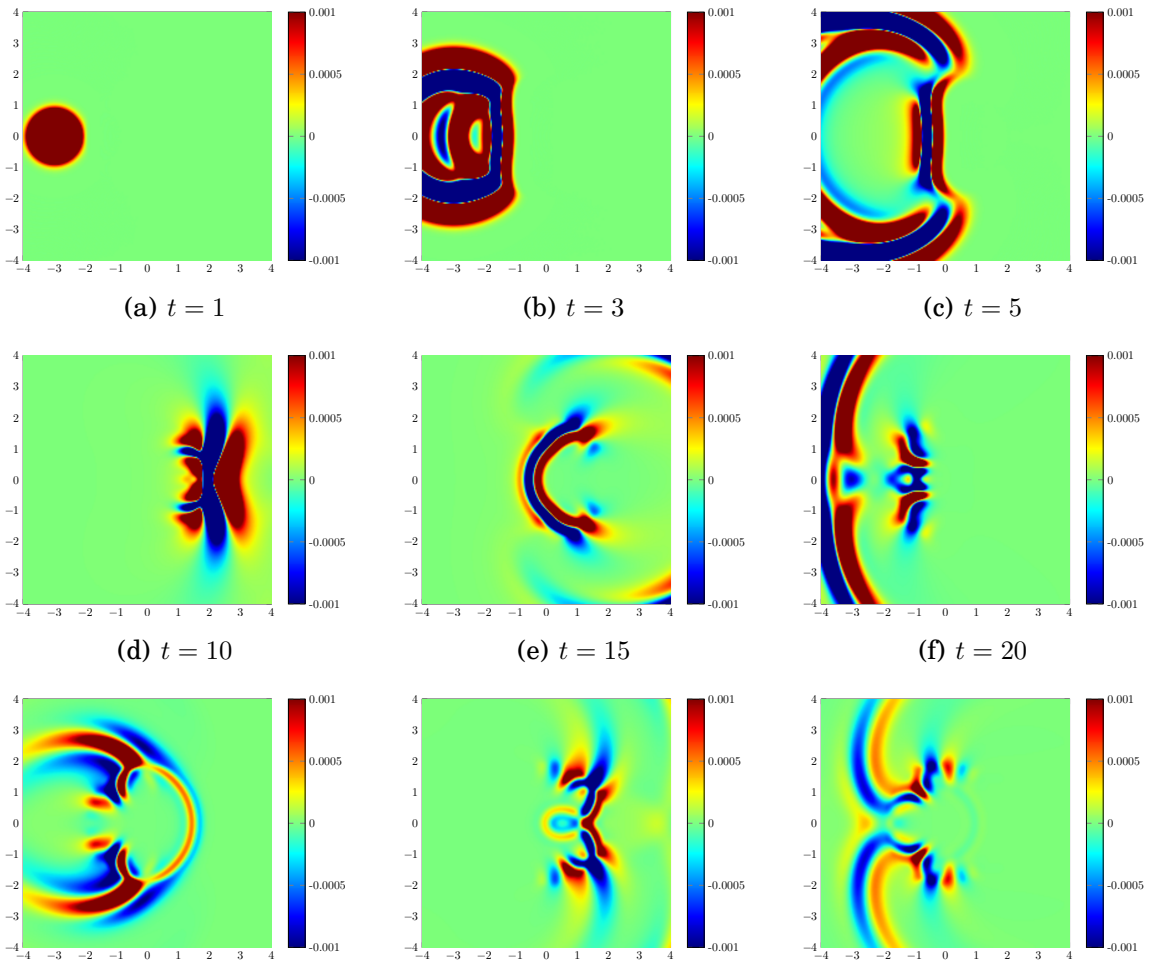
7.16).

In the Figure 7.12, we show the evolution of the numerical solution at different times. We can conclude that high-order implicit schemes, in particular Padé schemes, have better performance in terms of computational times. But they are much more memory consuming compared to the explicit schemes.

## 7.8 Conclusion

In this chapter we have developed ERK schemes with optimal stability function for linear operators having a typical spectrum as the set $Cabane$. We have provided the CFL and evaluated the efficiency of each scheme. The developed schemes have been used to solve the acoustic wave equation with the HDG formulation. We have observed that the schemes of order 2 are not accurate enough. We will not use them in practice.

From the schemes of order 4 up to 8 when we deal with irregular meshes the problem considered becomes stiff, that is to say the time-step chosen at the CFL limit provides a solution accurate enough. For this reason, the schemes with 8, 3 and 6 additional stages should be the most efficient among the presented



(a) $t = 1$ (b) $t = 3$ (c) $t = 5$

(d) $t = 10$ (e) $t = 15$ (f) $t = 20$

(g) $t = 25$ (h) $t = 30$ (i) $t = 40$

Figure 7.12: Solution obtained for the scattering from a dielectric disk at $t = 1, 3, 5, 10, 15, 20, 25, 30$ and the final time $t = 40$.

explicit schemes of order 4, 6 and 8 respectively.

While solving linear ODE with a source term, we have observed that the schemes with numerous additional stages are not always robust. Based on this fact, we will prefer ERK schemes with one and two ($l = 1, 2$) additional stages. These schemes will be used in the next chapter for locally implicit methods. In this chapter, Hörner's algorithm will be used to derive a locally implicit scheme, which is another reason to prefer a reduced number of additional stages.

Chapter 8

# High-order locally implicit time schemes for linear ODEs

In this chapter we construct a method that combines optimized explicit schemes and implicit schemes to form locally implicit schemes for linear ODEs, including in particular wave problems that are the problems of interest in this work.

## Contents

## 8.1 Introduction

Modeling wave propagation in complex domain requires refined meshes or high-order space approximations. Locally implicit time discretization schemes which employ implicit schemes in the part of the mesh recovered with the smallest cells and explicit schemes in the remaining part can be useful. Indeed, when using explicit schemes only, the time-step is restricted by the CFL condition which depends on the size of the smallest elements of the mesh and the order of the space discretization method. We have seen in the Chapters 2 and 7 that the maximal time-step suitable for explicit time schemes becomes smaller and smaller when we refine the mesh or increase the order of approximation.

An alternative to explicit schemes is to use implicit schemes. We have seen that implicit schemes have good stability properties. In particular, the implicit schemes we have presented in the Chapters 5 and 6 are unconditionally stable i.e. there is no stability constraint on those schemes. We have observed that with those schemes the only constraint on the time-step depends on the accuracy we aim for. But for 3D realistic problems, using only implicit methods is not possible since they are extremely memory consuming as we will see in the Chapter 9. There is thus a clear interest in using unconditionally stable schemes in a small region where an explicit scheme should require a very small time-step, but not in the all domain for large 3D like problems.

In many works (e.g. [30, 55, 67, 68]), the authors have developed locally explicit time stepping methods in which they choose different time-steps while solving the problem. The underlying idea consists of dividing the mesh into different zones distinguishing themselves from the size of the mesh elements. By this way, each zone has its own CFL and the time step is chosen zone by zone. This process allows to take small time-steps precisely where the smallest elements are located.

In [29], a second-order locally implicit method is proposed for Maxwell's equations. The construction of the time method is based on the splitting of the meshed domain into a coarse part and a fine part. It is shown in [28] that the component splitting used does not change the second-order convergence of the obtained scheme. In [73] a locally implicit time integration scheme is developed for Maxwell's equations. Its construction is based on the Crank-Nicolson scheme (equivalent to the second order Padé scheme) and the RK implicit midpoint rule (see the Chapter 4). The authors use the DG method (central or upwind fluxes) with locally implicit schemes to discretize the Maxwell's equations and prove the stability of the fully discrete system under a CFL condition and get convergence results of order two.

Following [55] and [29], we propose the construction of locally implicit schemes of arbitrary order for linear ODEs. The general rules for this development are the same as in [55] except that we use implicit schemes in the region recovered with a refined mesh. We validate the developed scheme by solving the acoustic

wave equation with the HDG formulation introduced in the Chapter 3.

## 8.2 Construction of locally time schemes

Let us consider the following ODE

$$y'(t) = Ay(t) + F(t). \tag{8.1}$$

where $A$ is a linear operator and $y$ is the unknown vector. We consider the following splitting

$$y(t) = y^{(c)}(t) + y^{(f)}(t) \tag{8.2}$$

with

$$y^{(c)} = (I - P)y(t), \tag{8.3a}$$
$$y^{(f)} = Py(t), \tag{8.3b}$$

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 the a degree of freedom in the refined mesh and $0$ elsewhere. After discretization we compute $y(t_n + \xi\Delta t)$ as follows

$$\begin{aligned} y(t_n + \xi\Delta t) = y(t_n) + \underbrace{\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt + \overbrace{\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt}^{\text{source term}}}_{\text{Coarse part}} \\ + \underbrace{\int_{t_n}^{t_n+\xi\Delta t} APy(t)dt + \overbrace{\int_{t_n}^{t_n+\xi\Delta t} PF(t)dt}^{\text{source term}}}_{\text{Fine part}} \end{aligned} \tag{8.4}$$

In the coarse part we apply an explicit method while an implicit method is used in the fine part of the mesh.

### 8.2.1 Treatment of the coarse part

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

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{i=1}^{s} b_i\, y(t_n + c_i\xi\Delta t). \tag{8.5}$$

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

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{i=1}^{s} b_i \sum_{j=0}^{r-1} \frac{(c_i\xi\Delta t)^j}{j!} y^{(j)}(t_n). \tag{8.6}$$

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

$$\sum_{i=1}^{s} b_i c_i^j = \frac{1}{j+1}, \quad j = 0, \cdots, r-1 \tag{8.7}$$

we then obtain

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} y^{(j)}(t_n). \tag{8.8}$$

To handle the derivative of $y$ in the previous sum, we differentiate $r-1$ times the ODE (8.1) and replaces it into the equation to obtain

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} \left( A^j y(t_n) + \sum_{\ell=1}^{j} A^{j-\ell} F^{(\ell-1)}(t_n) \right). \tag{8.9}$$

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

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

In order to coincide with explicit optimized schemes presented in the Chapter 7, extra-terms are added to get:

$$\begin{aligned}\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx\ & \xi\Delta t A(I-P)\left( \sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} A^j y(t_n) + \sum_{j=r}^{m} \alpha_{j+1}(\xi\Delta t)^j A^j y(t_n) \right) \\ &+ \xi\Delta t A(I-P)\left( \sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} \sum_{\ell=1}^{j} A^{j-\ell} F^{(\ell-1)}(t_n) + \sum_{j=r}^{m} \alpha_{j+1}(\xi\Delta t)^j \sum_{\ell=1}^{j} A^{j-\ell} F^{(\ell-1)}(t_n) \right)\end{aligned} \tag{8.10}$$

that can be written as

$$\begin{aligned}\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt\ \approx\ & \xi\Delta t A(I-P)\left( \sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j A^j y(t_n) \right) \\ &+ \xi\Delta t A(I-P)\left( \sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j \sum_{\ell=1}^{j} A^{j-\ell} F^{(\ell-1)}(t_n) \right)\end{aligned} \tag{8.11}$$

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, denoted by $Q$, using the quadrature points $c_i$. We use

$$F(t_n + \xi\Delta t) \ \approx \ Q(t_n + \xi\Delta t). \tag{8.12}$$

where the polynomial $Q$ reads

$$Q(t_n + \xi\Delta t) = \widetilde{Q}(\xi) = \sum_{i=1}^{s} F_i\, \widetilde{\varphi}_i(\xi), \tag{8.13}$$

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

$$\widetilde{w}_j \ = \ A^j y(t_n) + \sum_{\ell=1}^{j} A^{j-\ell} Q^{(\ell-1)}(t_n), \tag{8.14}$$

to end up with a simplified expression

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \ \approx \ \xi\Delta t A(I-P)\left(\sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j \widetilde{w}_j\right) \tag{8.15}$$

We use the following relation

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

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

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

In practice, we will compute the vectors

$$\zeta_j = \alpha_{j+1} A(I-P)\widetilde{w}_j. \tag{8.16}$$

An optimal way to do that is provided in the Algorithm 8. Now we treat the

**Algorithm 8** Computation of $\zeta_j$ (8.16)

Coefficients $D_{i,l}$ are precomputed 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**

second-integral in (8.4) corresponding to the source term in the coarse part. It is approximated as follows:

$$\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt \approx (I-P)\xi\Delta t \sum_{i=1}^{s} b_i Q(t_n + c_i \xi \Delta t). \tag{8.17}$$

For degrees of freedom (dofs) that are far from the fine region (i.e. dofs belonging to elements that are not adjacent or inside the fine part of the mesh), we have obtained the following expression for $y_{n+1}$ (obtained for $\xi = 1$):

$$y_{n+1} = y_n + \Delta t A (I-P) \sum_{j=0}^{m} \alpha_{j+1} \Delta t^j \widetilde{w}_j + \Delta t (I-P) \sum_{i=1}^{s} b_i F_i \tag{8.18}$$

We see that this expression is a simple linear combination of the vectors $F_i$ and $\zeta_j$ computed in the Algorithm 8.

**Remark** If the fine part is void (i.e. $P = 0$), the obtained explicit scheme can be written in the usual form given in the Chapter 7:

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

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** In the formula (8.18), we can observe that the coarse part is advanced by using Hörner's algorithm (instead of the stable algorithm). This algorithm is necessary to advance the fine part efficiently. That is the main reason why we preferred the optimized explicit schemes of Chapter 7 with a small number of additional stages, such that the Hörner's algorithm should not deteriorate the accuracy.

### 8.2.2 Treatment of the coarse part combined with the fine part

We recall that $Q$ (8.13) denotes the polynomial that interpolates the source function $F$. We introduce $\hat{Q}$ 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:

$$\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt \approx (I-P)\left(\hat{Q}(t_n+\xi\Delta t)-\hat{Q}(t_n)\right). \tag{8.19}$$

Using the equation (8.19) and the approximation (8.15) in the coarse region, the equation (8.4) 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

$$\begin{aligned} y(t_n+\tau) = y_n + A(I-P)\sum_{j=0}^{m}\alpha_{j+1}\,\tau^{j+1}\widetilde{w}_j + (I-P)\left(\hat{Q}(t_n+\tau)-\hat{Q}(t_n)\right) \\ + \int_{t_n}^{t_n+\tau} APy(t)+PF(t)dt \end{aligned} \tag{8.20}$$

Let $\widetilde{y}$ be defined as

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

Like in [55], we differentiate (8.20) with respect to $\tau$ to obtain the following ODE:

$$\frac{d\widetilde{y}(\tau)}{d\tau} = \overbrace{A(I-P)\sum_{j=0}^{m}(j+1)\alpha_{j+1}\,\tau^{j}\widetilde{w}_j + (I-P)Q(t_n+\tau)+PF(t_n+\tau)}^{\text{updated source term}} + AP\widetilde{y}(\tau) \tag{8.21}$$

The equation (8.21) 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 to approximate $F$ with $Q$ in this term as well. With this approach, we obtain the following ODE:

$$\frac{d\widetilde{y}(\tau)}{d\tau} = AP\widetilde{y}(\tau) + \widetilde{F}(\tau) \tag{8.22}$$

where

$$\widetilde{F}(\tau) = \sum_{j=0}^{m}(j+1)\,\tau^j\,\zeta_j + Q(t_n+\tau) = \sum_{j=0}^{m}(j+1)\,\tau^j\,\zeta_j + \sum_{i=1}^{s}\widetilde{\varphi}_i\left(\frac{\tau}{\Delta t}\right)F_i \tag{8.23}$$

The ODE (8.22) can be solved using either explicit schemes with small time step as in [55] or implicit schemes as we propose here. In this work we have used A-stable implicit schemes we have developed in the Chapters 5 and 6 to solve (8.22), 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 to increase the admissible values of time-step which contributes to reduce the computational costs.

**Remark** The ODE (8.22) is solved for close degrees of freedom, i.e. degrees of freedom that belong to an element inside the fine region or adjacent to the fine region. We use the Algorithm 9 to compute $y_{n+1}$ from $y_n$. In the Algorithm 9, the

**Algorithm 9** Computation of $y_{n+1}$

compute vectors $F_i$ and $\zeta_j$ with algorithm 8
Task 1 : compute $y_{n+1}$ for far degrees of freedom with formula (8.18)
Task 2 : compute $y_{n+1}$ for close degrees of freedom by solving the ODE (8.22) with an implicit scheme

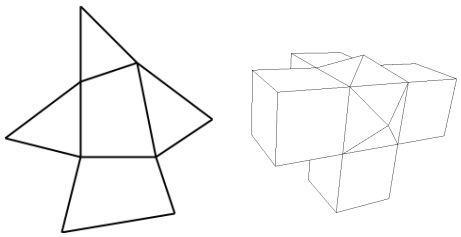
tasks 1 and 2 can be performed in parallel, but the cost of these two tasks are very different since the task 1 consists of a linear combination while task 2 will involve the solution of several linear systems.

**Remark** For HDG formulation, the linear system to be solved involves degrees of freedom associated with the unknown $\lambda$ for edges (faces in 3-D) of the fine region only. Then the unknowns $u$ and $v$ (for acoustics) are reconstructed in the close region (fine region + adjacent elements) as detailed in the chapter (3).

## 8.3 Splitting based on the local time step

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

$$\Omega = \bigcup K_i.$$



(a) 2D (b) 3D

Figure 8.1: Example of 2D and 3D small meshes used to evaluate the local time-step of each element.

We consider the following semi-discrete ODE for a sub-mesh $\Omega_i$ of the computational domain $\Omega$:

$$\begin{cases} M_h \dfrac{dy(t)}{dt} + K_h y(t) = F(t) \quad t \in (0, T] \\ y(0) = y_0 \end{cases} \tag{8.24}$$

The sub-mesh $\Omega_i$ 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 the Figure 8.1, we have represented examples of a sub-mesh $\Omega_i$ in 2D and 3D in the case of an hybrid mesh. $M_h$ is the local mass matrix and $K_h$ is the local stiffness matrix. The ODE (8.24) can be obtained after the discretization of the acoustic or Maxwell's equations (see the Chapter 3). We denote by $A_i = M_h^{-1} K_h$ the product of the inverse of the local mass matrix and the local stiffness matrix of the sub-mesh $\Omega_i$. The local time-step $\Delta t_i$ is then given as

$$\Delta t_i = \frac{\beta}{\rho(A_i)},$$

where $\rho(A_i)$ represents the spectral radius of $A_i$, and $\beta$ an unknown coefficient. $\Delta t_i$ hints on the stiffness nature of the problem in each element of the computational domain. This allows to split the domain into a fine and coarse regions following a simple rule:

$$\begin{cases} \text{if } \Delta t_i \leq \Delta t_{ref},\ K_i \in \Omega^{fine} = \text{ fine region} \\ \text{if } \Delta t_i > \Delta t_{ref},\ K_i \in \Omega^{coarse} = \text{ coarse region} \end{cases} \tag{8.25}$$

where $\Delta t_{ref}$ is the reference time step used to differentiate fine and coarse regions. For a given mesh, discretization method and order, the time-step $\Delta t_i$ locally defined for each element $K_i$ is computed once and stored in a file. This file can be used for other simulations with different time schemes (or different sources) and defines the fine and coarse region.

**Remark** $\Delta t_i$ doesn't represent the true time-step for an explicit scheme, since the coefficient $\beta$ is not known (it depends on the spectrum of $A_i$ and on the coefficients $\alpha_1, \alpha_2, \cdots, \alpha_{m+1}$ of the time scheme). If one wants to compute the true time step $\Delta t_i$, a solution consists of computing all the eigenvalues $\lambda_k$ of $A_h$ (and not only the largest eigenvalues) and of using a bisection method to find the maximal $\Delta t$ such that all the values $z = \Delta t \lambda_k$ are enclosed in the stability domain $\mathcal{S}(\alpha_1, \alpha_2, \cdots, \alpha_{m+1})$ of the considered explicit scheme.

In practice we do not compute the spectral radius for each element to evaluate $1/\rho(A_i)$, because it is expensive and does not give the true time step for a given time integration scheme. We rather use a geometric criterion:

$$h_i = \min_{e \text{ edge of } \Omega_i} \text{length}(e)$$

Then a reference mesh size $h_{\text{ref}}$ is defined to split the domain in two regions:

$$\begin{cases} \text{if } h_i \leq h_{ref},\ K_i \in \Omega^{fine} = \text{ fine region} \\ \text{if } h_i > h_{ref},\ K_i \in \Omega^{coarse} = \text{ coarse region} \end{cases} \tag{8.26}$$

## 8.4 Accuracy of the locally implicit methods

In this section, we want to show 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 locally time stepping method presented in [55].

By construction, the algorithm used to advance far degrees of freedom with formula (8.18) 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 of the stability, but we are rather convinced that the CFL of the global scheme will be controled by the CFL of the coarse part since the ODE of the close region is advanced with an A-stable scheme.

## 8.5 Numerical results

This section provides numerical results in order to validate the locally implicit schemes implemented and to evaluate the efficiency of the schemes. These schemes have been implemented in Montjoie for HDG formulation.

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 solution to the following boundary value problem:

$$\begin{cases} \rho\,\partial_t u - \operatorname{div} v = f, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ \mu^{-1}\partial_t v - \nabla u = 0, \quad \forall (x,t) \in \Omega \times \mathbb{R}^+ \\ u(x,0) = \partial_t u(x,0) = 0, \quad \forall x \in \Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad x \in \Gamma_D \quad \text{(Dirichlet condition)} \\ \mu \partial_n u = f_N, \quad x \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu \partial_n u + \sqrt{\rho\mu}\,\partial_t u = f_A, \quad x \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{8.27}$$

where $\Omega$ is the computational domain. $\Gamma_D$, $\Gamma_N$ and $\Gamma_A$ are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. $n$ is the normal vector outgoing to the considered boundary, $\rho$ and $\mu$ are physical indexes, which are piecewise constant. $f_D$, $f_N$ and $f_A$ are given source functions. The detail of the space discretization of this equation is presented in the Chapter 3.

### 8.5.1 Convergence curves and numerical results on a 2D regular mesh

The wave equation (8.27) is solved in 2D in a homogeneous medium

$$\rho = \mu = 1$$

in the computational domain $\Omega = [-5,5]^2$. We set a homogeneous Neumann condition on the boundaries. The domain $\Omega$ is meshed using quadrilateral elements and is locally refined next to the point $(0,0)$ as shown in the Figure 8.2. For the space discretization, we use $\mathbb{Q}_8$ polynomials and the HDG formulation as presented in the Chapter 3. The space error is about $10^{-11}$. The source term is provided by a gaussian volume source centered at $(0,0)$:

$$f(x,y) = exp(-\alpha_0(x^2+y^2)),\ \alpha_0 \approx -\frac{\log(10^{-6})}{4}$$

We choose $[0,200]$ for the time interval. The temporal source is the Ricker function (the second derivative of a Gaussian) with a central frequency $f_0 = 0.48$. We compute a reference solution using eighth order Padé schemes with $\Delta t = 0.001$. This reference solution is used to compute the $L^2$ relative error.

We compute the numerical solution using the locally implicit method presented previously. We consider two combinations of explicit and implicit schemes. The first one corresponds to a diagonal Padé scheme, developed in the Chapter 5, combined with a Linear-ERK scheme presented in the Chapter 7. The second

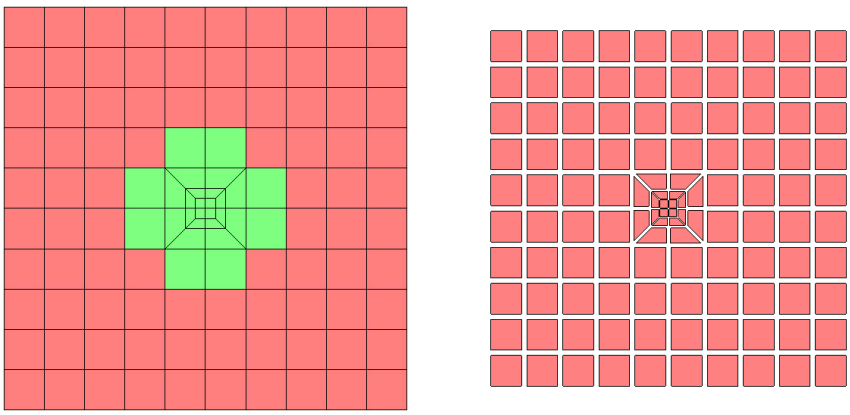


Figure 8.2: Mesh used for the convergence of the Locally implicit scheme (fine region is in green).

one combines a Linear-ERK scheme with a Linear-SDIRK scheme introduced in the Chapter 6. To produce the convergence curves, we compute the numerical solution for different time-steps then we evaluate the relative $L^2$ error between that solution and the reference solution at $t = 200$. In the Figure 8.3 we present the obtained errors with respect to the time-step for the locally implicit methods of order 4, 6 and 8. We observe a good convergence rate for the different schemes.

### *8.5.2 Numerical comparison for the resonant cavity in 2D*

In this test case we consider the scattering of the resonant cavity as described in the subsection 6.5.2. We recall that the computational domain $\Omega$ is meshed with quadrilaterals (see Figure 8.4). The external boundary is a circle of radius 1.5. The 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

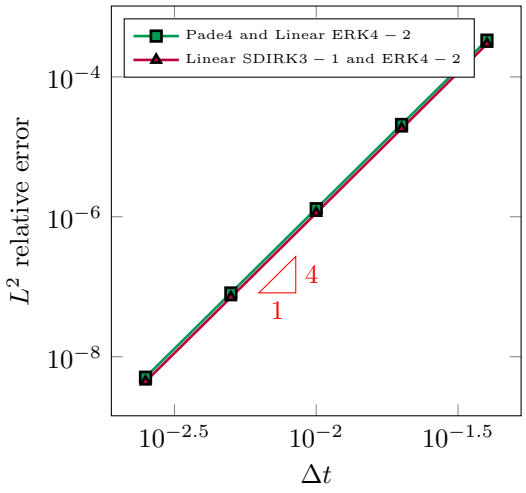
$$\rho = \begin{cases} 0.8 \text{ if } \sqrt{x^2+y^2} \leq 1.25 \\ 1.0 \text{ otherwise} \end{cases}, \quad \mu = \begin{cases} 1.2 \text{ if } \sqrt{x^2+y^2} \leq 1.25 \\ 1.0 \text{ otherwise} \end{cases}$$

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):

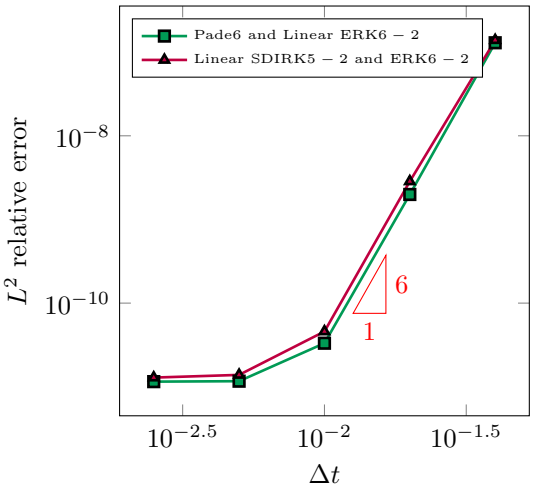
$$\mu \partial_n u + \sqrt{\rho\mu}\, \partial_t u = \mu \partial_n u^{\text{inc}} + \sqrt{\rho\mu}\, \partial_t u^{\text{inc}}, \quad \text{on } \Gamma_A$$

where the incident field is given by

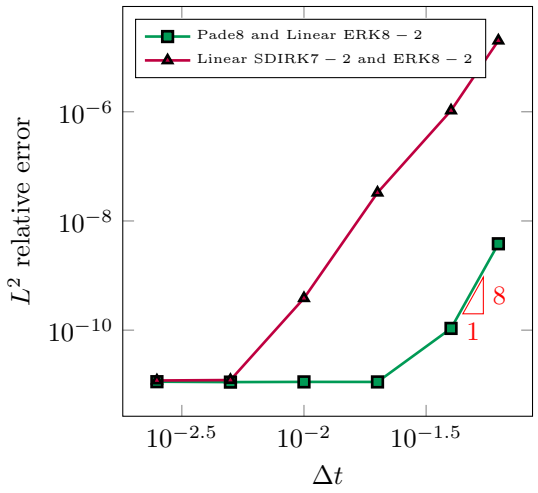
$$u^{\text{inc}} = h(t - 1.5 - x)$$




(a) Order 4




(b) order 6




(c) Order 8

Figure 8.3: Relative $L^2$ error between numerical solution and the reference solution for $t = 200$ with respect to the time step. Comparison of locally implicit methods of order 4 6 and 8 (Padé and Linear-SDIRK schemes combined with Linear-ERK schemes). The space discretization error is about $10^{-11}$.

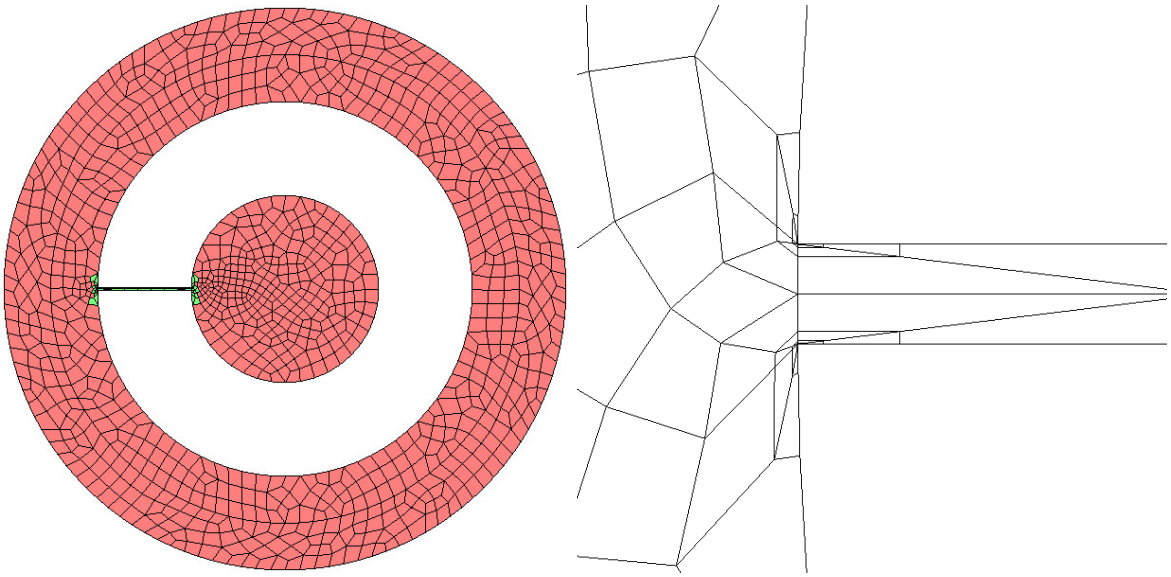


Figure 8.4: Mesh used for the resonant cavity (fine region is in green). On the right, detail of the mesh close to the entry of the cavity.

where

$$h(t) = \sin(\omega t) e^{-b(t-T)^2}$$

with

$$\omega = 16\pi,\ b = 4,\ T = \sqrt{\log(10^6)}.$$

For the space discretization, we use $\mathbb{Q}_{10}$ polynomials with the HDG formulation. The numerical solutions are computed for a time interval $I = [0, 10]$. The temporal source is a Gaussian with a parameter $\alpha = 4$ modulated by a sinus function ($F(t) = \mathrm{e}^{-\alpha(t-t_0)} \sin(2\pi f_0 t)$) at the frequency $f_0 = 8$. In Tables 8.1 and 8.2, we provide the computational times for the locally implicit methods and the fully implicit schemes. The $L^2$ relative errors are evaluated at the time $t = 10$ with a reference solution computed with the eighth order Padé scheme and $\Delta t = 5e-04$ (20 000 time-steps). We observe that due to the presence of a local refinement the time-step for the fully explicit scheme is severely constrained by the CFL number. For example, when using the eighth order Linear-ERK scheme with two additional stages (l=2) the maximal time-step is about $5e-07$ (which corresponds to 20 000 000 time-steps for the overall simulation). By considering the locally implicit method technique, the global number of time-steps for the corresponding scheme goes down to 20 000. This technique allows to get a stable solution with a lower cost and with less resource memory. However, by looking at the Table 8.2, we observe that the fully implicit scheme provides results accurate enough for less computational time. This test case shows an insight on the feature provided by locally implicit schemes to considerably release the time-step restriction and save memory.

| Order | Number of time-steps | Relative errors | Computational times |
|---|---|---|---|
| Linear SDIRK and ERK schemes with $l = 2$ | | | |
| 4 | 25000 | 1.6426e-07 | 2h14mn30s |
| 6 | 25000 | 3.1255e-09 | 3h21mn18s |
| 8 | 20000 | 3.5490e-07 | 3h28mn57s |
| Padé and Linear-ERK schemes with $l = 2$ | | | |
| 4 | 25000 | 1.0233e-07 | 1h56mn03s |
| 6 | 25000 | 8.8443e-13 | 2h40mn37s |
| 8 | 20000 | 8.3829e-12 | 2h49mn38s |

Table 8.1: Computational times for the locally implicit method for the scattering of the resonant cavity (2D case HDG Acoustic).

| Order | Number of time-steps | Relative errors | Computational times |
|---|---|---|---|
| Linear SDIRK schemes | | | |
| 4 | 7500 | 2.5100e-06 | 1h51mn47s |
| 6 | 2200 | 8.8877e-08 | 58mn59s |
| 8 | 1000 | 2.4961e-07 | 34mn24s |
| Padé schemes | | | |
| 4 | 8000 | 7.6997e-06 | 1h06mn54s |
| 6 | 2500 | 2.4095e-07 | 31mn39s |
| 8 | 1000 | 6.1501e-08 | 17mn42s |

Table 8.2: Computational times for the implicit method for the scattering of the resonant cavity (2D case HDG Acoustic).

# Part III

# 3D Numerical results

Chapter 9

# Numerical comparison results

This chapter focuses on the solution of 3D test cases while solving the acoustic wave equation and Maxwell's equations. The first test case corresponds to the scattering of spherical inclusions in a slab. The second one describes the scattering of a spherical resonant cavity.

## Contents

## 9.1 Introduction

We consider the acoustic and Maxwell's equations introduced in the Chapter 3. The implicit and explicit time integration schemes developed in the Chapters 5, 6 and 7 are combined with the HDG formulation to solve both equations. For each time scheme, we provide the computational resource needed to perform the simulation. Concerning the implicit schemes, we give also the resource memory used to factorize the condensed linear systems to be solved, i.e. the memory space used to store the factorization of the matrix $A_h$ introduced in the Chapter 3. This matrix involves only degrees of freedom associated with the unknown $\lambda$. The memory space used to factorize this matrix $A_h$ is specified as "Factorization". We have also displayed the resource memory space used to store the condensation blocks (i.e blocks $A_{12}, A_{21}, A_{22}^{-1}$ as introduced in the Chapter 3), this memory is specified as "Condensation Solver". For hexahedral elements, only blocks $\widetilde{B}$ are stored (see Chapter 3).

The numerical experiments are conducted in Federative Platform for Research in Computer Science and Mathematics known as PlaFRIM. The simulations are launched in nodes Miriel, each node having the following features:

- ✓ 2 Dodeca-core Haswell Intel Xeon E5-2680
- ✓ 2.5 GHz
- ✓ 24 cores
- ✓ RAM 128 GiB

For implicit time integration schemes, we have used the direct solver MUMPS [2] (version 5.0.1) to solve the linear systems. The computational time displayed is the maximal time spent in a single core (among all cores), it corresponds to the real time we had to wait to obtain the solution. The number of degrees freedom displayed is the number of degrees of freedom for the unknown $\lambda$ in HDG formulation.

## 9.2 Spherical inclusion

In this section we are interested in solving the time domain Maxwell's equations reading as:

$$\begin{cases} \mu\dfrac{\partial \mathbf{H}}{\partial t} + \mathbf{curl}\ \mathbf{E} = 0, \quad \forall (x,t) \in \Omega \times I \\ \varepsilon\dfrac{\partial \mathbf{E}}{\partial t} + \widetilde{\sigma}\mathbf{E} - \mathbf{curl}\ \mathbf{H} = 0, \quad \forall (x,t) \in \Omega \times I \\ \mathbf{E}(x,0) = \mathbf{H}(x,0) = 0, \quad \forall x \in \Omega, \quad \text{(initial conditions)} \\ \mathbf{n} \times (\mathbf{E} \times \mathbf{n}) = f_D, \quad \forall (x,t) \in \Gamma_D \times I, \quad \text{(Dirichlet condition)} \\ \mathbf{n} \times \mathbf{H} + \sqrt{\dfrac{\varepsilon}{\mu}}\,(\mathbf{n} \times \mathbf{E}) \times \mathbf{n} = 0, \quad \forall (x,t) \in \Gamma_A \times I, \quad \text{(Silver-Müller condition)} \end{cases} \tag{9.1}$$

where $\Omega$ is a bounded domain of $\mathbb{R}^3$ whose boundary $\partial\Omega$ is composed of $\Gamma_D$ and $\Gamma_A$ with $\Gamma_D \cap \Gamma_A = \emptyset$. The computational domain $\Omega$ is defined by a slab $[-8,8] \times [-8,8] \times [-1,3]$ containing 40 spherical inclusions, each inclusion having a radius of 0.1.

The domain is meshed using hexahedral elements as represented in the Figure 9.1. For the permittivity, the permeability and the conductivity we take the following dimensionless values:

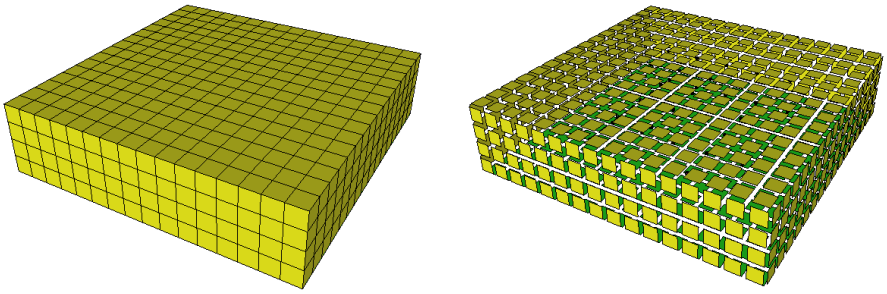
$$\epsilon = \begin{cases} 4.0 & \text{if inside an inclusion} \\ 1.0 & \text{otherwise} \end{cases}, \quad \mu = \begin{cases} 2.0 & \text{if inside an inclusion} \\ 1.0 & \text{otherwise} \end{cases},$$

and

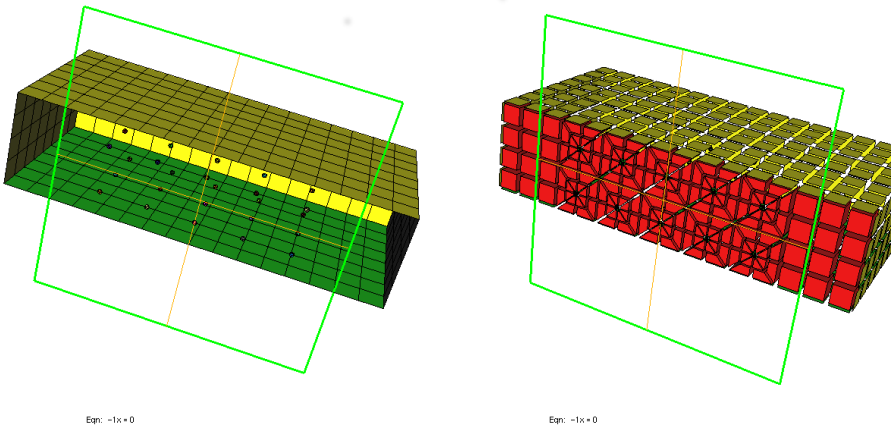
$$\sigma = \begin{cases} 0.1 & \text{if inside an inclusion} \\ 0 & \text{otherwise} \end{cases}$$

An inhomogeneous Dirichlet condition is set on the bottom surface of the domain ($\Gamma_D$, green surface in the Figure 9.1) with Gaussian source in space centered at (0,0,-1), Ricker source in time at central frequency $f_0 = 1$. On the other boundary surfaces ($\Gamma_A$, surface in yellow) the Silver-Müller condition is set. We are using $\mathbb{Q}_5$ polynomial functions and the HDG formulation as detailed in the Chapter 3, on the mesh is depicted in the Figure 9.1.

In the Table 9.1, we provide the computational resources needed to compute the numerical solution for both explicit and implicit schemes. As expected, we observe that implicit schemes require much more memory space than explicit schemes. In particular, the Padé schemes are the most expensive in terms of memory consumption. This is essentially due to the factorization process when we solve the linear system as detailed in the Section 5.2. The Linear-SDIRK schemes require only one factorization while for Padé schemes of order $2m$, $m \in \mathbb{N}^*$ we need $m$ factorizations. For example we see in Table 9.1 that the memory space used in the factorization step for the eighth-order Padé scheme (Pade8) is two times bigger than that used for the fourth-order Padé scheme (Pade4).



(a) Mesh



(b) Mesh spherical inclusion (Cut)

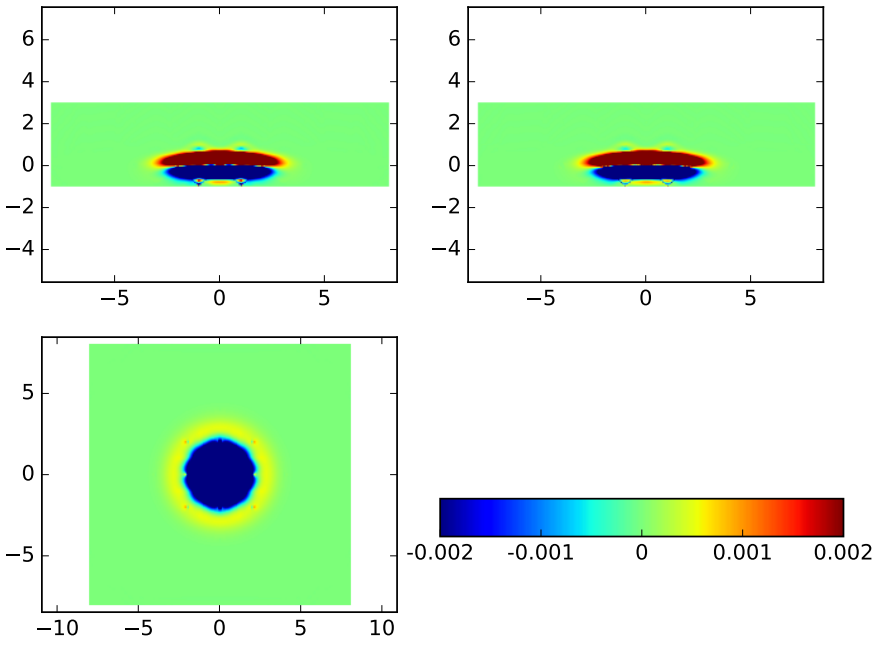
Figure 9.1: Mesh used for the scattering of spherical inclusions in a slab.

| $\mathbb{Q}_5$ | 4 624 hexahedra | | 1 026 432 degrees of freedom | |
|---|---|---|---|---|
| Schemes | Global Memory (GiB) | Factorization (GiB) | Condensation Solver (GiB) | Time vector (MiB) |
| LinearERK$4-2$ | 0.704 | - | - | 274.34 |
| LinearERK$6-2$ | 1.13 | - | - | 365.94 |
| LinearERK$8-2$ | 1.21 | - | - | 457.46 |
| LinearSDIRK$3-1$ | 29.43 | 20.1 | 8.2 | 365.89 |
| LinearSDIRK$5-2$ | 29.57 | 20.1 | 8.2 | 503.16 |
| LinearSDIRK$7-2$ | 29.66 | 20.1 | 8.2 | 594.69 |
| Pade4 | 57.41 | 40.15 | 15.99 | 502.97 |
| Pade6 | 85.76 | 60.25 | 24.19 | 548.70 |
| Pade8 | 113.65 | 80.30 | 31.98 | 594.43 |

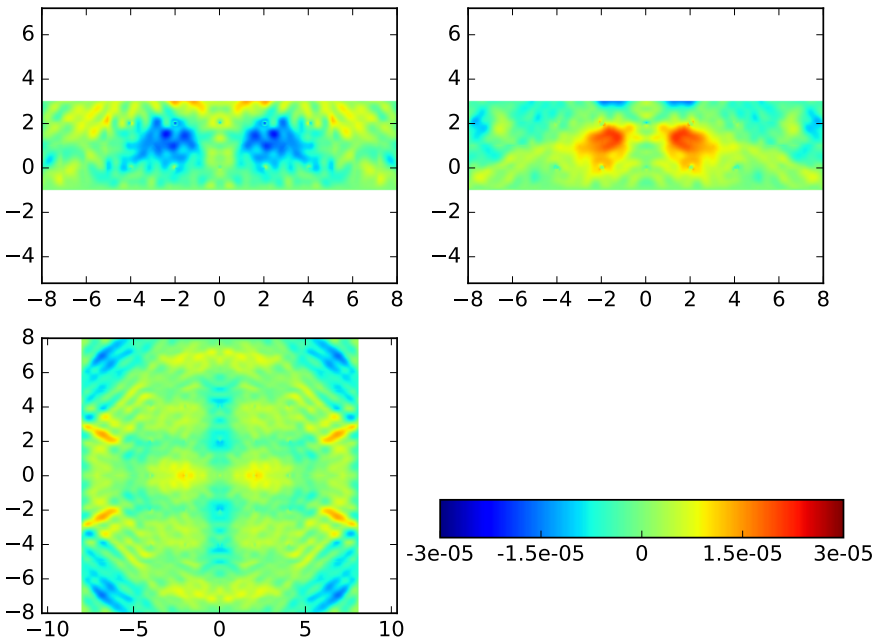
Table 9.1: Computational memory space needed for explicit and implicit schemes for the scattering of spherical inclusions (3D case HDG Maxwell). A direct solver is used for implicit schemes.

In fact, for Pade4 the factorization is performed for one complex root while it is done for two complex roots for Pade8 (see the Section 5.2). In the case of Pade6, the factorization is performed for one complex root and one real root which corresponds exactly to the memory space consumed for one Linear-SDIRK scheme and Pade4 during the factorization step.

To evaluate the computational time of each scheme, we consider the time interval $I = [0, 20]$. The temporal source is the Ricker function (the second derivative of a Gaussian function) at central frequency $f_0 = 1$. In Table 9.2 we present the computational time required by the different schemes to obtain a relative $L^2$ error below $0.001\%$ at $t = 20$. The reference solution is computed with $\Delta t = 0.005$ (4 000 time-steps) and the eighth order Padé scheme. The simulations are performed using the experimental platform PlaFRIM. We have used six nodes Miriel, each node having twenty four (24) cores. The relative errors are computed using the x-component of the electric field. We observe in the Table 9.2 that explicit schemes are more efficient than implicit schemes. They require less computational time to achieve the targeted accuracy. Among implicit schemes, Padé schemes take less computational time to reach the targeted accuracy compared to Linear-SDIRK schemes, even though they require much more memory space. The x-component of the electric field is shown on the Figure 9.2 at $t = 2$ and $t = 20$. Here the spacial discretization is rather coarse, that is why the displayed solution exhibits some numerical artifacts.




(a) $t = 2$




(b) $t = 20$

Figure 9.2: x-component of the electric field at $t = 2$ and $t = 20$ on three planes Oyz, Oxz and Oxy with $O = (0, 0, -0.1)$.

| $\mathbb{Q}_5$ | 4 624 hexahedra | 1 026 432 degrees of freedom | |
|---|---|---|---|
| Schemes | Number of time-steps | Relative errors | Computational times |
| Linear-ERK$4-2$ | 4000 | 6.1918e-06 | 1mn08s |
| Linear-ERK$6-2$ | 4000 | 3.4500e-09 | 1mn29s |
| Linear-ERK$8-2$ | 2500 | 6.3281e-10 | 1mn11s |
| Linear-SDIRK$3-1$ | 2300 | 7.4326e-06 | 41mn60s |
| Linear-SDIRK$5-2$ | 900 | 4.3470e-06 | 30mn07s |
| Linear-SDIRK$7-2$ | 500 | 5.5613e-06 | 20mn09s |
| Pade4 | 3500 | 7.9893e-06 | 43mn14s |
| Pade6 | 1000 | 6.0541e-06 | 21mn51s |
| Pade8 | 500 | 4.2474e-06 | 16mn04s |

Table 9.2: Computational times for explicit and implicit schemes for the scattering of spherical inclusions (3D case HDG Maxwell). Times computed when using 6 nodes Miriel and 144 cores on the PlaFRIM platform. A direct solver is used for implicit schemes. The relative $L^2$ error is bellow $0.001\%$.

## 9.3 Spherical resonant cavity

We consider the acoustic wave equation with Neumann and absorbing boundary conditions:

$$\begin{cases} \rho\,\partial_t u - \mathbf{div}\,\mathbf{v} = 0, \quad \forall (\mathbf{x},t) \in \Omega \times I \\ \mu^{-1}\partial_t \mathbf{v} - \nabla u = 0, \quad \forall (\mathbf{x},t) \in \Omega \times I \\ u(\mathbf{x},0) = \partial_t u(\mathbf{x},0) = 0, \quad \forall \mathbf{x} \in \Omega \quad \text{(null initial conditions)} \\ \mu \partial_n u = f_N, \quad \mathbf{x} \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu \partial_n u + \sqrt{\rho\mu}\,\partial_t u = 0, \quad \mathbf{x} \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{9.2}$$

In this experiment the 3D wave equation (9.2) is solved in a homogeneous medium

$$\rho = \mu = 1$$

in a domain $\Omega$ designing a spherical resonant cavity. $\Omega$ is meshed with hybrid elements (tetrahedra, hexahedra, pyramids and wedges) as shown in the Figure 9.3. The external boundary is a cube of size $[-0.75, 0.75]^3$. The internal boundary is a sphere of radius $0.5$, the cavity is a sphere of radius $0.35$. The spherical cavity is linked with the external domain (the cube) by a cylinder of radius $0.03$ as displayed in the Figure 9.3(b). We can observe that the mesh is refined in the cylinder and around the cylinder. An inhomogeneous Neumann condition is set on the internal boundary to simulate the scattering of a plane wave:

$$\frac{\partial u}{\partial n} = -\frac{\partial u^{\text{inc}}}{\partial n}$$

| $\mathbb{Q}_4$ | 24 885 elements | | 1 633 475 degrees of freedom | |
|---|---|---|---|---|
| Hybrid mesh | 3423 tetra, 16571 hexa, 4246 pyramids, 645 wedges | | | |
| Schemes | Global Memory (GiB) | Factorization (GiB) | Condensation Solver (GiB) | Time vector (MiB) |
| LinearERK$4-2$ | 1.1 | - | - | 452.94 |
| LinearERK$6-2$ | 1.25 | - | - | 603.95 |
| LinearERK$8-2$ | 1.40 | - | - | 754.98 |
| LinearSDIRK$3-1$ | 35.04 | 29.38 | 4.41 | 603.90 |
| LinearSDIRK$5-2$ | 35.27 | 29.38 | 4.41 | 830.43 |
| LinearSDIRK$7-2$ | 35.41 | 29.38 | 4.41 | 981.47 |
| Pade4 | 68.86 | 58.67 | 8.71 | 830.24 |
| Pade6 | 102.73 | 88.05 | 13.12 | 905.72 |
| Pade8 | 136.40 | 117.34 | 17.43 | 981.21 |

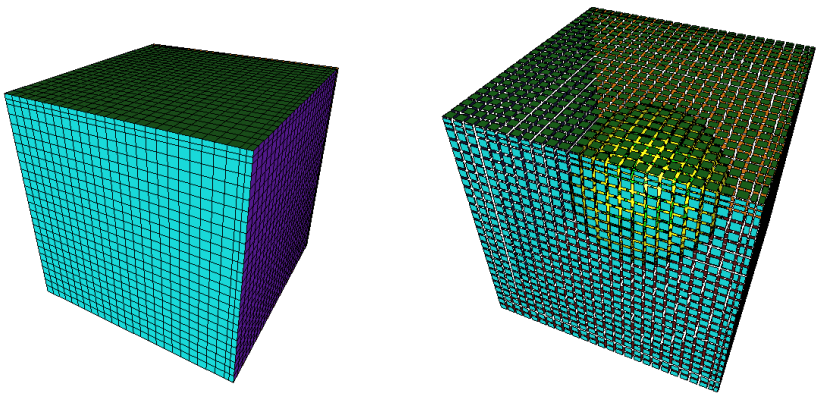
Table 9.3: Computational resources needed for explicit and implicit schemes for the scattering of spherical resonant (3D case HDG Acoustic). A direct solver is used for implicit schemes.

where the incident field $u^{\text{inc}}$ is given as

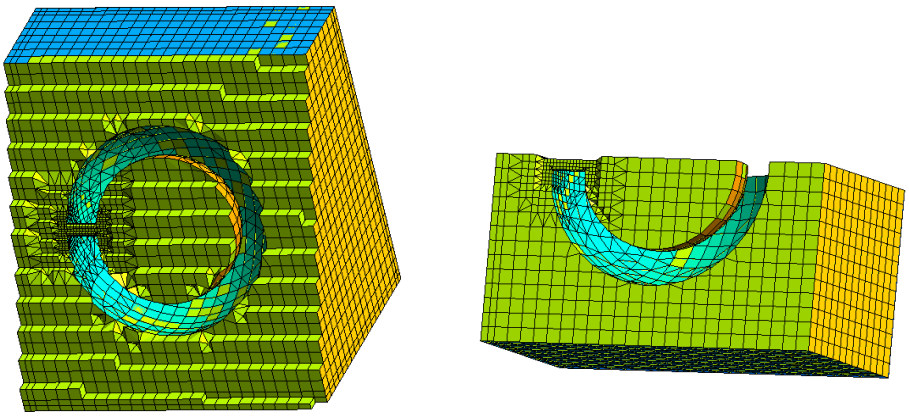
$$u^{\text{inc}} = h(t - 0.75 - x)$$

where $h$ is the temporal source. A homogeneous absorbing boundary condition is set on the external boundary (the six surfaces of the cube). We are using $\mathbb{Q}_4$ polynomials and the HDG formulation as detailed in the Chapter 3 for the space discretization. The solutions are computed for the time interval $I = [0, 6]$. The temporal source is a Gaussian with parameter $\alpha = 16$ modulated by a sinus function ($h(t) = \mathrm{e}^{\alpha(t-t_0)} \sin(2\pi f_0)$) at frequency $f_0 = 10$. In Table 9.3 we present the computational memory space consumed by explicit and implicit schemes to perform the simulation. The explicit schemes are less memory space consuming compared to implicit schemes. Padé schemes are the most expensive in terms of memory space usage, in particular due to the factorization step.

In Table 9.4, we provide the computational time taken by each scheme to complete the overall simulation. For implicit schemes, we target the same accuracy as for the explicit schemes. The time-steps for explicit scheme are chosen according to the limit of their CFL number. The relative $L^2$ errors are evaluated at $t = 6$ using a reference solution computed with the eighth-order Padé scheme and the time-step $\Delta t = 0.001$ (6000 time-steps). We observe that Padé schemes takes less computational time compared to the Linear-SDIRK and Linear-ERK schemes. They require fewer number of time-steps to achieve the targeted accuracy which made them efficient even though the memory space used is larger. In the Figure 9.4, we display the numerical solution at $t = 1, 2, 4$ and $6$. Here we did not use curved elements that is why we observe a loss of symmetry of the



(a) Mesh



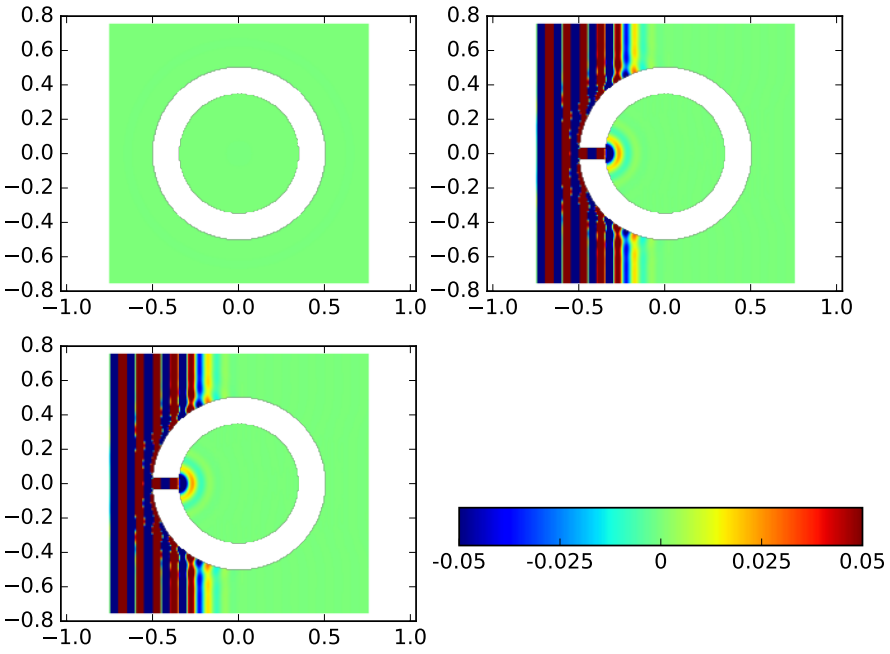
(b) Mesh (Cut)

Figure 9.3: Mesh used for a scattering of the spherical resonant cavity. On the top we show the full domain and a view on the spherical domain (obtained with MEdit software). At the bottom we represent a cross-section of the computational domain (green elements) and the refined region (obtained with Gmsh software).

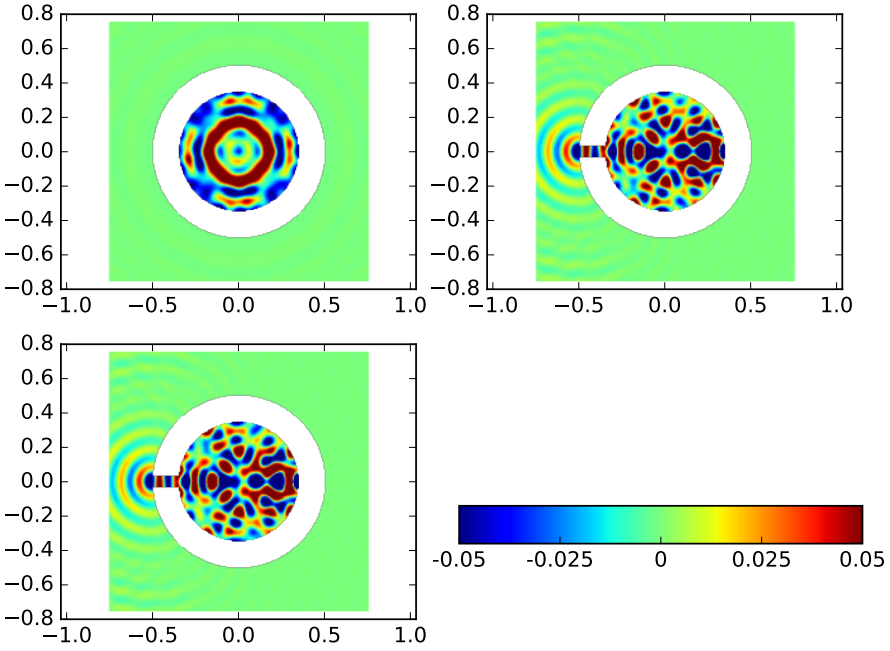
| $\mathbb{Q}_4$ | 24 885 elements | 1 633 475 degrees of freedom | |
|---|---|---|---|
| Hybrid mesh | 3423 tetra, 16571 hexa, 4246 pyramids, 645 wedges | | |
| Schemes | Number of time-steps | Relative errors | Computational times |
| Linear-ERK$4-2$ | 24000 | 3.9784e-04 | 1h08mn25s |
| Linear-ERK$6-2$ | 24000 | 6.195e-03 | 1h13mn14s |
| Linear-ERK$8-2$ | 15000 | 1.9971e-05 | 1h16mn16s |
| Linear-SDIRK$3-1$ | 3000 | 1.0070e-03 | 1h19mn30s |
| Linear-SDIRK$5-2$ | 1920 | 1.0388e-03 | 1h31mn05s |
| Linear-SDIRK$7-2$ | 900 | 1.0881e-03 | 44mn03s |
| Pade4 | 3000 | 4.2042e-04 | 40mn31s |
| Pade6 | 2000 | 4.0110e-04 | 47mn57s |
| Pade8 | 750 | 7.7757e-04 | 20mn08s |

Table 9.4: Computational times for explicit and implicit schemes for the scattering of spherical resonant (3D case HDG Acoustic), computed when using 6 nodes Miriel and 144 cores on the PlaFRIM platform. A direct solver is used for implicit schemes.

numerical solution.




(a) $t = 1$




(b) $t = 6$

Figure 9.4: Solution obtained for the scattering of a spherical resonant cavity at $t = 1$ and $t = 6$ (total field).

Chapter 10

# Conclusion

In this thesis, we have studied and developed different families of time integration schemes for linear ODEs, focusing on the solution of acoustic and electromagnetic wave propagation problems. However, it is worth mentioning that these schemes can be used for any linear ODE with constant coefficients obtained from the spatial discretization of a boundary value problem based upon a Partial Differential equation.

First in the Chapter 2 we have introduced definitions and properties used for the development of the time integration schemes. This chapter aimed at stating the type of ODEs considered and to clarify the stability, dissipation and dispersion properties of time integration schemes.

Then, in the Chapter 3, we have introduced the wave propagation problems considered and the methods used to discretize them in space. Considering the first order formulation of the acoustic wave equation, we have used the mixed spectral finite element method to provide a semi-discrete formulation. The technique of mass lumping has been used to obtained diagonal mass matrix when quadrilaterals or hexahedra elements are used. Since for other types of elements the mass matrix is not diagonal anymore, we have also considered the HDG formulation which leads to block diagonal mass matrix. For both types of space discretization methods, the technique of static condensation has been detailed in order to remove internal degrees of freedom and reduce the size of the linear system to solve for implicit time schemes. Plus the classical formulation with absorbing, Dirichlet and Neumann boundary conditions, the variational formulation integrating PMLs (2D and 3D) is given for the acoustic and Maxwell's equations. The linear system to be solved while considering an implicit scheme is provided and the static condensation based on the computation of the Schur complement is used to reduce the computational burden.

In the Chapter 4, we have presented a review of high-order Runge-Kutta (RK) schemes commonly used in the literature. We have presented the stabil-

ity, dispersion and dissipation properties of the explicit RK schemes of order 2, 3, 4, 6 and 8 and we have seen that all of them have a bounded stability domain. As a consequence, the explicit schemes have a time-step restriction involving the so-called CFL condition. Since the problems we are considering can be stiff, especially when we are using high order polynomials for the space approximation or if the computational domain contains a local refinement which is likely, we have investigated implicit RK schemes which are A-stable (unconditionally stable). From this investigation two families of implicit schemes have been considered. The first ones which are Gauss like RK schemes are fully implicit and require much resource of memory space. The second ones, the so-called Singly Diagonally Implicit RK (SDIRK) schemes, have been preferred. However, A-stable SDIRK are not common beyond an order 5 of accuracy. We have provided the coefficients for these SDIRK schemes up to order 5. To obtain higher-order schemes, we attempted to construct SDIRK schemes from A-stable stability function. We did not success in this process due to numerous order conditions plus the A-stability condition and the particular properties of SDIRK schemes. For this reason, our work has focused on the development of A-stable time integration schemes for linear ODEs introduced in the second part of this thesis.

The second part of the manuscript which consists of our main contribution, has been developed with two main objectives. The first was the construction of A-stable time integration schemes for an arbitrary order with low-dissipation and low-dispersion effects. This task has been completed by proposing two families of implicit schemes: the diagonal Padé schemes and the SDIRK schemes for linear ODEs only. The diagonal Padé schemes have been constructed using the Padé approximation of an exponential function in the Chapter 5. It has been proved that they are A-stable and non-dissipative when considering the test equation $y' = i\lambda y$. We also proved that these diagonal Padé schemes are equivalent to Gauss-RK schemes for linear ODEs. The advantage in using the diagonal Padé schemes instead of Gauss-RK lies in the fact that with Padé schemes we solve many smaller linear systems to compute the solution at one step while with Gauss-RK schemes we have a very large linear system to solve. We have proposed an optimal algorithm to take advantage of the algebraic properties of the stability function of these schemes in order to significantly reduce the computational burden. In Chapter 6, we have proposed a methodology to construct A-stable SDIRK schemes for an arbitrary order. We have provided the stability of these schemes up to order 12. We have also provided an algorithm to efficiently implement them in particular when solving linear ODEs with source term. This work on the development of A-stable high-order implicit schemes has been recently accepted for publication (see [5]).

The second objective of this part, was the development of explicit schemes with an optimal CFL number and a locally implicit method. To complete this

task, we have considered a polynomial stability function based on the Taylor series expansion of an exponential function. Then, we have added extra terms beyond the terms of the Taylor expansion without changing the order of accuracy. The coefficients of those extra terms have been computed by optimizing the CFL number such that the stability region of the developed scheme include a typical profile of spectrum of a linear operator obtained from the HDG discretization. This has been done in the Chapter 7. We have also determined the CFL number and the efficiency on the typical profile for each explicit scheme. Pursuing our aim, we have proposed a methodology to construct locally implicit methods of arbitrary order for linear ODEs in the Chapter 8. We have presented the locally implicit methods obtained from the combination of the A-stable implicit schemes we have developed and explicit schemes with optimal CFL number. The method has been used to solve the acoustic wave equation and we have provided convergence curves demonstrating the performance of schemes of order 4, 6 and 8.

Plus the different 1D and 2D validation tests performed in the Chapters 4, 5, 6 and 7 while solving the acoustic wave equation, we have performed numerical simulation for 3D acoustic wave and Maxwell's equations that are given in the Chapter 9. The first test case was about the electromagnetic wave scattering from spherical inclusions in a slab and the second one deals with the acoustic wave scattering from a spherical resonant cavity. For both cases, we have used the HDG formulation and we have compared the explicit and implicit time integration schemes.

As a perspective the first goal will be to finalize our work on the construction of the locally implicit methods. This will consist of performing simulations in 3D and show the feature proposed by this kind of technique. We can also combine the locally implicit technique with the local time stepping for further optimization.

The algorithm used to split the computational domain into fine and coarse region can be improved. Instead of using only a geometric criteria, we can think on how to compute the true CFL number, local to each element, which involves the order of approximation as well. For a small problem, one can compute all the eigenvalues of the linear operator associated with an element and its neighbors. But for 3-D large problems and with a high-order approximation, this process becomes extremely expensive.

In the theoretical view, it might be interesting to investigate the stability and the convergence order of the obtained locally implicit schemes. We believe that the CFL number of the locally implicit schemes, as we have developed, corresponds to the CFL number of the explict schemes used. However, we did not give any theoretical proof on that.

Another point that can be interesting to look at, is the dispersion and dis-

sipation order. In fact, for the different explicit and implicit schemes, we have only provided validation curves to show the dissipation and dispersion errors we may introduce when we take a large time-step. To obtain further informations, it will be interesting to provide the dissipation and dispersion order analyticaly. This can be done, for example, by considering the Taylor expansion of the dissipation and dispersion formula (see the Chapter 2) for a given stability function. For higher-order schemes, it is more and more complicate to express the dissipation and dispersion order.

# Conlusion Générale

Dans cette thèse, nous avons étudié et développé différentes familles de schémas d'intégration en temps pour les EDO linéaires. Ces schémas ont été utilisés pour résoudre des problèmes de propagations d'ondes acoustiques et électromagnétiques en dimension un, deux et trois. Cela dit, ils sont tout à fait utilisables pour d'autres type de problèmes nécessitant la résolution d'EDO linéaires.

Après avoir introduit les définitions et propriétés utilisées pour construire les schémas en temps, nous avons présenté les méthodes utilisées pour discrétiser les équations d'ondes acoustiques et électromagnétiques en espace. Nous avons tout d'abord décrit une méthode des éléments finis spectraux dans laquelle on utilise la condensation de masse pour obtenir une matrice de masse diagonale (avec des quadrilatères et hexaèdres). Puis nous avons décrit une formulation HDG (méthode Galerkin discontinue hybride) qui conduit à une matrice de masse bloc-diagonale. Dans les deux cas de discrétisations spatiales, on utilise la technique de la condensation statique pour réduire la taille des systèmes linéaires à résoudre avec un schéma implicite en temps. En plus de proposer une formulation variationnelle avec les conditions de bords classiques comme Neumann, Dirchlet et ABC, on propose aussi une formulation intégrant les PML (en 2D et 3D).

Dans le Chapitre 4, nous avons présenté une revue des schémas de Runge-Kutta (RK) qui sont couramment utilisés dans la littérature. Commençant par des schémas explicites RK (ERK), nous nous sommes finalement tournés vers des schémas implicites RK (IRK) car les ERK ont une contrainte sur le pas de temps liée à stabilité (condition CFL) qui les rend inefficace pour des problèmes raides. Parmi les schémas IRK, nous avons présenté des schémas A-stables (inconditionnellement stable).

Dans la seconde partie de ce manuscrit, nous avons présenté des schémas d'intégration en temps pour les EDO linéaires. Nous avons tout d'abord présenté une méthodologie pour construire deux familles de schémas A-stable. La première famille a été obtenue à partir de l'approximation de Padé d'une fonction exponentielle. Elle a été décrite dans le Chapitre 5. La deuxième famille a été développée dans le Chapitre 6. Les schémas de cette famille, fourni jusqu'à

l'ordre 12, ont été construits en approchant la fonction exponentielle par une fraction de polynôme ayant un unique pôle et sous contrainte d'obtenir un schéma A-stable. Notre travail sur le développement de ces schémas implicites d'ordre élevé et A-stable est aussi décrit dans [5].

Puis dans le Chapitre 7, nous avons fourni des schémas explicites, construits en maximisant leur nombre CFL pour un profil de spectre donné. Ces schémas explicites ont été combinés aux schémas implicites A-stable, présentés dans les chapitres 5 et 6, pour construire des schémas localement implicites dans le Chapitre 8. Après la construction et les tests de validations des schémas en dimension un et deux d'espace, nous avons présenté des résultats numériques obtenus en résolvant les problèmes d'ondes acoustiques et électromagnétiques en dimensions trois. Les résultats obtenus montre la faiblesse des schémas explicites pour des problèmes raides et la limite des méthodes implicites qui consomment de plus en plus d'espace mémoire pour des problèmes de taille grande. Ainsi on voit apparaître la nécessité d'utiliser une méthode localement implicites, qui utilisera moins d'espace mémoire et aura moins de contraintes sur le pas de temps.

Pour les jours à venir, notre objectif est de finaliser nos travaux sur la construction des schémas localement implicites. Dans un premier temps, cela se traduira en parallélisant le code pour pouvoir les tester sur les problèmes en trois dimensions. Puis l'idée serait de combiner les schémas localement implicites et les schémas à pas de temps local pour plus d'optimisations.

# Bibliography


[1] ALEXANDER, R. Diagonally implicit Runge-Kutta methods for stiff O.D.E.'s. *SIAM Journal on Numerical Analysis 14* (1977). 2, 7, 76, 83, 84, 128, 129

[2] AMESTOY, P., DUFF, I., KOSTER, J., AND L'EXCELLENT, J.-Y. A fully asynchronous multi-frontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications 23* (2001), 15–41. 209

[3] APPADU, A. R. Optimized low dispersion and low dissipation Runge-Kutta algorithms in computational aeroacoustics. *Applied Mathematics & Information Sciences 8* (2014), 57–68. 91

[4] AXELSSON, O., AND KUCHEROV, A. B. Real valued iterative methods for solving complex linear systems. Tech. rep., Department of Mathematics, University of Nijmegen, 1999. 3, 7, 106

[5] BARUCQ, H., DURUFLÉ, M., AND 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* (2017), 1–39. 220, 224

[6] BATHE, K.-J. *Finite element procedures, 2nd edition*. 2014. 30

[7] BERENGER, J.-P. A perfectly matched layer for absorption of electromagnetic wave. *Journal of Computational Physics 114* (1994), 185–200. 1, 5, 30

[8] BERGOT, M. *Eléments finis d'ordre élevé pour maillage hybrides. Application à la résolution de systèmes hyperboliques linéaires en régimes harmonique et temporel.* PhD thesis, Université Paris Dauphine, 2010. 29, 31, 33, 41, 53

[9] BERGOT, M., AND DURUFLÉ, M. Higher-order discontinuous Galerkin method for pyramidal elements using orthogonal bases. *Numerical Methods for Partial Differential Equations* (2012). 44

[10] BERLAND, J., BOGEY, C., AND BAILLY, C. Low-dissipation and low-dispersion fourth-order Runge-Kutta algorithm. *Computers & Fluids 35* (2006), 1459–1463. 91

[11] BONNASSE-GAHOT, M. *Simulation de la propagation d'ondes elastiques en domaine fréquentiel par des méthodes Galerkin discontinues*. PhD thesis, Université de Nice-Sophia Antipolis, 2015. 29, 41

[12] BURMAN, E., ERN, A., AND FERNÁNDEZ, M. A. Explicit Runge-Kutta schemes and finite elements with symmetric stabilization for first-order linear PDE systems. *SIAM Journal on Numerical Analysis 48* (2010), 2019–2042. 1, 5

[13] BURRAGE, K. A special family of Runge-Kutta methods for solving stiff differential equations. *BIT Numerical Mathematics 18* (1978), 22–41. 3, 8, 130, 132

[14] BUTCHER, J. C. *Numerical Methods for Ordinary Differential Equations Second Edition*. John Willey & Sons Ltd, 2008. 66, 67, 68, 69, 70, 77, 78, 89, 101, 127, 159, 161

[15] CASH, J. R. On the exponential fitting of composite, multiderivative linear multistep methods. *SIAM Journal on Numerical Analysis 18* (1981), 808–821. 2, 6

[16] CHABASSIER, J., AND IMPERIALE, S. Fourth-order energy-preserving locally implicit time discretization for linear wave equations. *International Journal for Numerical Methods in Engineering 106* (2016), 596–622. 2, 6, 8

[17] CHAURASIA, H. K. *A time-spectral hybridizable discontinuous Galerkin method for periodic flow problems*. PhD thesis, Massachusetts institute of technology, 2014. 29

[18] CHEW, W. C., JIN, J.-M., MICHIELSSEN, E., AND SONG, J. *Fast and efficient algorithms in Computational electromagnetics*. Artech House, INC, 2001. 49

[19] CHIN-JOE-KONG, M. J. S., MULDER, W. A., AND VELDHUIZEN, M. V. High-order triangular and tetrahedral finite elements with mass lumping for solving the wave equation. *Journal of Engineering Mathematics 35* (1999), 405–426. 33

[20] CHRISTOPHE-ARGENVILLIER, A., DESCOMBES, S., AND LANTERI, S. An implicit hybridized discontinuous Galerkin method for the 3D time-domain Maxwell's equations. *Applied Mathematics and Computation 319* (2017), 395–408. 29, 58

[21] CIARLET, P. Notes de cours sur les équations de Maxwell. Lecture Note Master Modélisation et simualtion (M2) ENSTA, 2014. 49

[22] COCKBURN, B., GOPALAKRISHNAN, J., AND LAZAROV, R. Unified hybridization of discontinuous Galerkin, mixed and continuous Galerkin methods for second order elliptic problems. *SIAM Journal on Numerical Analysis 47* (2009), 1319–1365. 41

[23] COHEN, G., AND FAUQUEUX, S. Mixed finite elements with mass-lumping for the transient wave equation. *Journal of Computational Acoustics 8* (2000), 171–188. 1, 5, 29, 30

[24] COHEN, G., AND PERNET, S. *Finite element and discontinuous Galerkin methods for transient wave equations*. Springer, 2017. 1, 5, 30, 33

[25] COHEN, G. C. *Higher-order numerical methods for transient wave equations*. Springer, 2002. 30

[26] COLLINO, F., AND TSOGKA, C. Application of the perfectly matched absorbing layer model to the linear elastodynamic problem in anisotropic heterogeneous media. *Geophysics 66* (2001), 294–307. 35

[27] DEMAILLY, J.-P. *Analyse numérique et équations Différentielles*. EDP Sciences, 2006. 15, 16, 18

[28] DESCOMBES, S., LANTERI, S., AND MOYA, L. Locally implicit time integration strategies in a discontinuous Galerkin method for Maxwell's equations. Tech. rep., INRIA, 2012. 2, 6, 8, 191

[29] DESCOMBES, S., LANTERI, S., AND 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* (2016), A2611–A2633. 191

[30] DIAZ, J., AND GROTE, M. Energy conserving explicit local time stepping for second-order wave equations. *SIAM Journal on Scientific Computing 31* (2009), 1985–2014. 1, 2, 5, 6, 191

[31] DURUFLÉ, M. *Intégration numérique et éléments finis d'ordre élevé appliqués aux équations de Maxwell en regime harmonique*. PhD thesis, Université Paris Dauphine, 2006. 4, 8, 27, 29, 30, 52, 53, 58, 69, 91, 178

[32] DURUFLÉ, M., AND N'DIAYE, M. Optimized high-order explicit Runge-Kutta-Nyström schemes. In *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2016* (2016), M. Bittencourt, N. Dumont, and J. S. Hesthaven, Eds., Springer. 159

[33] EHLE, B. L. *On Padé approximations to the exponential function and A-stable methods for the numerical solution of initial value problems*. PhD thesis, University of Waterloo, 1969. 100

[34] EHLE, B. L. A-stable methods and Padé approximations to the exponential. *SIAM Journal on Numerical Analysis 4* (1973), 671–680. 3, 7, 78, 100, 102

[35] GALLOPOULOS, E., AND SAAD, Y. On the parallel solution of parabolic equations. Tech. rep., Research Institute for Advanced Computer Science NASA Ames Research Center, 1989. 105

[36] GILBERT, J.-C., AND JOLY, P. Higher order time stepping for second order hyperbolic problems and optimal CFL conditions. *Computational Methods in Applied Sciences 16* (2008), 67–93. 159

[37] GIRALDO, F. X., AND TAYLOR, M. A. A diagonal-mass matrix triangular-spectral-element method based on cubature points. *Journal of Engineering Mathematics 56* (2006), 307–322. 33

[38] GIVOLI, D. High-order local non-reflecting boundary conditions: a review. *Wave Motion 39* (2004), 319–326. 1, 5, 30

[39] GRIESMAIER, R., AND MONK, P. Error analysis for a hybridizable discontinuous Galerkin method for the Helmholtz equation. *Journal of Scientific Computing 49* (2011), 291–310. 41

[40] GROTE, M., AND MITKOVA, T. Explicit local time-stepping methods for Maxwell's equations. *Journal of Computational and Applied Mathematics 234* (2010), 3283–3302. 1, 5

[41] HAIRER, E., NORSETT, S. P., AND WANNER, G. *Solving ordinary differential equations I nonstiff problem*. Springer, 2008. 1, 5

[42] HAIRER, E., AND WANNER, G. *Solving ordinary differential equations II stiff and differential-algebraic problem*. Springer, 2010. 2, 6, 7, 20, 21, 22, 68, 69, 76, 84, 85, 159

[43] HESTHAVEN, J. S., AND WARBURTON, T. *Nodal discontinuous Galerkin methods algorithms, analysis, and applications*. Springer, 2008. 1

[44] HOUWEN, P. J. V. D., AND SOMMEIJER, B. P. Phase-lag analysis of implicit Runge-Kutta methods. *SIAM Journal on Numerical Analysis 26* (1989), 214–229. 2, 7, 21, 155

[45] HU, F. Q., HUSSAINI, M. Y., AND MANTHEY, J. L. Low-dissipation and low-dispersion Runge-Kutta schemes for computational acoustics. *Journal of Computational Physics 124* (1996), 177–191. 1, 5, 89, 91

[46] JOLY, P., AND RODRÌGUEZ, J. Optimized higher order time discretization of second order hyperbolic problems: Construction and numerical study. *Journal of Computational and Applied Mathematics 234* (2010), 1953–1961. 1, 5, 159

[47] KANEVSKY, A., CARPENTER, M. H., GOTTLIEB, D., AND HESTHAVEN, J. S. Application of implicit-explicit high order Runge-Kutta methods to discontinuous-Galerkin schemes. *Journal of Computational Physics 225* (2007), 1753–1781. 2, 6, 8

[48] KETCHESON, D. I., AND AHMADIA, A. J. Optimal stability polynomials for numerical integration of initial value problems. *Communications in Applied Mathematics And Computational Science 7* (2012), 247–271. 89, 90, 91, 159, 162, 163, 166

[49] KETCHESON, D. I., PARSANI, M., AND AHMADIA, A. J. Rk-Opt Software for the design of Runge-Kutta methods version 0.2. Tech. rep., King Abdullah University of Science and Technology (KAUST), 2013. 89, 166

[50] KRONBICHLER, M., SCHOEDER, S., MÜLLER, C., AND 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* (2016), 712–739. 1, 5, 29, 41

[51] LANTERI, S., LI, L., AND PERRUSSEL, R. A hybridizable discontinuous Galerkin method for time-harmonic Maxwell's equations. Tech. rep., INRIA, 2011. 29, 58

[52] MAGOULÈS, F., ROUX, F.-X., AND HOUZEAUX, G. *Parallel Scientific Computing*. ISTE Ltd and John Wiley & Sons, Inc, 2016. 82

[53] MAXWELL, J. C. *The dynamical theory of the electromagnetic field*. The Torrance Collection, 1996. 48

[54] MEAD, J. L., AND RENAUT, R. A. Optimal Runge-Kutta methods for first order pseudospectral operators. *Journal of Computational Physics* (1999), 404–419. 90, 159

[55] MEHLIN, M., MITKOVA, T., AND GROTE, M. Runge-Kutta-based explicit local time-stepping methods for wave propagation. *SIAM Journal on Scientific Computing 37* (2015), A747–A775. 2, 3, 6, 8, 191, 196, 197, 199

[56] MOLER, C., AND LOAN, C. V. Nineteen dubious ways to compute the exponential of a matrix, twenty five years later. *SIAM Review 45*, 1 (2003), 3–49. 3, 7, 15, 99

[57] MONK, P. *Finite element methods for Maxwell's equations*. Oxford Science Publications, 2003. 1, 29, 49, 50, 52

[58] MULDER, W. Higher-order mass-lumped finite elements for the wave equation. *Journal of Computational Acoustics 9* (2001), 671–680. 33

[59] NAJAFI-YAZDI, A., AND MONGEAU, L. A low-dispersion and low-dissipation implicit Runge-Kutta scheme. *Journal of Computational Physics 233* (2013), 315–323. 89, 90, 155

[60] NGUYEN, N., PERAIRE, J., AND COCKBURN, B. High-order implicit hybridizable discontinuous Galerkin methods for acoustics and elastodynamics. *Journal of Computational Physics 230* (2011), 3695–3718. 1, 5, 41

[61] NGUYEN, N. C., AND PERAIRE, J. Hybridizable discontinuous Galerkin methods for partial differential equations in continuum mechanics. *Journal of Computational Physics 231* (2012), 5955–5988. 41

[62] NOH, G., HAM, S., AND BATHE, K. J. Performance of an implicit time integration scheme in the analysis of wave propagations. *Computer and Structures 123* (2013), 93–105. 1, 5

[63] NORSETT, S. P. Restricted Padé approximations to the exponential function. *SIAM Journal on Numerical Analysis 15* (1978), 1008–1029. 103

[64] PARSANI, M., KETCHESON, D. I., AND DECONINCK, W. Optimized explicit Runge-Kutta schemes for the spectral difference method applied to wave propagation problems. *SIAM Journal on Scientific Computing 35* (2013), A957–A986. 1, 5, 159

[65] PERNET, S. *Etude de méthodes d'ordre élevé pour résoudre les équations de Maxwell dans le domaine temporel. Application à la détection et à la compatibilité électromagnétique.* PhD thesis, Université Paris Dauphine, 2004. 29, 55

[66] PIETRO, D., ANTOINE, D., AND ERN, A. *Mathematical aspects of discontinuous Galerkin methods*. Springer, 2012. 1

[67] PIPERNO, S. Symplectic local time-stepping in non-dissipative DGTD methods applied to wave propagation problems. *ESAIM Mathematical Modelling and Numerical Analysis 40* (2006), 815–841. 2, 6, 191

[68] RODRÌGUEZ, J. *Raffinement de maillage spatio-temporel pour les équations de l'élastodynamique*. PhD thesis, Université Paris Dauphine, 2004. 2, 6, 191

[69] SAFF, E. B., AND VARGA, R. S. On the zeros and poles of Padé approximants to $e^z$. iii. *Numerische Mathematik 30* (1978), 241–266. 103

[70] SIEGEL, D. M. *Innovation in Maxwell's electromagnetic theory: Molecular vortices, displacement current and light*. Cambridge University Press, 1991. 49

[71] SKVORTSOV, L. M. Diagonally implicit Runge-Kutta methods for stiff problems. *Computational Mathematics and Computational Physics 46* (2006), 2110–2123. 2, 7, 85, 155

[72] SOLIN, P., SEGETH, K., AND DOLEZEL, I. *Higher-order finite element methods*. Chapman & Hall, 2003. 30

[73] STURM, A. *Locally implicit time integration for linear Maxwell's equations*. PhD thesis, Karlsruhe Institute of Technology, 2017. 191

[74] TAM, C. K. W., AND WEBB, J. C. Dispersion-relation-preserving finite difference schemes for computational acoustics. *Computational Physics 107* (1993), 262–281. 21, 76, 86

[75] WANNER, G. Dahlquist's classical papers on stability theory. *BIT Numerical Mathematics 46* (2006), 671–683. 2, 6

[76] WARBURTON, T. A low storage curvilinear discontinuous Galerkin time-domain method for electromagnetics. *IEEE URSI International Symposium of Electromagnetic Theory* (2010). 44

[77] WARBURTON, T., AND HAGSTROM, T. Taming the CFL number for discontinuous Galerkin methods on structured meshes. *SIAM Journal on Scientific Computing 46* (2008), 3151–3180. 1, 6

[78] WIMP, J. On the zeros of a confluent hypergeometric function. *American Mathematical Society 16* (1965), 281–283. 102

## Abstract

In this thesis, we study and develop different families of time integration schemes for linear ODEs. After presenting the space discretisation methods and a review of classical Runge-Kutta schemes in the first part, we construct high-order A-stable time integration schemes for an arbitrary order with low-dissipation and low-dispersion effects in the second part. Then we develop explicit schemes with an optimal CFL number for a typical profile of spectrum. The obtained CFL number and the efficiency on the typical profile for each explicit scheme are given. Pursuing our aim, we propose a methodology to construct locally implicit methods of arbitrary order. We present the locally implicit methods obtained from the combination of the A-stable implicit schemes we have developed and explicit schemes with optimal CFL number. We use them to solve the acoustic wave equation and provide convergence curves demonstrating the performance of the obtained schemes. In addition of the different 1D and 2D validation tests performed while solving the acoustic wave equation, we present numerical simulation results for 3D acoustic wave and the Maxwell's equations in the last part.

**Key words**: ODEs, time integration, high-order schemes, Padé, Runge-Kutta, SDIRK, acoustic wave, Maxwell's equations, FEM, HDG.

---

## Résumé

Dans cette thèse, nous étudions et développons différentes familles de schémas d'intégration en temps pour les EDO linéaires. Dans la première partie, après avoir introduit les définitions et propriétés utilisées pour construire les schémas en temps, nous présentons deux méthodes de discrétisation en espace et une revue des schémas de Runge-Kutta (RK) qui sont couramment utilisés dans la littérature. Dans la seconde partie on présente une méthodologie pour construire deux familles de schémas A-stable pour un ordre quelconque. Puis on fournit des schémas explicites, construits en maximisant leur nombre CFL pour un profil de spectre donné. Ces schémas explicites sont ensuite combinés aux schémas implicites A-stable, pour construire des schémas localement implicites que nous décrivons. En plus des tests de validations des schémas pour des problèmes en dimension un et deux de l'espace, nous présentons des résultats numériques obtenus en résolvant des problèmes de propagation d'ondes acoustiques et électromagnétiques en dimension trois dans la troisième partie.

**Mots clés** : EDO, intégration en temps, schémas d'ordre élevé, Padé, Runge-Kutta, SDIRK, ondes acoustiques, équations de Maxwell, FEM, HDG.