THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE BORDEAUX

Délivré par : *l'Université de Bordeaux*

Présentée et soutenue le *(8/12/2014)* **par :**

Yohann DUDOUIT

Raffinement spatio-temporel par une approche de Galerkin discontinue en élastodynamique pour le calcul haute performance

Spatio-temporal refinement using a discontinuous Galerkin approach for elastodynamic in a high performance computing framework

École doctorale et spécialité :
Mathématiques et informatique : Mathématiques appliquées et calcul scientifique
Unité de Recherche :
Inria Bordeaux-Sud Ouest, projet HIEPACS

À mon grand-père, Pierre

Résumé

Cette thèse étudie le raffinement local de maillage à la fois en espace et en temps pour l'équation de l'elastodynamique du second ordre pour le calcul haute performance. L'objectif est de mettre en place des méthodes numériques pour traiter des hétérogénéités de petite taille ayant un impact important sur la propagation des ondes. Nous utilisons une approche par éléments finis de Galerkin discontinus avec pénalisation pour leur flexibilité et facilité de parallélisation. La formulation éléments finis que nous proposons a pour particularité d'être élasto-acoustique, pour pouvoir prendre en compte des hétérogénéités acoustiques de petite taille. Par ailleurs, nous proposons un terme de pénalisation optimisé qui est mieux adapté à l'équation de l'élastodynamique, conduisant en particulier à une meilleure condition CFL. Nous avons aussi amélioré une formulation PML du second ordre pour laquelle nous avons proposé une nouvelle discrétisation temporelle qui rend la formulation plus stable. En tirant parti de la p-adaptivité et des maillages non-conformes des méthodes de Galerkin discontinues combiné à une méthode de pas de temps local, nous avons grandement réduit le coût du raffinement local. Ces méthodes ont été implémentées en C++, en utilisant des techniques de template metaprogramming, au sein d'un code parallèle à mémoire distribuée (MPI) et partagée (OpenMP). Enfin, nous montrons le potentiel de notre approche sur des cas tests de validation et sur des cas plus réalistes avec des milieux présentant des hydrofractures.

Mots clefs: élastodynamique, Galerkin discontinu, raffinement spatio-temporel, maillage cartésien, non-conforme, pas de temps local, couplage élasto-acoustique, hydrofracture, hpc, OpenMP, MPI, PML, IPDG, stabilité, schéma *hp*

Abstract

This thesis studies local mesh refinement both in time and space for the second order elastodynamic equation in a high performance computing context. The objective is to develop numerical methods to treat small heterogeneities that have global impact on wave propagation. We use an internal penalty discontinuous Galerkin finite element approach for its flexibity and parallelization capabilities. The elasto-acoustic finite element formulation we discuss is elasto-acoustic in order to handle local acoustic heterogeneities. We also propose an optimized penalty term more suited to the elastodynamic equation that results in better CFL condition. We improve a second order PML formulation with an original time discretization that results in a more stable formulation. Using the p-adaptivity and non-conforming mesh capabilities of discontinuous Galerkin methods combined with a local time stepping method, we greatly reduce the high computational cost of local refinements. These methods have been implemented in C++, using template metaprogramming, in a distributed memory (MPI) and shared memory (OpenMP) parallel code. Finally, we show the potential of our methods on validation test cases and on more realistic test cases with medium including hydrofractures.

Keywords: elastodynamic, discontinuous Galerkin, spatio-temporal refinement, Cartesian mesh, non-conforming, local time step, elasto-acoustic coupling, hydrofracture, hpc, OpendMP, MPI, PML, IPDG, stability, *hp* scheme

Remerciements:
Tout d'abord je voudrais remercier mes encadrants, Luc Giraud, Florence Millot et Sébastien Pernet, pour m'avoir apporté leurs connaissances, leur expérience, leur temps et leur soutien jusqu'à l'aboutissement de cette thèse. J'aimerais particulièrement remercier Sébastien Pernet pour les nombreuses conversations que nous avons eues qui m'ont particulièrement motivé tout au long de cette thèse. Il a toujours su faire part d'honnêteté intellectuelle, prendre le temps de me partager ses connaissances et m'encourager à chercher lorsque mes questions dépassaient ses compétences.

J'aimerais remercier Jean-Luc Boelle et Issam Tarrass pour m'avoir apporté leurs lumières sur la géophysique, ainsi que pour avoir accompagné cette thèse du début jusqu'à sa fin.

Je voudrais aussi remercier les membres du jury, Stéphane Lanteri et en particulier les rapporteurs Christophe Geuzaine et Philippe Helluy qui ont pris le temps de lire avec attention mon manuscrit.

J'aimerais aussi remercier tous les membres de l'équipe algo du CERFACS que j'ai rencontré au cours de ces années; le multiculturalisme de l'équipe a vraiment été une belle expérience qui m'a beaucoup apporté personnellement et professionnellement. Je voudrais aussi remercier Marie de l'équipe GlobC pour m'avoir supporté pendant ces précieuses pauses qui m'ont permis de me changer les idées quand j'en avais besoin.

Je voudrais aussi remercier mes amis, en particulier ceux qui m'ont soutenu dans les moments difficiles: Clément, Adrien, Charlotte, Karen, Mr Teitgen, Victor; mais aussi ceux qui ont partagé ma vie de tous les jours: Brice, Ben, Simon, Benji, Maëlle, Milie, Clem', Mirabelle, Lars, Coline, Fréd, Romain, Renaud, Florian, Antoine, Arnaud, Baptiste, JBL, JBO, David, Benjamin.

Mes remerciements vont aussi naturellement à ma famille, en particulier à ma mère, Christine, qui m'a toujours soutenu tout au long de mes études.

Contents

Introduction

Presentation of the context

Oil exploration began with a mixture of luck and superstition. Prospectors were content with drilling near seeps or in favorable locations, or just randomly. But the days when prospectors were throwing their hats in the air and were drilling where their hat fell is long gone. If we had continued in this way, our reserves would be far to suffice us.

We can decompose the oil exploitation process in three main parts, exploration, drilling and exploitation. The exploration consist in seeking places where the topography of the ground can "trap" the black gold. Drilling is the key to oil exploration. This step is the main and most of the total cost of an oil installation. This is why exploration is crucial, making a useless drilling is an economical disaster. The final step is the extraction, this last step can be divided in two repeating sub-steps: estimation and recovery. Once an oil field is actually detected by a drilling, a step of evaluation by several tests is performed to determine the amount of oil (volume and porosity of the reservoir) and ease to extract it (permeability of the rock) and to determine the composition of what is extracted. This evaluation process is performed to estimate at the end the profitability to exploit the well. When exploitation is decided comes the step of oil recovery. According to the different phases in the life of the oil field the techniques to dig out the oil varies. Each step in the oil exploitation process requires its own scientific methods. However, It is only relatively late in the history of oil extraction that scientific methods have been used, but modelling methods are nowadays at the heart of any geophysical interpretation approach. Our work fall within the exploration phase.

Exploration is a step involving multiple knowledge, geologists, geophysicists, mathematicians, numerical analysts. All bringing their share of knowledge to determine the constitution of the ground with the limited information available. With the intensive exploitation of oil fields, it has become increasingly difficult to find new untapped fields. The vast majority of "easy" to find fields have already been found.
 Without obeying to specific physical laws, the existence of oil is based on two basic criteria:

- Hydrocarbons (oil) must have formed in favourable grounds called bedrock; these lands necessarily correspond to certain stages of marine sedimentation with deposition of organic materials whose physico-chemical evolution leads to the formation of hydrocarbons.

- In order to create an oilfield, oil must have been, after their formation, collected, and then "trapped" in "reservoirs". The term "reservoir" stands for a sealed space at the top, bounded by clay or by an impermeable rock, wherein there is a porous rock, comparable to a sponge. This porous rock is impregnated with gas and / or oil and / or salt water.

Reservoir quality is characterized by its porosity (the more the rock is porous, the greater the volume of oil content is) and permeability (the ability to extract oil). Exploration consists of recovering a lot of data to end up with a more or less sophisticated model of the ground. These data are mainly composed of seismic data, coring and geological knowledge. On land, the wave generation is done either with explosives or with vibrator trucks. At sea, a boat towing a device for generating waves compressed air and a network of pressure sensors divided into lines (streamers) up to 10 km long. Numerical methods can be useful before the data acquisition to help predicting the quality of the acquisition. Numerical methods are also the key to accurate ground modeling.

Interpreting geophysical data in complex geological terrains requires solutions of the partial differential equations (PDE) governing the physics. Since ground modeling is performed through simulations that seek to match the acquired field data, this problem is what we call an *inverse problem*. What we call the *direct problem* is the simulation of a wave propagation in a defined ground model. The *inverse problem* is the opposite problem: seeking the ground model such that we get the known wave propagation corresponding to the field acquired data. When solving the inverse problem most approaches require to solve many direct problems to approximate iteratively the ground model solution.

Our work is focused on the direct problem, among all the numerical methods available, the most common are: the spectral method [? ? ?], very efficient and accurate but generally restricted to simple earth structures, often layered earth; the pseudo-spectral [? ? ?], finite difference [? ? ?] and finite volume methods [? ? ?] based on the strong formulation of the partial differential equations, easy to implement and usually representing a good compromise between accuracy, efficiency, and flexibility; and the continuous [? ?] or discontinuous Galerkin finite-element methods [? ?] based on the weak formulation, leading to more accurate earth representations and therefore more accurate solutions but with a higher computational cost and a more complex usage. The choice between these different approaches is still difficult and depends on the applications. Spectral methods are often called with the more general term analytical or semi-analytical methods, whereas all other methods are numerical methods.

On top of the different numerical methods, different physics models are used according to the desired cost/accuracy from the simplest to the more realistic we have: the acoustic model, the isotropic elastodynamic model, the anisotropic elastodynamic model, and we can even add some porosity physics to these models. The diversity in solving geophysical modelling may, however, reflect the different challenges in geophysics, and these challenges may require different practical solutions. One shall not think that the simplest methods and models are the old ones, a large portion of geophysics codes still use finite differences and/or acoustic model. For instance, to be economically valuable, the migration of hundreds of thousand shots of a marine data set to obtain a structural image from compressional waves demands a different implementation of the wave propagation problem that the precise modelling of surface waves generated by a superficial earthquake. The methodological effort for years has conducted to sophisticated tools well tuned for specific purposes.

The increasing difficulty to find reservoirs has bring the need to always render the physic more accurately. In particular, being capable to render small details that have a considerable impact on the wave propagation is becoming mandatory. In the presence of complex geometry and complex geological models, adaptivity and mesh refinement are key features for efficient numerical solution of the elastodynamic equation. Refined meshes impose severe stability constraints on explicit time-stepping schemes to respect the CFL condition to insure the stability of the method. When mesh refinement is restricted to a small area, the time step defined by the spatially smallest element has to be used. Overcoming this limitation is crucial for achieving high performance and high numerical accuracy. Decreasing the interpolation order if refinement ratio is low is a practical approach [**?** **?**] since the CFL condition is larger for lower interpolation orders. However, when the spatial refinement becomes to steep local time-stepping schemes with local stability conditions will be the method of choice.

Collino *et al.* [**?** **?**] proposed a second-order local time-stepping method for the wave equation and for Maxwell's equations. The approach remains explicit inside the coarse and fine meshes but requires at every time step the solution of a linear system at the interface between the two grids. Piperno [**?**] proposed an explicit local time-stepping scheme conserving a discrete energy and second-order accurate in time by combining a symplectic integrator for the Maxwell's equation while Dumbser *et al.* [**?**] combine both p-adaptivity and local time stepping using the ADER integration scheme which is a dissipative scheme. Alternatively, Diaz and Grote [**?**] have proposed a fully explicit local time-stepping approach with the conservation of a discrete energy with arbitrarily high accuracy for the scalar wave equation while Dolean *et al.* [**?**] have proposed an hybrid implicit-explicit (or locally implicit) method.

Local time stepping methods bring two main problems. Firstly, their accuracy and stability cannot always be guarantee. Secondly, they introduce more or less sophisticated algorithms that lead to difficult parallelization.

Objectives and contributions of the thesis

The objective of this thesis is to develop a numerical method with local mesh refinement both in space and time on Cartesian grids adapted to a high-performance environment for the elastodynamic equation in isotropic medium.

The targeted average rate of refinement being around 20, which is substantial, the refinement method must guarantee a priori the stability of such refinements. In addition, such refinements lead to refined areas with high computational costs. This imbalance involves having a viable strategy in a high performance environment. Indeed, as we mentioned earlier, temporal local mesh refinement methods induce a particular treatment making difficult the load balance. One of the reason to achieve local mesh refinement is to simulate hydrofractures, this implies to be capable to handle multiphysics media, typically elastodynamic and acoustic media. The interest is usually not on simulating a single hydrofracture, but a network of hydrofractures for the cumulated physical effects they produce, *e.g.* wave scattering. Besides, the lack of precise information on the ground and also the tools used on the side, promote a Cartesian grid approach. This means that space refinements should be non-conforming, see Figure 1 for an illustration of a non-conforming mesh. These non-conforming meshes induce stability problems on most numerical methods, or at least require complicated numerical schemes.

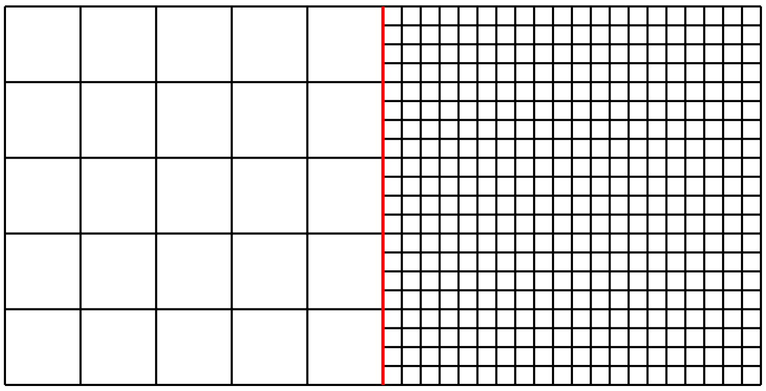


Figure 1: A non-conforming mesh, elements are non-conforming on the red interface.

In the first chapter we introduce the numerical method to discretize our EDP: the interior penalty discontinuous Galerkin methods, with an emphasis on its symmetric version. Standard approach on Cartesian grids would use finite difference or finite volume methods for their reduced cost, but all the requirements mentioned in the objectives seemed unreachable with such methods. Because discontinuous Galerkin methods are local, they are particularly well-suited for the development of explicit local time-stepping schemes. Additionally, non-conforming mesh refinements are naturally handled by these methods. These methods are not commonly used in seismic simulation due to their relative high cost and difficult implementation compared to finite difference and finite volume methods, a first preliminary attempt of this method for seismic imaging has been performed by De la Puente [**?**] only in 2010. For this reason, we decided to dedicate the first chapter to a detailed introduction to discontinuous Galerkin method for the second order elastodynamic equation in the time domain. This introduction presents how this method is built and recall some properties of it. In particular, we performed a dispersion error analysis and a study of the stability condition that arise in explicit schemes, also called the CFL condition.

We also propose a new formulation for the penalty term which is more suited for the elastodynamic equation due to its vector components. This new formulation leads to a better stability condition and dispersion error, and paves the way for multiphysics simulations.

In the second chapter we introduce absorbing layers, called perfectly matched layers (PML). Indeed, in our context, the simulations are never made on the whole earth, so there must be absorbing conditions to simulate an unbounded medium. We decided to choose PML over other absorbing methods for its flexibility and reliability. We based our PML scheme on Imbo's formulation [**?**] due to its second order PDE form contrary to most other formulations that lead to a system of first order PDE. We proposed an original discontinuous Galerkin approximation of this PML formulation. Even though PML schemes often lead to weakened CFL conditions [**?**], we found through extensive numerical experimentations that the choices we made for our discontinuous Galerkin approximation and for the temporal discretization do not weaken the CFL condition.

In the third chapter we introduce our local time stepping approach, based on Diaz-Grote's local time stepping method [**?**]. Diaz and Grote's local time stepping method appears in a high performance computing context as the best suited method for two main reasons. Firstly, The stability of the local time-stepping method can be proven through the conservation of a discrete energy. Secondly, the computational complexity is more homogeneous for Diaz-Grote's method than for other methods since the scheme is fully explicit.

The third chapter can be subdivided in four parts. The first part is dedicated to the construction of a scheme, which we call the *$\tilde{z}$-exact scheme*. The aim of this scheme is to give a better insight to Diaz-Grote's scheme, since this last one is an approximation of the $\tilde{z}$-exact scheme. Contrary to Diaz-Grote's scheme, the $\tilde{z}$-exact scheme does not use a local time step. Nevertheless, most of the numerical properties of both schemes are the same, especially the stability condition. Indeed, the stability condition is not impacted by the area receiving a special treatment which is precisely what is desired from such schemes.

In the second part we introduce Diaz-Grote's local time stepping algorithm. Diaz-Grote's algorithm uses the global stiffness matrix which is usually not assembled. Moreover, writing the local time stepping algorithm in this manner hides the locality of the algorithm. Using the locality of the operators of the discontinuous Galerkin methods, we proposed specific local algorithms for elements at either fine or coarse time step.

In the third part we propose an analysis of the optimal computational cost we can expect for an ideal local time stepping method. To overcome the quick growth in computational cost of local spatio-temporal mesh refinement we propose some strategies based on discontinuous Galerkin methods flexibility. The first idea is to use use lower polynomial orders in refined elements, this uses what is called p-adaptivity, *i.e.* the ability to change polynomial orders between elements.

In the fourth part we propose to analyze the numerical behavior of the local time stepping method and of the non-conforming mesh refinement. In particular, we seek to observe spurious effects created by these special treatments.

In the fourth chapter we attempt to validate our choices of methods. In a first time, we introduce our approach to achieve multiphysics, *i.e.* elasto-acoustic media. This multiphysics formulation is highly helped by the flexibility of discontinuous Galerkin methods. Secondly, we validate the different aspects of our methods on canonical test cases. In particular, we simulate an hydrofracture and we compare our results to reference results. Finally, we illustrate the capabilities of our methods on illustrative experiments showing the impact of small heterogeneities.

In the fifth chapter we introduce our implementation and our approach to parallelization. In our implementation we attempt to exploit the industrial constraints to gain efficiency compared to standard implementation approaches. Our approach is based on the decomposition of the computational domain into subdomains. These subdomains are the entities that are distributed to achieve distributed memory parallelism. However, the size of these subdomains and the polynomial order of approximation of the elements has an significant impact on the sequential performances. Therefore, we study the sequential performances according to the size and polynomial orders of the subdomains. Considering our parallel approach we introduce our shared and distributed strategies. We propose an asynchronous non-blocking MPI implementation, that shows consistent scalability. The distributed memory parallel approach showed much better performances than our shared memory parallel approach.

Brief introduction to elastodynamic

The linear isotropic elastic model

Mechanical properties of materials have a very complex behavior. Most materials have a nonlinear *elastoplastic* behavior, *heterogeneity* and *anisotropy.* This means that mechanical properties may vary due to many different aspects, especially deformation and load history. Depending on the type of targeted applications these behaviors and properties can be simplified.
In the case of small strain it is reasonable to assume the *elastoplastic* behavior to be purely *elastic.* In the context of seismic wave propagation, materials are often assumed to be isotropic and locally homogeneous.

Wave types

Seismic waves can be sorted in two categories, *body waves* and *surface waves.* As their name suggests body waves spread over the volume, forming spherical wave-fronts around the source point. This implies a faster decay of the energy, and hence the displacement amplitude, for body waves with distance from the source than for surface waves.

Body waves: They propagate inside the earth. Their propagation speed depends on the medium, which typically, increases with depth.

- **P-waves** or *primary* waves, also called compressional waves and longitudinal waves. They are the fastest waves and therefore the first to be recorded on seismograms. The particle motion is pure dilatation or pressure. These ground motions are parallel to the direction of the wave propagation. The P-waves correspond to the acoustic waves in a fluid, *e.g.* in air or water. They are responsible for the low rumble that can be heard at the beginning of an earthquake.

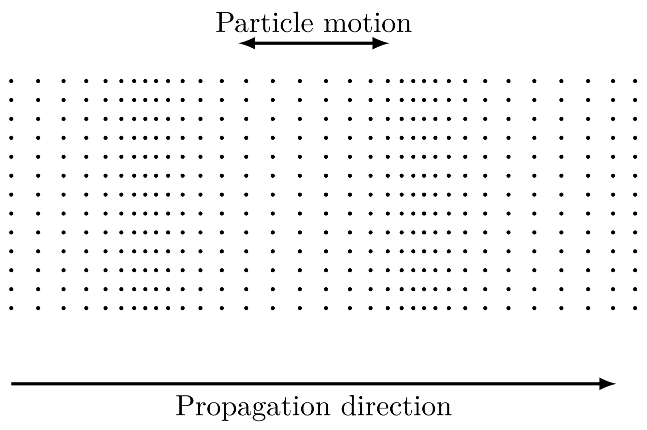


Figure 2: P-wave

- **S-waves** or *secondary* waves, also called shear waves and transversal waves, they arrive after P-waves. The ground motion is perpendicular to the direction of the wave propagation. These waves do not propagate in fluid media.

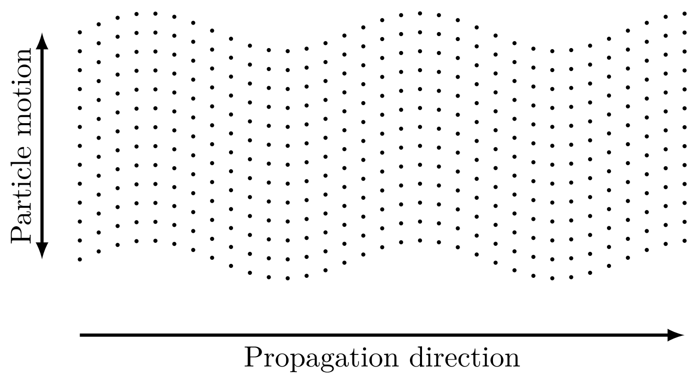


Figure 3: S-wave

When a wave encounters a free surface, or an interface between two media, a partial conversion from P-waves to S-waves and vice versa may occur.

Surface waves: They propagate along a free surface, *e.g.* earth surface, or along an interface between two media especially fluid-solid interface. Their velocity is lower than body waves, but their amplitude is often the highest and for this reason they are the most destructive waves. We introduce here two commonly referred surface waves, but more types of surface waves exist.

- **Rayleigh waves** typically run on the Earth surface, but also on fluid-solid interface. These waves are somewhat slower than S-waves, and contain both pressure and shear components in the displacement field.

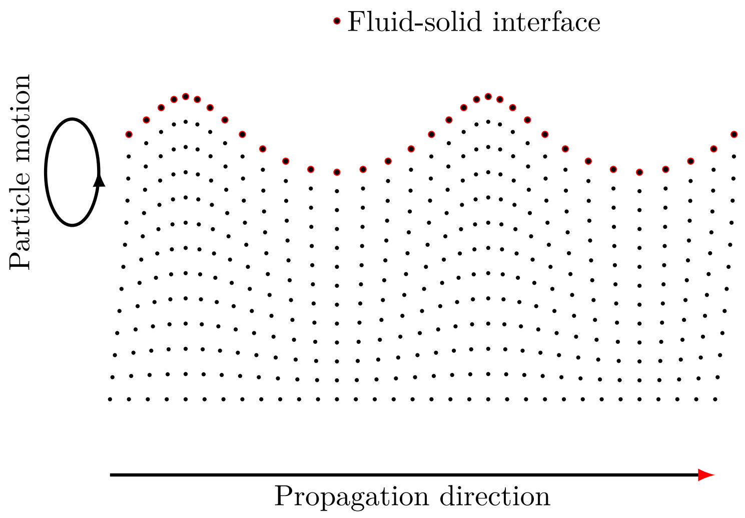


Figure 4: Rayleigh wave

- **Love waves** may arise due to multiple reflections of S-waves between two interfaces, they consist of trapped S-waves because reflection on interfaces are total, these shear waves are polarized normally to the interfaces.

Chapter 1

Discontinuous Galerkin for elastodynamic

1.1 Introduction

DG methods were first introduced in 1973 by Reed and Hill [**?**], and have gain slowly popularity until twenty years ago when a keen interest began. A large number of variants and results have slowly emerged since the first formulation [**? ? ? ? ?**]. DG methods can be viewed as finite element methods allowing for discontinuities between elements. These discontinuities require to introduce numerical fluxes between elements at interfaces as for finite volume methods. Working with discontinuous discrete spaces offers a substantial amount of flexibility, *e.g.* *hp*-adaptivity, non-conforming meshes, truly explicit schemes, and also the possibility to achieve multi-physics simulations as shown in Chapter 4.

First of all, we shall recall the various features desired for our software. We want a method that handle Cartesian non-conforming meshes, local time stepping and elasto-acoustic media. Without going too much into details, especially on local time-stepping since it is the subject of Chapter 3, we have to choose a method that can handle non-conforming meshes and elasto-acoustic interfaces. Both of these features are achievable with DG finite element methods, non-conforming meshes are naturally handled by DG methods whereas a small change in the formulation of the DG methods is used to manage elasto-acoustic interfaces seamlessly to the user.

Geo-science has been widely and mainly using finite difference methods for its ease of implementation and efficiency on simple simulations. With the increasing complexity of problems, finite difference had to become more and more complex, making them less attractive. However, comparing the performances of DG methods and finite difference methods is not simple, as one can easily build test cases where one method is better suited than the other. What should be noticed is that DG methods and finite difference methods should not be used the same way, especially if we consider accuracy and dispersion aspects.

In this chapter, we mainly focus on the application of the **I**nterior **P**enalty **D**iscontinuous **G**alerkin (IPDG) finite element methods to time-dependent elastic wave propagation, with an emphasis on the ***S****ymmetric* ***I****nterior* ***P****enalty* ***D****iscontinuous* ***G****alerkin* method (SIPDG). In Section 1.2 we introduce the mathematical formulation of our problem, called the model problem. Our approach is relatively standard in the sense that we use DG methods for the discretization in space, and finite difference for the discretization in time. For this reason, in Section 1.3 we introduce the principal ideas of the construction of the IPDG approximation for the stationary elasticity operator and why it is built that way. We also introduce in this section a new penalty better suited for the elastodynamic equation. In Section 1.4 we briefly give the IPDG formulation of the model problem, that is the elastodynamic equation in the time domain, and then we discretize in time the semi-discrete problem with the well known leap-frog finite difference scheme. In Section 1.5 we study through a plane wave analysis the dispersion and stability properties in homogeneous infinite medium of our DG approximation. In Section 1.6 we study our DG approximation through an energy analysis the stability properties with heterogeneities, *hp* non-conforming mesh, and boundary conditions. This second study is less accurate than the plane wave analysis, but gives valuable information about the impact of heterogeneities, *hp*-adaptivity, and boundary conditions on the stability.

1.2 Model problem

Let Ω be a polygonal domain of $\mathbb{R}^d$, $d = 1, 2$ or 3. The sides of the boundary $\partial\Omega$ are grouped into two disjoints sets Γ_D and Γ_N. Let $\mathbf{n}$ be the unit normal vector to the boundary exterior to Ω. We consider the following hyperbolic linear elastodynamic problem:

Find $\mathbf{u} : \Omega \times [0, T] \to \mathbb{R}^d$ such that

$$\begin{cases} \rho \dfrac{\partial^2 \mathbf{u}}{\partial t^2} - \operatorname{div}(\sigma(\mathbf{u})) = f, & \text{in } \Omega, \\ \mathbf{u} = 0, & \text{on } \Gamma_D, \\ \sigma(\mathbf{u}) \cdot \mathbf{n} = 0, & \text{on } \Gamma_N, \\ \mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}), & \forall \mathbf{x} \in \Omega, \\ \dfrac{\partial \mathbf{u}}{\partial t}(\mathbf{x}, 0) = \mathbf{v}_0(\mathbf{x}), & \forall \mathbf{x} \in \Omega, \end{cases} \tag{1.1}$$

where $\sigma(\cdot)$ is the Cauchy stress tensor, $\mathbf{u}(\mathbf{x}, t)$ is the displacement field, ρ is the mass density, the vector $\mathbf{x}$ is the position in space and t is the time. In homogeneous and isotropic materials, the Cauchy stress tensor can be written as:

$$\sigma(\mathbf{u}) := 2\mu e(\mathbf{u}) + \lambda tr(e(\mathbf{u}))I,$$

where $e(\mathbf{u}) = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)$ is the strain tensor, λ and μ are the Lamé parameters, I the identity matrix and $tr(.)$ the trace function.

We recall that the Lamé parameters are linked to P- and S- waves velocities by the relations

$$\lambda + 2\mu = \rho v_p^2, \quad \mu = \rho v_s^2.$$

Existence and uniqueness of the solution for the elastodynamic equation: We state now a classical result of existence and uniqueness obtained by semigroup theory [**?**]. We define the elasticity operator by $\mathcal{A}\mathbf{u} := -\operatorname{div}(\sigma(\mathbf{u}))$. The domain of this operator is

$$\mathcal{D}(\mathcal{A}) := \{\mathbf{v} \in H^1(\Omega)\,,\, \mathcal{A}\mathbf{v} \in L^2(\Omega) \text{ and } \sigma(\mathbf{v})\mathbf{n} = 0 \text{ on } \Gamma_N\}.$$

With the theorem of Hille-Yosida [**?**] we obtain the following classical result:

Theorem 1.1
Under the hypothesis:

- $\lambda, \mu, \rho \in L^\infty(\Omega)$, and $\exists \lambda_0, \mu_0, \rho_0 > 0$, such that $\forall x \in \Omega, \lambda(x) > \lambda_0$ and $\mu(x) > \mu_0$ and $\rho(x) > \rho_0$;
- $(\mathbf{u}_0 \, , \, \mathbf{v}_0) \in \mathcal{D}(\mathcal{A}) \times \tilde{H}^1_0(\Omega)$, where $\tilde{H}^1_0(\Omega) = \{v \in H^1(\Omega) \; : \; v = 0 \text{ on } \partial\Omega \cap \Gamma_D\}$;
- $f \in C^1(\mathbb{R}^+ \, ; \, L^2(\Omega))$;

our problem has a unique solution:

$$\mathbf{u} \in C^2(\mathbb{R}^+ \, ; \, L^2(\Omega)) \cap C^1(\mathbb{R}^+ \, ; \, \tilde{H}^1_0(\Omega)) \cap C^0(\mathbb{R}^+ \, ; \, \mathcal{D}(\mathcal{A})). \tag{1.2}$$

1.3 Discontinuous Galerkin approximations of the elasticity operator

Building the interior penalty discontinuous Galerkin methods requires two main steps. The first step is to derive an equivalent formulation called variational formulation. The second step is to use finite dimension approximation spaces to discretize in space the variational formulation.

In short, the principle of the variational approach for solving partial differential equations is to replace the original equation by an equivalent formulation obtained by integrating the equation multiplied by any function, called test function. The main idea of the variational approach is to show the existence and uniqueness of the solution of the variational formulation, leading to the same result for the model problem. However, this theory does not work unless the space in which we seek the solution and wherein the test functions are is a Hilbert space. This is not the case for $C^1_0(\Omega)$ with its usual scalar product. This is why we seek our solution in the Sobolev spaces, in particular $H^1_0(\Omega)$ which is a Hilbert space. A brief introduction to Sobolev spaces can be found in Appendix A. However, what must be remembered is that we use functional spaces of sufficient regularity.

To introduce the discontinuous Galerkin approximation of the elasticity operator $\mathcal{A} := -\text{div}\sigma(\mathbf{u})$, we consider the following stationary problem:
Find $\mathbf{u} \, : \, \Omega \to \mathbb{R}^d$ solution of

$$\left\{ \begin{array}{rcl} -\text{div}\sigma(\mathbf{u}) & = & f \text{ in } \Omega, \\ \mathbf{u} & = & 0 \text{ on } \Gamma_D, \\ \sigma(\mathbf{u})\mathbf{n} & = & 0 \text{ on } \Gamma_N. \end{array} \right. \tag{1.3}$$

This problem can be written as follows:
Find $\mathbf{u} \in \tilde{H}^1_0(\Omega)$ such that

$$\forall \mathbf{v} \in \tilde{H}^1_0(\Omega), \quad a(\mathbf{u}, \mathbf{v}) = \ell(\mathbf{v}), \tag{1.4}$$

where $a(\mathbf{u}, \mathbf{v}) := \displaystyle\int_\Omega \sigma(\mathbf{u}) : \nabla \mathbf{v} \, d\mathbf{x}$ and $\ell(\mathbf{v}) := \displaystyle\int_\Omega f \cdot \mathbf{v} \, d\mathbf{x}$.
In particular, under the hypothesis:

- the measure of Γ_D is non-null,

- $f \in L^2(\Omega)$,
- $\lambda, \mu \in L^\infty(\Omega)$, and $\exists \lambda_0, \mu_0 > 0$, such that $\forall x \in \Omega, \lambda(x) > \lambda_0$ and $\mu(x) > \mu_0$,

the Lax-Milgram theorem ensures the well-posedness of the problem (1.4).

We shall now present the construction of a discontinuous Galerkin approximation of the weak solution of (1.4). In this kind of approach, the discrete solution is sought in a finite dimension space, V_h, defined by piece on a subdivision of Ω. In particular, no continuity is assumed between the elements of the subdivision and thus V_h is not included in $\tilde{H}_0^1(\Omega)$.

1.3.1 Properties of a "good" discontinuous Galerkin approximation

Before explaining the construction of the formulations, we shall clarify what is meant by "good" discontinuous Galerkin approximation (or other). For this, we consider the following abstract formulation:

Find $\mathbf{u}_h \in V_h$ such that

$$\forall \mathbf{v}_h \in V_h, \quad a_h(\mathbf{u}_h, \mathbf{v}_h) = l(\mathbf{v}_h),$$

where a_h is the bilinear form underlying the selected scheme.

Suppose that:

- a_h verifies a uniform inf-sup condition, *i.e*,

  $\exists \beta > 0$ such that

$$\inf_{\mathbf{u}_h \in V_h} \sup_{\mathbf{v}_h \in V_h} \frac{a_h(\mathbf{u}_h, \mathbf{v}_h)}{\|\mathbf{u}_h\|_h \|\mathbf{v}_h\|_h} \quad \geq \quad \beta > 0, \tag{1.5}$$

- There exists a norm $\|\cdot\|_{V(h)}$ on the space $V(h) := \tilde{H}_0^1(\Omega) + V_h$ (we have to define $V(h)$ since V_h might not included in $\tilde{H}_0^1(\Omega)$ and is never included for DG methods), such that the injections are continuous for the norms $||.||_{H^1(\Omega)}$ and $||.||_h$,
- We can extend a_h in a continuous bilinear form of $V(h) \times V_h$ (still noted a_h), *i.e.*,

  $\exists C > 0$ such that $\forall \mathbf{v} \in V(h)$ and $\forall \mathbf{v}_h \in V_h$,

$$a_h(\mathbf{v}, \mathbf{v}_h) \quad \leq \quad C \|\mathbf{v}\|_{V(h)} \|\mathbf{v}_h\|_h. \tag{1.6}$$

We thus get (we refer to [**?**] for a proof of these results)

- the stability of the discrete solution according to the data of the problem:

$$\|\mathbf{u}_h\|_h \quad \leq \quad \frac{C}{\beta} \|f\|_0, \tag{1.7}$$

- the *a priori* error estimate:

$$\|\mathbf{u} - \mathbf{u}_h\|_{V(h)} \quad \leq \quad \left(1 + \frac{C}{\beta}\right) \inf_{\mathbf{v}_h \in V_h} \|\mathbf{u} - \mathbf{v}_h\|_{V(h)}. \tag{1.8}$$

  In particular, this estimate is used to show the convergence of the scheme and determine its order (under some hypothesis on the regularity of the exact solution).

In practice, we seek to provide formulations verifying the hypothesis (1.5) and (1.6) in order to obtain a "good" discontinuous Galerkin approximation, *i.e*, a stable and converging approximation.

1.3.2 Construction of interior penalty discontinuous Galerkin approximations

We will introduce now the formal construction of several standard discontinuous Galerkin formulations called interior penalty discontinuous Galerkin. We refer to [**?**] for a deeper insight into these methods.

Let Ω be subdivided into square elements in 2D and cubes in 3D (they can have more complex shapes in the general case), we denote this partition by $\mathcal{T}_h$. In order to achieve spatial local mesh refinement, we allow non-conforming elements. We denote by $\mathcal{F}_h$ the set of all faces. A face shared by two elements is called an interior face, we denote by $\mathcal{F}_h^{\mathcal{I}}$ the set of all interior faces. Likewise, a boundary face of $K \in \mathcal{T}_h$ is $\partial K \cap \partial\Omega$, we denote by $\mathcal{F}_h^{\mathcal{B}}$ the set of all boundary faces. We also denote by $\mathcal{F}_K$ the set of faces of an element K.

For any piecewise smooth function $\mathbf{v}$, we define the following trace operators. Let $F \in \mathcal{F}_h^{\mathcal{I}}$ be an interior face shared by two neighboring elements K_1 and K_2. We assume that the normal vector $\mathbf{n}_F$ to the face F is oriented from K_1 to K_2, we define the average and jump of $\mathbf{v}$ on F by

$$\{\!\{\mathbf{v}\}\!\} := \frac{1}{2}(\mathbf{v}|_{K_1} + \mathbf{v}|_{K_2}), \quad [\![\mathbf{v}]\!] := \mathbf{v}|_{K_1} - \mathbf{v}|_{K_2},$$

respectively.
Let $F \in \mathcal{F}_h^{\mathcal{B}} \cap \Gamma_D$, we define $\{\!\{\mathbf{v}\}\!\} := \mathbf{v}$ and $[\![\mathbf{v}]\!] := \mathbf{v}$.
Let $F \in \mathcal{F}_h^{\mathcal{B}} \cap \Gamma_N$, we define $\{\!\{\sigma(\mathbf{v})\mathbf{n}\}\!\} := 0$ and $[\![\sigma(\mathbf{v})\mathbf{n}]\!] := 0$.
We note $|.|$ the measure of an element or a face, and we note h_K or h_F the length of the edges of an element or a face, respectively. Hence, with a Cartesian grid $\forall K \in \mathcal{T}_h, |K| = h_K^d$ and for a side F of an element $K \in \mathcal{T}_h, |F| = h_K^{d-1}$.

The basic idea of the finite element method is to replace the Sobolev spaces on which the variational formulation is posed by a subspace V_h of finite dimension. The better approximation the space V_h is, the better the solution $\mathbf{u}_h$ will approximate the exact solution $\mathbf{u}$. For DG finite element methods, this subspace is always composed of functions whose support is only on one element, which is why we call these methods discontinuous. We usually approximate the space $H^s(\mathcal{T}_h)$ with usual functional spaces.

For a given partition $\mathcal{T}_h$ of Ω, we wish to approximate $\mathbf{u}$ in the finite element space

$$V_h := \{\mathbf{v} \in L^2(\Omega)^d \ : \ \forall K \in \mathcal{T}_h \ \mathbf{v}|_K \in V_h(K)\},$$

where $V_h(K)$ is a finite dimension space approximating $H^s(K)$.

We begin with the second order form of the elastic wave equation

$$-div(\sigma(\mathbf{u})) = f. \tag{1.9}$$

Multiplying (1.9) by a test function $\mathbf{v}_h \in V_h$, we obtain

$$-div(\sigma(\mathbf{u})) \cdot \mathbf{v}_h = f \cdot \mathbf{v}_h.$$

We integrate (1.3.2) on Ω, which gives

$$-\int_{\Omega} div(\sigma(\mathbf{u})) \cdot \mathbf{v}_h \, dx = \int_{\Omega} f \cdot \mathbf{v}_h \, dx.$$

As $\Omega = \bigcup\limits_{K \in \mathcal{T}_h} K$, we have

$$-\sum_{K \in \mathcal{T}_h} \int_K div(\sigma(\mathbf{u})) \cdot \mathbf{v}_h \, dx = \sum_{K \in \mathcal{T}_h} \int_K f \cdot \mathbf{v}_h \, dx.$$

If we use the Theorem A.2 on one element K, we have

$$\int_K div(\sigma(\mathbf{u})) \cdot \mathbf{v}_h \, dx = -\int_K \sigma(\mathbf{u}) \cdot \nabla \mathbf{v}_h \, dx + \int_{\partial K} (\sigma(\mathbf{u})\mathbf{n}) \cdot \mathbf{v}_h \, ds.$$

This is now that all the differences between standard finite element and discontinuous Galerkin finite element methods arise. Standard, or continuous, finite element methods choose carefully the basis of $V_h(K)$ inside each element in such a way that the boundary terms vanishes by imposing the continuity of the basis functions between elements (two neighboring elements share mutual degrees of freedom that impose the continuity between the two elements). DG methods, in contrast, leave these boundary terms, and let the scheme finds the continuities by itself. Hence, for DG methods the continuity between elements is only approximated, whereas it is enforced for standard finite element methods. Enforcing the continuity for standard finite element methods makes it tedious to have high order polynomial basis functions and even more difficult to have *p-adaptivity* or non-conforming meshes, which are needed in our problem.

Let $F = \partial K^+ \cap \partial K^-$, where K^+ and K^- denotes two neighboring elements, thus, we have

$$\sum_{K \in \mathcal{T}_h} \int_{\partial K} (\sigma(\mathbf{u})\mathbf{n}) \cdot \mathbf{v}_h \, ds = \sum_{F \in \mathcal{F}_h} \int_F (\sigma(\mathbf{u}^+)\mathbf{n}^+) \cdot \mathbf{v}_h^+ + (\sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot \mathbf{v}_h^- \, ds.$$

Using the relation $ab + cd = \frac{1}{2}(a+c)(b+d) + \frac{1}{2}(a-c)(b-d)$, we have

$$\begin{aligned}\int_F (\sigma(\mathbf{u}^+)\mathbf{n}^+) \cdot \mathbf{v}_h^+ + (\sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot \mathbf{v}_h^- \, ds = &\int_F \frac{1}{2}(\sigma(\mathbf{u}^+)\mathbf{n}^+ + \sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot (\mathbf{v}_h^+ + \mathbf{v}_h^-) \\ &+ \frac{1}{2}(\sigma(\mathbf{u}^+)\mathbf{n}^+ - \sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot (\mathbf{v}_h^+ - \mathbf{v}_h^-) \, ds.\end{aligned} \tag{1.10}$$

The solution $\mathbf{u} \in \{\tilde{H}_0^1(\Omega)^d : div(\sigma(\mathbf{u})) \in L^2(\Omega)\}$ implies that $[\![\mathbf{u}]\!] = 0$ and $div(\sigma(\mathbf{u})) \in L^2(\Omega)^d$ implies that $[\![\sigma(\mathbf{u})\mathbf{n}]\!] = 0$. Injecting these relations in (1.10) yields
$\forall \mathbf{v}_h \in V_h$,

$$\begin{aligned}\int_F (\sigma(\mathbf{u}^+)\mathbf{n}^+) \cdot \mathbf{v}_h^+ + (\sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot \mathbf{v}_h^- \, ds &= \int_F \frac{1}{2}(\sigma(\mathbf{u}^+)\mathbf{n}^+ - \sigma(\mathbf{u}^-)\mathbf{n}^-) \cdot (\mathbf{v}_h^+ - \mathbf{v}_h^-) \, ds \\ &= \int_F \{\!\{\sigma(\mathbf{u})\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, ds.\end{aligned}$$

At this point we have the following variational formulation:
Find $\mathbf{u}_h \in V_h$ approximation of the exact solution $\mathbf{u}$ such that

$$\forall \mathbf{v}_h \in V_h, \quad a_h(\mathbf{u}_h, \mathbf{v}_h) = l(\mathbf{v}_h),$$

where

$$a_h(\mathbf{u}_h, \mathbf{v}_h) := \sum_{K\in\mathcal{T}_h} \int_K \sigma(\mathbf{u}_h) \cdot \nabla \mathbf{v}_h \, dx - \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\sigma(\mathbf{u}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, ds.$$

Unfortunately, this problem is not equivalent to the model problem since the boundary conditions are not included in this formulation. Moreover, this equation does not verify the inf-sup condition (1.5) because of the unsigned boundary term $\int_F \{\!\{\sigma(\mathbf{u}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, ds$. A sufficient condition to have the inf-sup condition is the *coercivity* of the bilinear form $a_h(.,.)$.

In order to obtain the coercivity of $a_h(.,.)$, a *penalty term* is added. Adding the penalty term (1.11) appears natural when looking at the coercivity proof (see [**?**] or Section 1.3.4). Furthermore, the penalty term imposes weakly the Dirichlet boundary condition. We note that this penalty term is consistent with the model problem since it is null for the exact solution.

$$\sum_{F\in\mathcal{F}_h} \int_F \alpha_F [\![\mathbf{u}_h]\!] \cdot [\![\mathbf{v}_h]\!] \, ds, \tag{1.11}$$

where $\alpha_F \geq 0$.

Remark 1.1. *The penalty term* (1.11) *insures the coercivity by enforcing the continuity of the displacement. There exists a second kind of penalty term (we refer to [**?**]) that penalizes the jumps of derivatives of the displacement*

$$\tilde{\alpha}_F \int_F [\![\sigma(\mathbf{u}_h) \cdot \mathbf{n}]\!] \cdot [\![\sigma(\mathbf{v}_h) \cdot \mathbf{n}]\!],$$

where $\tilde{\alpha}_F \geq 0$.

Another term can be added to obtain the class of *interior penalty discontinuous Galerkin* (IPDG) methods, which writes as

$$\varepsilon \sum_{F\in\mathcal{F}_h} \int_F [\![\mathbf{u}_h]\!] \cdot \{\!\{\sigma(\mathbf{v}_h)\mathbf{n}\}\!\} \, ds,$$

where $\varepsilon \in \{-1, 0, 1\}$. As we shall see this last term has a great impact on the properties of the method, *e.g.* stability, convergence rate.

Finally, the IPDG approximation is
Find $\mathbf{u}_h \in V_h$ such that $\forall \mathbf{v}_h \in V_h$,

$$\sum_{K\in\mathcal{T}_h} \int_K \sigma(\mathbf{u}_h) \cdot \nabla \mathbf{v}_h \, dx - \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\sigma(\mathbf{u}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, ds + \varepsilon \sum_{F\in\mathcal{F}_h} \int_F [\![\mathbf{u}_h]\!] \cdot \{\!\{\sigma(\mathbf{v}_h)\mathbf{n}\}\!\} \, ds$$
$$+ \sum_{F\in\mathcal{F}_h} \int_F \alpha_F [\![\mathbf{u}_h]\!] \cdot [\![\mathbf{v}_h]\!] \, ds = \sum_{K\in\mathcal{T}_h} \int_K f \cdot \mathbf{v}_h \, dx.$$

Remark 1.2. *The* penalty term *has been a really cumbersome parameter all along our work since it has an important impact on the* CFL condition.

1.3.3 Properties of interior penalty discontinuous Galerkin approximations

In this section we first briefly introduce the approximating spaces that are the most commonly used in discontinuous Galerkin methods: the polynomial spaces. However, we could use any other functional spaces, but the convergence of the DG methods would completely change. Then we recall some classical results of IPDG methods with polynomial approximating spaces.

The two most commonly polynomial spaces used in elastodynamic are the following:

P_k polynomial spaces: $P_k(K)$ the space of polynomial of total degree less or equal to k on the element K,

$$P_k(K) := \text{span}\{x_1^{i_1}x_2^{i_2}...x_d^{i_d}, \text{ such that } \sum_{j=1}^{d} i_j \leq k\}.$$

Q_k polynomial spaces: $Q_k(K)$ the space of polynomial of degree at most k in each variable on the element K,

$$Q_k(K) := \text{span}\{x_1^{i_1}x_2^{i_2}...x_d^{i_d}, \text{ such that } \forall j \in 1,..,d, i_j \leq k\}.$$

Remark 1.3. *It is possible to use P_k polynomial bases on any shape of element for DG methods contrary to standard finite element methods. It is especially interesting in our Cartesian grid case since standard finite element methods need to use Q_k basis, and if the polynomial order k is larger than 4 ($k \geq 4$), P_k DG methods have less degrees of freedom than standard finite element methods on the same mesh.*

Once we selected the approximate space, we have to select a basis. The choice of the basis does not influence the properties of the method, but can influence the numerical behavior, in particular the condition number or the sparsity of the stiffness matrix, but these are concerns of implicit methods. Furthermore, we recall that orthogonal basis functions result in a diagonal mass matrix, leading to truly explicit methods.

Remark 1.4. *This freedom in the choice of the basis, because of the lack of continuity constraint, can be exploited to get many interesting properties,* e.g. *hierarchical bases, orthogonal bases, etc...*

We display in Figure 1.1 and 1.2 a representation of two common bases of Q_3, where the points represent the degrees of freedom of Lagrange polynomial bases. These points mean that the value of the solution is equal to the value of the degree of freedom, this is a special case where we do not need to use $\mathbf{u}_h(\mathbf{x}) = \sum_{i=1}^{N_K} u_i^K \varphi_i^K(\mathbf{x})$.

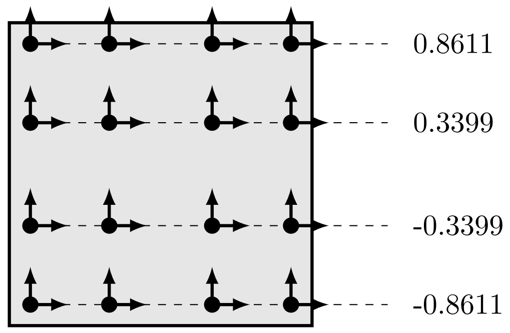


Figure 1.1: Degrees of freedom for Q_3 Legendre-Gauss basis functions.

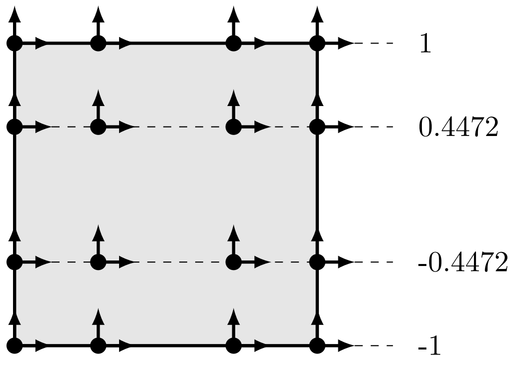


Figure 1.2: Degrees of freedom for Q_3 Legendre-Gauss-Lobatto basis functions.

We now recall the converge results for polynomial approximating spaces. But first we have to define the norms that appear in these results.

We define the *discontinuous Galerkin energy* norm as

$$||u||_h = \left(\sum_{K \in \mathcal{T}_h} \int_K \sigma(u) \cdot \nabla u + \sum_{F \in \mathcal{F}_h} \alpha_F \int_F [\![u]\!] \cdot [\![u]\!] \right)^{\frac{1}{2}},$$

and the broken Sobolev norm as

$$|||\mathbf{v}|||_{H^s(\mathcal{T}_h)} = \left(\sum_{K \in \mathcal{T}_h} ||\mathbf{v}||_{H^s(K)} \right)^{1/2}.$$

We recall the nomenclature of the IPDG methods according to the values of ε and α_F:

- If $\varepsilon = -1$, and α_F is bounded below by a large enough constant, the resulting method is called *symmetric interior penalty discontinuous Galerkin* (SIPDG) method, introduced in the late 1970s by Wheeler [**?**] and Arnold [**?**].

- If $\varepsilon = 1$, the resulting method is called *non-symmetric interior penalty discontinuous Galerkin* (NIPDG) method, introduced in 1999 by Rivière, Wheeler and Girault [**?**]. The particular case with $\alpha_F = 0$ was introduced in 1998 by Oden, Babuska, and Baumann [**?**].

- If $\varepsilon = 0$, and α_F is bounded below by a large enough constant, the resulting method is called *incomplete interior penalty discontinuous Galerkin* (IIPDG) method, introduced in 2004 by Dawson, Sun and Wheeler [**?**].

Theorem 1.2 - Error estimates in the energy norm.
Assume that the exact solution belongs to $H^s(\mathcal{T}_h)$ for $s > 3/2$. Assume also that the penalty parameter α is large enough for the SIPDG and IIPDG methods and that $k \geq 2$ for the NIPDG method with zero penalty. Then, there is a constant C independent of h such that the following optimal a priori error estimate holds:

$$||\mathbf{u} - \mathbf{u}_h||_h \leq Ch^{min(k+1,s)-1}|||\mathbf{u}|||_{H^s(\mathcal{T}_h)}.$$

Theorem 1.3 - Error estimates in the L^2 norm.
Assume that Theorem 1.2 holds. There is a constant C independent of h such that

$$||\mathbf{u} - \mathbf{u}_h||_{L^2(\Omega)} \leq Ch^{min(k+1,s)}|||\mathbf{u}|||_{H^s(\mathcal{T}_h)}.$$

This estimate is valid for the SIPDG method unconditionally. The numerical error for both the NIPDG and IIPDG methods satisfies the following sub-optimal error estimate:

$$||\mathbf{u} - \mathbf{u}_h||_{L^2(\Omega)} \leq Ch^{min(k+1,s)-1}|||\mathbf{u}|||_{H^s(\mathcal{T}_h)}.$$

We refer to [**?**] for a proof of these theorems.

1.3.4 New optimized penalty term and coercivity

In this section, we want to introduce a new penalty and study its impact on the discontinuous Galerkin approximation of the stationary elasticity operator. The idea is to penalize differently normal and tangential parts of displacement in order to avoid an over penalization. In fact, in a homogeneous isotropic medium we can easily see that the normal part is associated with P-waves (that controls the divergence) and the tangential part with the S-waves (that controls the rotational). But the penalization used in the IPDG methods is usually only a function of the P-wave velocity v_P, which is always superior to the velocity of S-waves v_S. Therefore, this causes an "over-penalization" of the tangential part of the displacement. We propose to restore the dependence in v_S for the control of S-waves. This allows us in particular to significantly improve the temporal stability condition, *i.e.* the CFL condition, of the explicit scheme that we will present later in this chapter.

Let the subscripts N and T denote the normal and tangential component of a vector, respectively. We denote $\mathcal{F}_h^b := \mathcal{F}_h^{\mathcal{B}} \cap \Gamma_D$.

First, we state the following lemma, which will help us to reveal the polynomial order dependency in the coercivity constant, and thus the polynomial dependency of the penalty.

Lemma 1.1 - Inverse estimation.
We have the following inverse estimation:
Let $K \in \mathcal{T}_h$ and $\Gamma \subset \partial K$.

$$\forall \mathbf{u}_h \in V_h, \quad ||\mathbf{u}_h||_{L^2(\Gamma)} \leq C_{inv}(p)||\mathbf{u}_h||_{L^2(K)},$$

where p is the polynomial order of the space $V_h(K)$ and $C_{inv}(p) = (p+1)^2$ for square elements.

Our new discontinuous Galerkin approximation is:

$$\begin{aligned} a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{v}_h) := & \int_\Omega \sigma_h(\mathbf{u}_h) : \nabla_h \mathbf{v}_h \, d\mathbf{x} - \int_{\mathcal{F}_h} \{\!\{\sigma_h(\mathbf{u}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma \\ & - \varepsilon \int_{\mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b} [\![\mathbf{u}_h]\!] \cdot \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \, d\gamma \\ & + \int_{\mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b} \alpha_N [\![\mathbf{u}_h]\!]_N [\![\mathbf{v}_h]\!]_N \, d\gamma + \int_{\mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b} \alpha_T [\![\mathbf{u}_h]\!]_T [\![\mathbf{v}_h]\!]_T \, d\gamma, \end{aligned} \tag{1.12}$$

where

$$\begin{aligned} \alpha_N \, : \, \mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b & \to \mathbb{R} \\ \Gamma & \mapsto \alpha_N(\Gamma) = \delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda + 2\mu)\}\!\}}{h_\Gamma}, \\ \alpha_T \, : \, \mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b & \to \mathbb{R} \\ \Gamma & \mapsto \alpha_T(\Gamma) = \delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma}, \end{aligned} \tag{1.13}$$

with δ_N, $\delta_T \geq 0$ two real numbers, h_Γ the measure of the face Γ.

Theorem 1.4 states that under the chosen penalty the bilinear form $a_h^{\text{new},\varepsilon}$ is coercive. As we mention earlier, there is a close link between the coercivity constant C_{coer} and the inf-sup constant β of the relation (1.5). Indeed, we have the relation:

$$\forall \mathbf{u}_h \in V_h, \quad \sup_{\mathbf{v}_h \in V_h} \frac{a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{v}_h)}{||\mathbf{v}_h||_h} \geq \frac{a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{u}_h)}{||\mathbf{u}_h||_h}.$$

Using the coercivity result we get

$$\forall \mathbf{u}_h \in V_h, \quad \sup_{\mathbf{v}_h \in V_h} \frac{a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{v}_h)}{||\mathbf{v}_h||_h} \geq C_{coer}||\mathbf{u}_h||_h.$$

Taking the infinimum we get

$$\inf_{\mathbf{u}_h \in V_h} \sup_{\mathbf{v}_h \in V_h} \frac{a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{v}_h)}{||\mathbf{u}_h||_h ||\mathbf{v}_h||_h} \geq C_{coer}.$$

Thus, with (1.5)

$$\beta \geq C_{coer}.$$

Moreover, if $\exists \mathbf{u}_h \in V_h$ such that $a_h^{\text{new},\varepsilon}(\mathbf{u}_h,\mathbf{u}_h) = C_{coer}||\mathbf{u}_h||_h^2$ then $\beta = C_{coer}$.

This means that the larger the coercivity constant is, the more the method is stable in the sense of the relation (1.7) and the closer the solution is to the optimal solution in V_h according to the relation (1.8).

Theorem 1.4
Given $C_{coer} \in]0,1[$. If $\varepsilon = 0$ or 1 and if we choose the penalty coefficients δ_N and δ_T as

follows:

$$\begin{aligned}
&\bullet \forall F \in \mathcal{F}_h^{\mathcal{I}},\ \delta_N,\, \delta_T \geq \delta_N^* = \delta_T^* := \frac{(1+\varepsilon)^2}{2(1-C_{coer})^2},\\
&\bullet \forall F \in \mathcal{F}_h^{b},\ \delta_N,\, \delta_T \geq \delta_N^* = \delta_T^* := \frac{(1+\varepsilon)^2}{(1-C_{coer})^2}.\\
&\bullet \forall F \in \mathcal{F}_h \cap \Gamma_N,\ \delta_N,\, \delta_T \geq \delta_N^* = \delta_T^* := 0.
\end{aligned} \tag{1.14}$$

Thus,

$$\forall \mathbf{v}_h \in V_h, \quad a_h^{\text{new},\varepsilon}(\mathbf{v}_h, \mathbf{v}_h) \geq C_{coer} \|\mathbf{v}_h\|_h^2, \tag{1.15}$$

where

$$\|\mathbf{v}_h\|_h^2 := \int_\Omega \sigma_h(\mathbf{v}_h) : \nabla_h \mathbf{v}_h \, d\mathbf{x} + \int_{\mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b} \alpha_N [\![\mathbf{v}_h]\!]_N [\![\mathbf{v}_h]\!]_N \, d\gamma + \int_{\mathcal{F}_h^{\mathcal{I}} \cup \mathcal{F}_h^b} \alpha_T [\![\mathbf{v}_h]\!]_T [\![\mathbf{v}_h]\!]_T \, d\gamma.$$

Moreover, if $\varepsilon = -1$ then $\forall \delta_N,\, \delta_T \geq 0$,

$$a_h^{\text{new},\varepsilon}(\mathbf{v}_h, \mathbf{v}_h) \ = \ \|\mathbf{v}_h\|_h^2, \ \forall \mathbf{v}_h \in V_h. \tag{1.16}$$

Proof. In order to prove this result, it suffices to estimate the unsigned term $\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma$ for $\Gamma \in \mathcal{F}_h$.

- First case: $\Gamma = K \cap T \in \mathcal{F}_h^{\mathcal{I}}$. We begin with a decomposition in normal and tangential part of this term:

$$\begin{aligned}
\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma = &\int_\Gamma \{\!\{(\sigma_h(\mathbf{v}_h)\mathbf{n}) \cdot \mathbf{n}\}\!\} [\![\mathbf{v}_h]\!]_N \, d\gamma \\
&+ \int_\Gamma \{\!\{(\sigma_h(\mathbf{v}_h)\mathbf{n}) \cdot \tau\}\!\} [\![\mathbf{v}_h]\!]_T \, d\gamma.
\end{aligned} \tag{1.17}$$

Now, using the Cauchy-Schwarz inequality in $\mathrm{L}^2(\Gamma)$:

$$\begin{aligned}
\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma \leq &\frac{1}{2} \|(\sigma_h(\mathbf{v}_K)\mathbf{n}) \cdot \mathbf{n}\|_{L^2(\Gamma)} \, \|[\![\mathbf{v}_h]\!]_N\|_{L^2(\Gamma)} \\
+ \frac{1}{2} \|(\sigma_h(\mathbf{v}_T)\mathbf{n}) \cdot \mathbf{n}\|_{L^2(\Gamma)} \, \|[\![\mathbf{v}_h]\!]_N\|_{L^2(\Gamma)} &+ \frac{1}{2} \|(\sigma_h(\mathbf{v}_K)\mathbf{n}) \cdot \tau\|_{L^2(\Gamma)} \, \|[\![\mathbf{v}_h]\!]_T\|_{L^2(\Gamma)} \\
&+ \frac{1}{2} \|(\sigma_h(\mathbf{v}_T)\mathbf{n}) \cdot \tau\|_{L^2(\Gamma)} \, \|[\![\mathbf{v}_h]\!]_T\|_{L^2(\Gamma)}.
\end{aligned} \tag{1.18}$$

Since we work with Cartesian grids (with optional refined areas), we immediately get:

$$\begin{aligned}
&(\sigma(\mathbf{v})\mathbf{n}) \cdot \mathbf{n} = \lambda \operatorname{div} \mathbf{v} + 2\mu \partial_z \mathbf{v}_z \text{ with } z = x \text{ if } \mathbf{n} = (\pm 1, 0)^T \text{ and } z = y \text{ otherwise},\\
&(\sigma(\mathbf{v})\mathbf{n}) \cdot \tau = \mu(\partial_2 \mathbf{v}_1 + \partial_1 \mathbf{v}_2).
\end{aligned} \tag{1.19}$$

Introducing (1.19) in (1.18), we get:

$$\begin{aligned}\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!]\, d\gamma \leq & \frac{1}{2}\Big(\|\lambda_K \mathrm{div}\mathbf{v}_K + 2\mu_K \partial_{z_\Gamma}\mathbf{v}_{K,z_\Gamma}\|_{L^2(\Gamma)} \\ & + \|\lambda_T \mathrm{div}\mathbf{v}_T + 2\mu_T \partial_{z_\Gamma}\mathbf{v}_{T,z_\Gamma}\|_{L^2(\Gamma)}\Big) \|[\![\mathbf{v}_h]\!]_N\|_{L^2(\Gamma)} \\ & + \frac{1}{2}\Big(\|\mu_K(\partial_2\mathbf{v}_{K,1} + \partial_1\mathbf{v}_{K,2})\|_{L^2(\Gamma)} \\ & + \|\mu_T(\partial_2\mathbf{v}_{T,1} + \partial_1\mathbf{v}_{T,2})\|_{L^2(\Gamma)}\Big) \|[\![\mathbf{v}_h]\!]_T\|_{L^2(\Gamma)}. \end{aligned} \tag{1.20}$$

Using the inverse estimation (Lemma 1.1), (1.20) becomes:

$$\begin{aligned}\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!]\, d\gamma \leq & \frac{1}{2}\Big(\frac{C_{inv}(p^K)}{h_\Gamma^{1/2}} \|\lambda_K \mathrm{div}\mathbf{v}_K + 2\mu_K \partial_{z_\Gamma}\mathbf{v}_{K,z_\Gamma}\|_{L^2(K)} \\ & + \frac{C_{inv}(p^T)}{h_\Gamma^{1/2}} \|\lambda_T \mathrm{div}\mathbf{v}_T + 2\mu_T \partial_{z_\Gamma}\mathbf{v}_{T,z_\Gamma}\|_{L^2(K)}\Big) \|[\![\mathbf{v}_h]\!]_N\|_{L^2(\Gamma)} \\ & + \frac{1}{2}\Big(\frac{C_{inv}(p^K)}{h_\Gamma^{1/2}} \|\mu_K(\partial_2\mathbf{v}_{K,1} + \partial_1\mathbf{v}_{K,2})\|_{L^2(K)} \\ & + \frac{C_{inv}(p^T)}{h_\Gamma^{1/2}} \|\mu_T(\partial_2\mathbf{v}_{T,1} + \partial_1\mathbf{v}_{T,2})\|_{L^2(K)}\Big) \|[\![\mathbf{v}_h]\!]_T\|_{L^2(\Gamma)}. \end{aligned} \tag{1.21}$$

Applying a triangular inequality to (1.21) yields:

$$\begin{aligned}\int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!]\, d\gamma \leq & \frac{1}{2}\Big(\frac{C_{inv}(p^K)}{h_\Gamma^{1/2}} \lambda_K^{1/2} \|\lambda_K^{1/2} \mathrm{div}\mathbf{v}_K\|_{L^2(K)} \\ & + \frac{C_{inv}(p^K)}{h_\Gamma^{1/2}} (2\mu_K)^{1/2} \|(2\mu_K)^{1/2} \partial_{z_\Gamma}\mathbf{v}_{K,z_\Gamma}\|_{L^2(K)} \\ & + \frac{C_{inv}(p^T)}{h_\Gamma^{1/2}} \lambda_T^{1/2} \|\lambda_T^{1/2} \mathrm{div}\mathbf{v}_T\|_{L^2(T)} \\ & + \frac{C_{inv}(p^T)}{h_\Gamma^{1/2}} (2\mu_T)^{1/2} \|(2\mu_T)^{1/2} \partial_{z_\Gamma}\mathbf{v}_{T,z_\Gamma}\|_{L^2(K)}\Big) \|[\![\mathbf{v}_h]\!]_N\|_{L^2(\Gamma)} \\ & + \frac{1}{2}\Big(\frac{C_{inv}(p^K)}{h_\Gamma^{1/2}} \mu_K^{1/2} \|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1} + \partial_1\mathbf{v}_{K,2})\|_{L^2(K)} \\ & + \frac{C_{inv}(p^T)}{h_\Gamma^{1/2}} \mu_T^{1/2} \|(\mu_T)^{1/2}(\partial_2\mathbf{v}_{T,1} + \partial_1\mathbf{v}_{T,2})\|_{L^2(K)}\Big) \|[\![\mathbf{v}_h]\!]_T\|_{L^2(\Gamma)}. \end{aligned} \tag{1.22}$$

If we sum on all faces of $\mathcal{F}_h^{\mathcal{I}}$ and use Cauchy-Schwarz inequality in $\mathbb{R}^N$ we get:

$$
\begin{aligned}
&\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}} \int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\}\cdot[\![\mathbf{v}_h]\!]\,d\gamma \leq\\
&\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|\lambda_K^{1/2}\mathrm{div}\mathbf{v}_K\|^2_{L^2(K)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^K)^2}{h_\Gamma}\lambda_K\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}\\
&+\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|\lambda_T^{1/2}\mathrm{div}\mathbf{v}_T\|^2_{L^2(T)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^T)^2}{h_\Gamma}\lambda_T\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}\\
&+\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|(2\mu_K)^{1/2}\partial_{z_\Gamma}\mathbf{v}_{K,z_\Gamma}\|^2_{L^2(K)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^K)^2}{h_\Gamma}(2\mu_K)\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}\\
&+\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|(2\mu_T)^{1/2}\partial_{z_\Gamma}\mathbf{v}_{T,z_\Gamma}\|^2_{L^2(T)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^T)^2}{h_\Gamma}(2\mu_T)\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}\\
&+\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1}+\partial_1\mathbf{v}_{K,2})\|^2_{L^2(K)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^K)^2}{h_\Gamma}\mu_K\|[\![\mathbf{v}_h]\!]_T\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}\\
&+\frac{1}{2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\|\mu_T^{1/2}(\partial_2\mathbf{v}_{T,1}+\partial_1\mathbf{v}_{T,2})\|^2_{L^2(T)}\Bigg)^{1/2}\Bigg(\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{C_{inv}(p^T)^2}{h_\Gamma}\mu_T\|[\![\mathbf{v}_h]\!]_T\ \|^2_{L^2(\Gamma)}\Bigg)^{1/2}.
\end{aligned}
\tag{1.23}
$$

Finally, using Young's inequality $ab\leq\xi^2a^2+\frac{1}{4\xi^2}b^2$ we get:

$$
\begin{aligned}
\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}} \int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\}\cdot[\![\mathbf{v}_h]\!]\,d\gamma \leq &\frac{\xi^2}{2}\sum_{K\in\mathcal{T}_h}C(K)(\|\lambda_K^{1/2}\mathrm{div}\mathbf{v}_K\|^2_{L^2(K)}\\
&+\|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1}+\partial_1\mathbf{v}_{K,2})\|^2_{L^2(K)})\\
&+\frac{\xi^2}{2}\sum_{K\in\mathcal{T}_h}\widetilde{C}(K)\|(2\mu_K)^{1/2}\partial_1\mathbf{v}_{K,1}\|^2_{L^2(K)}\\
&+\frac{\xi^2}{2}\sum_{K\in\mathcal{T}_h}\widetilde{C}'(K)\|(2\mu_K)^{1/2}\partial_2\mathbf{v}_{K,2}\|^2_{L^2(K)}\\
&+\frac{1}{4\xi^2}\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{1}{h_\Gamma}\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\\
&+\frac{1}{4\xi^2}\sum_{\substack{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\\ \Gamma=K\cap T}}\frac{1}{h_\Gamma}\{\!\{C_{inv}(p)^2\mu\}\!\}\|[\![\mathbf{v}_h]\!]_T\ \|^2_{L^2(\Gamma)},
\end{aligned}
\tag{1.24}
$$

where $C(K) \leq 4$ is the cardinal number of $\partial K \cap \mathcal{F}_h^{\mathcal{I}}$, $\widetilde{C}(K) \leq 2$ is the cardinal number of the set of vertical faces of $\partial K \cap \mathcal{F}_h^{\mathcal{I}}$ and $\widetilde{C}'(K) \leq 2$ is the cardinal number of the set of horizontal faces of $\partial K \cap \mathcal{F}_h^{\mathcal{I}}$.

- Second case: $\Gamma \in \mathcal{F}_h^b$ such that $\Gamma \subset \partial K$. Proceeding as in the first case, we immediately get:

$$\begin{aligned}
\sum_{\substack{\Gamma \in \mathcal{F}_h^b \\ \Gamma \subset \partial K}} \int_\Gamma \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma \leq & \xi_b^2 \sum_{K \in \mathcal{T}_h} C_b(K) (\|\lambda_K^{1/2} \mathrm{div} \mathbf{v}_K\|_{L^2(K)}^2 \\
& + \|\mu_K^{1/2}(\partial_2 \mathbf{v}_{K,1} + \partial_1 \mathbf{v}_{K,2})\|_{L^2(K)}^2) \\
& + \xi_b^2 \sum_{K \in \mathcal{T}_h} \widetilde{C}_b(K) \|(2\mu_K)^{1/2} \partial_1 \mathbf{v}_{K,1}\|_{L^2(K)}^2 \\
& + \xi_b^2 \sum_{K \in \mathcal{T}_h} \widetilde{C}_b'(K) \|(2\mu_K)^{1/2} \partial_2 \mathbf{v}_{K,2}\|_{L^2(K)}^2 \\
& + \frac{1}{4\xi_b^2} \sum_{\substack{\Gamma \in \mathcal{F}_h^b \\ \Gamma \subset \partial K}} \frac{1}{h_\Gamma} \{\!\{C_{inv}(p)^2(\lambda + 2\mu)\}\!\} \|[\![\mathbf{v}_h]\!]_N \|_{L^2(\Gamma)}^2 \\
& + \frac{1}{4\xi_b^2} \sum_{\substack{\Gamma \in \mathcal{F}_h^b \\ \Gamma \subset \partial K}} \frac{1}{h_\Gamma} \{\!\{C_{inv}(p)^2 \mu\}\!\} \|[\![\mathbf{v}_h]\!]_T \|_{L^2(\Gamma)}^2,
\end{aligned} \tag{1.25}$$

where $C_b(K) \leq 4$ is the cardinal number of $\partial K \cap \mathcal{F}_h^b$, $\widetilde{C}_b(K) \leq 2$ is the cardinal number of the set of vertical faces of $\partial K \cap \mathcal{F}_h^b$ and $\widetilde{C}_b'(K) \leq 2$ is the cardinal number of the set of horizontal faces of $\partial K \cap \mathcal{F}_h^b$.

- Using the definition of the isotropic stress tensor, we get:

$$\begin{aligned}
\int_\Omega \sigma_h(\mathbf{v}_h) : \nabla_h \mathbf{v}_h \, d\mathbf{x} = & \sum_{K \in \mathcal{T}_h} (\|\lambda_K^{1/2} \mathrm{div} \mathbf{v}_K\|_{L^2(K)}^2 \\
& + \|(2\mu_K)^{1/2} \partial_1 \mathbf{v}_{K,1}\|_{L^2(K)}^2 + \|(2\mu_K)^{1/2} \partial_2 \mathbf{v}_{K,2}\|_{L^2(K)}^2 \\
& + \|\mu_K^{1/2}(\partial_2 \mathbf{v}_{K,1} + \partial_1 \mathbf{v}_{K,2})\|_{L^2(K)}^2).
\end{aligned} \tag{1.26}$$

If we look now the coercivity of the form $a_h^{\mathrm{new},\varepsilon}$:

Using (1.24), (1.25) et (1.26), we have

$$\begin{aligned}
a_h^{\mathrm{new},\varepsilon}(\mathbf{v}_h,\mathbf{v}_h) &= \int_\Omega \sigma_h(\mathbf{v}_h) : \nabla_h \mathbf{v}_h \, d\mathbf{x} - (1+\varepsilon)\int_{\mathcal{F}_h} \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\cup\mathcal{F}_h} \delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\cup\mathcal{F}_h} \delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)} \\
&\geq \sum_{K\in\mathcal{T}_h} \Big((1-(1+\varepsilon)(\xi_b^2 C_b(K) + \frac{\xi^2}{2}C(K)))\|\lambda_K^{1/2}\mathrm{div}\mathbf{v}_K\|^2_{L^2(K)} \\
&\quad + (1-(1+\varepsilon)(\xi_b^2 \widetilde{C}_b(K) + \frac{\xi^2}{2}\widetilde{C}(K)))\|(2\mu_K)^{1/2}\partial_1\mathbf{v}_{K,1}\|^2_{L^2(K)} \\
&\quad + (1-(1+\varepsilon)(\xi_b^2 \widetilde{C}'_b(K) + \frac{\xi^2}{2}\widetilde{C}'(K)))\|(2\mu_K)^{1/2}\partial_2\mathbf{v}_{K,2}\|^2_{L^2(K)} \\
&\quad + (1-(1+\varepsilon)(\xi_b^2 C_b(K) + \frac{\xi^2}{2}C(K)))\|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1}+\partial_1\mathbf{v}_{K,2})\|^2_{L^2(K)} \Big) \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}} (1-\frac{(1+\varepsilon)}{4\xi^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}} (1-\frac{(1+\varepsilon)}{4\xi^2\delta_T})\delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)} \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{b}} (1-\frac{(1+\varepsilon)}{4\xi_b^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&\quad + \sum_{\Gamma\in\mathcal{F}_h^{b}} (1-\frac{(1+\varepsilon)}{4\xi_b^2\delta_T})\delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)}.
\end{aligned} \tag{1.27}$$

Choosing $\xi_b^2 = \xi^2/2$, we get

$$(1-(1+\varepsilon)(\xi_b^2 C_b(K) + \frac{\xi^2}{2}C(K))) = 1-4(1+\varepsilon)\xi^2/2,$$

$$(1-(1+\varepsilon)(\xi_b^2 \widetilde{C}_b(K) + \frac{\xi^2}{2}\widetilde{C}(K))) = 1-2(1+\varepsilon)\xi^2/2,$$

and

$$(1-(1+\varepsilon)(\xi_b^2 \widetilde{C}'_b(K) + \frac{\xi^2}{2}\widetilde{C}'(K))) = 1-2(1+\varepsilon)\xi^2/2.$$

To get a coercivity constant $C_{coer}\in]0,1[$ when $\varepsilon\neq -1$, we have to choose

$$\begin{aligned}
&\bullet\forall\Gamma\in\mathcal{F}_h^{\mathcal{I}},\ \delta_N,\,\delta_T \geq \delta_N^* = \delta_T^* := \frac{(1+\varepsilon)^2}{2(1-C_{coer})^2}, \\
&\bullet\forall\Gamma\in\mathcal{F}_h^{b},\ \delta_N,\,\delta_T \geq \delta_N^* = \delta_T^* := \frac{(1+\varepsilon)^2}{(1-C_{coer})^2}.
\end{aligned} \tag{1.28}$$

□

Remark 1.5. • *Dirichlet boundary condition implies a penalty two times larger on the faces of $\mathcal{F}_h^b$ than on the faces of $\mathcal{F}_h^{\mathcal{I}}$ (See [?] or the coercivity proof of Theorem 1.4),*

- *If $\varepsilon = 0$ or -1, getting a better coercivity constant implies rising the penalty. Moreover, we have the limit case: $C_{coer} \to 1^- \Rightarrow \delta_N,\, \delta_T \to +\infty$,*

- *If $\varepsilon = 1$, $C_{coer} = 1$ for all $\delta_N,\, \delta_T \geq 0$.*

Remark 1.6. *A priori error results from Section 1.3.3 can easily be extended to this approximation with the new penalty.*

1.4 The interior penalty discontinuous Galerkin methods for the elastodynamic equation in the time domain

In this section, we introduce the IPDG approximation for the model problem (1.1), that is the elastodynamic equation in the time domain. Thus we briefly give the semi-discrete IPDG approximation in space for the elastodynamic equation in the time domain from the previous section. Then, we discretize in time the equation with a standard leap-frog finite difference scheme.

1.4.1 Semi-discrete IPDG approximation

The general semi-discrete IPDG approximation of the model problem (1.1) is

Find $\forall t \in [0,T], \mathbf{u}_h(.,t) \in V_h$ such that

$$\begin{cases} (\partial_{tt}\mathbf{u}_h, \mathbf{v}_h) + a_h(\mathbf{u}_h, \mathbf{v}_h) = (f, \mathbf{v}_h), \quad \forall \mathbf{v}_h \in V_h, \forall t \in [0,T], \\ \mathbf{u}_h|_{t=0} = \Pi_h u_0, \\ \partial_t \mathbf{u}_h|_{t=0} = \Pi_h v_0, \end{cases} \tag{1.29}$$

where Π_h denotes the L^2-projection onto V_h and the discrete bilinear form a_h on $V_h \times V_h \to \mathbb{R}$ is given by

$$a_h(\mathbf{u},\mathbf{v}) = \sum_{K\in\mathcal{T}_h} \int_K \sigma_h(\mathbf{u}) : \nabla\mathbf{v}\, dx - \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\sigma_h(\mathbf{u})\mathbf{n}\}\!\} \cdot [\![\mathbf{v}]\!]\, d\gamma + \varepsilon \sum_{F\in\mathcal{F}_h} \int_F [\![\mathbf{u}]\!] \cdot \{\!\{\sigma_h(\mathbf{v})\mathbf{n}\}\!\}\, d\gamma$$
$$+ \sum_{F\in\mathcal{F}_h} \int_F \alpha_N [\![\mathbf{u}]\!]_N \cdot [\![\mathbf{v}]\!]_N\, d\gamma + \sum_{F\in\mathcal{F}_h} \int_F \alpha_T [\![\mathbf{u}]\!]_T \cdot [\![\mathbf{v}]\!]_T\, d\gamma.$$

Let $K \in \mathcal{T}_h$, we dneote by $\{\phi_i^K\}$ a basis of $V_h(K)$. Let $N_K = |\{\phi_i^K\}|$ be the number of degrees of freedom on element K and $N = \sum_{K\in\mathcal{T}_h} N_K$ is the total number of degrees of freedom.

The semi-discrete solution can be expanded in the global basis functions by

$$\forall t \in [0,T],\ \forall x \in \Omega, \quad \mathbf{u}_h(t,\mathbf{x}) = \sum_{K\in\mathcal{T}_h} \sum_{i=1}^{N_K} U_i^K(t)\phi_i^K(\mathbf{x}). \tag{1.30}$$

We note $\mathbf{U} := (U_i)_{1\leq i\leq N}$. The semi-discrete IPDG formulation (1.29) is equivalent to the second-order system of ordinary differential equations

$$\begin{cases} M\dfrac{d^2\mathbf{U}}{dt^2} + K\mathbf{U} = F, \\ \mathbf{U}(0) = \mathbf{U}_0, \\ \dfrac{d\mathbf{U}}{dt}(0) = \mathbf{V}_0, \end{cases}$$

where $M = (M_{ij})_{ij}$ is the $N \times N$ mass matrix, and $K = (K_{ij})_{ij}$ is the $N \times N$ stiffness matrix, and they are defined by

$$\forall i,j \in [\![1,N]\!] \quad M_{ij} = (\phi_j, \phi_i)_\Omega, \quad K_{ij} = a_h(\phi_j,\phi_i).$$

Remark 1.7. *Because of the lack of continuity constraints between mesh elements for the test functions, the basis functions have a support contained in one element. Therefore, the mass matrix is always block diagonal, and diagonal if we choose orthogonal basis functions. In contrast, for standard finite element methods the mass matrix has an arbitrary structure depending on the element indexing in the mesh, preventing these methods to be directly implemented in a truly explicit way since the mass matrix has to be inverted. This problem can be circumvented for low polynomial orders by sophisticated techniques of mass lumping. In our case of quadrilateral meshes mass lumping techniques are well understood for an arbitrary order, and lead to so-called* spectral element methods*.*

1.4.1.1 Local DG formulation

Here, we introduce a local formulation of the DG approximation. This local formulation shows why the mass matrix is always block diagonal, and why parallelizing DG methods is straightforward, this formulation is also useful for some theoretical studies of DG schemes.

We denote by $V_F(K)$ the neighboring element of the element K on a face F. We introduce the local bilinear form a_h^K:

$$\begin{aligned} a_h^K(\mathbf{u},\mathbf{v}) := &\int_K \sigma_h^K(\mathbf{u}) : \nabla\mathbf{v}|_K\, dx - \sum_{F\in\mathcal{F}_K} \int_F \{\!\{\sigma_h(\mathbf{u})\mathbf{n}\}\!\} \cdot \mathbf{v}|_K\, d\gamma + \varepsilon \sum_{F\in\mathcal{F}_K} \int_F \beta[\![\mathbf{u}]\!] \cdot \sigma_h^K(\mathbf{v})\mathbf{n}_F\, d\gamma \\ &+ \sum_{F\in\mathcal{F}_K} \int_F \alpha_N [\![\mathbf{u}]\!]_N \cdot \mathbf{v}|_K\, d\gamma + \sum_{F\in\mathcal{F}_K} \int_F \alpha_T [\![\mathbf{u}]\!]_T \cdot \mathbf{v}|_K\, d\gamma, \end{aligned} \tag{1.31}$$

where $\sigma_h^K := \sigma_h|_K$, and we have the following relations for β

$$\beta = \begin{cases} \frac{1}{2} & \text{on } \mathcal{F}_h^{\mathcal{I}}, \\ 1 & \text{on } \mathcal{F}_h \cap \Gamma_D, \\ 0 & \text{on } \mathcal{F}_h \cap \Gamma_N. \end{cases}$$

Thus, we have $a_h(\mathbf{u},\mathbf{v}) = \sum_{K\in\mathcal{T}_h} a_h^K(\mathbf{u},\mathbf{v})$.

Semi-discrete local DG approximation: We can rewrite the semi-discrete DG approximation in a local way

$$\forall K \in \mathcal{T}_h, \quad M^K \partial_{tt} \mathbf{u}^K + K^K \mathbf{u}^K + \sum_{F \in \mathcal{F}_K} F^{V_F(K)} \mathbf{u}^{V_F(K)} = \ell^K,$$

where

$$u_h^K(t, \mathbf{x}) := \sum_{i=1}^{N_K} u_i^K(t) \varphi_i^K(\mathbf{x}), \quad \text{and} \quad \mathbf{u}^K := \left(u_i^K \right)_{1 \le i \le N^K},$$

and

$$M_{ij}^K := \rho_K(\varphi_j^K, \varphi_i^K)_K, \quad K_{ij}^K := a_h^K(\varphi_j^K, \varphi_i^K), \quad F_{ij}^{V_F(K)} := a_h^K(\varphi_j^{V_F(K)}, \varphi_i^K),$$

and

$$\ell_j^K := \ell(\varphi_j^K).$$

1.4.2 Full discretization of the discontinuous Galerkin approximation

After discretizing the equation in space with a discontinuous Galerkin method, we finish the discretization of the problem using a finite difference method in time. This form is called the fully discretized IPDG formulation. We note by $\mathbf{U}^n$ the approximation of $\mathbf{U}(t_n)$ using the well-known finite difference second-order leap frog scheme for temporal derivatives. Hence, we get

$$M \frac{\mathbf{U}^{n+1} - 2\mathbf{U}^n + \mathbf{U}^{n-1}}{\Delta t} + K\mathbf{U}^n = F^n. \tag{1.32}$$

Full discrete local DG approximation: In the same way, we can rewrite the full discrete DG approximation

$$\forall K \in \mathcal{T}_h, \quad M^K \frac{\mathbf{u}_{n+1}^K - 2\mathbf{u}_n^K + \mathbf{u}_{n-1}^K}{\Delta t^2} + K^K \mathbf{u}_n^K + \sum_{F \in \mathcal{F}_K} F^{V_F(K)} \mathbf{u}_n^{V_F(K)} = l^K,$$

where

$$u_h^K(t_n, \mathbf{x}) := \sum_{i=1}^{N_K} u_{n,i}^K \varphi_i^K(\mathbf{x}), \quad \text{and} \quad \mathbf{u}_n^K := \left(u_{n,i}^K \right)_{1 \le i \le N^K}.$$

1.5 Plane wave analysis

The plane wave analysis [**?**], although based on simplified problems, *i.e.* infinite homogeneous medium, provides accurate information about the properties of a numerical method. This information is precise enough to be used in real simulations. It helps to apprehend two majors properties: dispersion and stability. The dispersion is a numerical phenomenon that creates a phase difference between the physical wave and the numerical wave, *i.e.* the numerical velocity only approximates the physical velocity. The dispersion is used to determine the spatial discretization according to the desired precision, *i.e.* the number of elements per wavelength that must be used to achieve the desired accuracy. Stability is given by a CFL condition which is a relation between the time step Δt and the space step h of the form $\frac{\Delta t}{h} \le C$, where C is a constant that depends on physical and numerical parameters (dimension, polynomial approximation order, velocities).

The principle of a plane wave analysis is to seek the conditions, in the form of a discrete dispersion relation, for which a numerical plane wave is a solution of the scheme. Plane waves provide an accurate analysis because they constitute a basis of solution to the infinite homogeneous elastodynamic problem.

1.5.1 Dispersion relation formulation

In geoscience, having the correct propagation velocity is a major aspect. Since direct propagations (the forward problem) are often used in the iterations of an inverse problem to know the structure of the ground, errors in propagation velocities result in bad ground imaging. Therefore, having a good control on the dispersion error is critical.

In order to get the dispersion relation, we begin with the local semi-discrete DG approximation in which we inject plane waves. By doing so, we get simple relations between all degrees of freedom. After some algebraic manipulations, we get a generalized eigenvalue problem that reveals which modes our numerical method propagates. Since a plane wave is monotonic our method should propagate only one mode, however the eigenvalue analysis reveals that more than one mode is propagated.

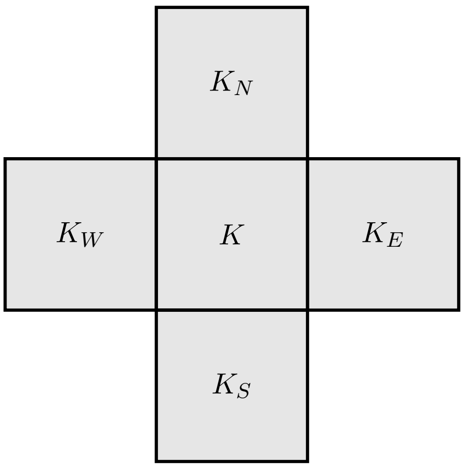


Figure 1.3: Neighboring elements of the element K.

We formulate the dispersion relation in an arbitrary dimension since the process is identical for any dimension.

The local DG approximation (see Section 1.4.1.1) is

$$M\partial_{tt}\mathbf{u}^K + K\mathbf{u}^K + \sum_{f\in\mathcal{F}_K} F^f\mathbf{u}^{V_f(K)} = 0, \tag{1.33}$$

where

$$u_h^K(t,\mathbf{x}) = \sum_{i=1}^{N_K} u_i^K(t)\varphi_i^K(\mathbf{x}), \quad \text{with} \quad \mathbf{u}^K = \left(u_i^K\right)_{1\le i\le N^K},$$

and

$$M_{ij} = \rho_K(\varphi_j^K,\varphi_i^K)_K, \quad K_{ij} = a_h^K(\varphi_j^K,\varphi_i^K), \quad F_{ij}^f = a_h^K(\varphi_j^{V_f(K)},\varphi_i^K).$$

Since the displacement is a plane wave, then

$$u_j^K = A_j e^{-i(\mathbf{k}\cdot\mathbf{x}-\omega_h t)},$$

where $\mathbf{k}$ is the wavenumber, ω_h the pulsation and A_j the amplitude.
The plane wave assumption implies that

$$\mathbf{u}^{V_f(K)} = e^{i\mathbf{k}\cdot\mathbf{x}_f}\mathbf{u}^K, \tag{1.34}$$

where

$$\mathbf{x}_E = h\mathbf{e}_x, \quad \mathbf{x}_W = -h\mathbf{e}_x, \quad \mathbf{x}_N = h\mathbf{e}_y, \quad \mathbf{x}_S = -h\mathbf{e}_y, \quad \mathbf{x}_T = -h\mathbf{e}_z, \quad \mathbf{x}_B = h\mathbf{e}_z.$$

Injecting (1.34) in (1.33) yields the following generalized eigenvalue problem:

$$\omega_h^2 M\mathbf{u}^K + \left(K + \sum_{f\in\mathcal{F}_K} e^{i\mathbf{k}\cdot\mathbf{x}_f}F^f\right)\mathbf{u}^K = 0.$$

We choose a space step such that $h = \dfrac{\lambda}{N}$, where λ is the wavelength and $N \in \mathbb{N}^*$. Let $\mathbf{k} = k\mathbf{d}$, where $\mathbf{d}$ is a unit vector representing the direction of the wave. We introduce $\varkappa := \dfrac{1}{(p+1)N}$, which corresponds to the inverse of points per wavelength, where p is the order of the polynomial space.

We get the relation

$$kh = 2\pi(p+1)\varkappa.$$

The eigenvalue problem becomes:

$$-h^2\omega_h^2\hat{M}\mathbf{u}^K + \left(\hat{K} + \sum_{f\in\mathcal{F}_K} e^{ikh(\mathbf{d}\cdot\mathbf{e}_f)}\hat{F}^f\right)\mathbf{u}^K = 0,$$

where $\hat{M} := \frac{1}{h^d}M$, $\hat{K} := \frac{1}{h^{2-d}}K$ and $\hat{F}^f := \frac{1}{h^{2-d}}F^f$.

We rewrite this problem as a function of $\varkappa$:

$$-(2\pi)^2(p+1)^2\varkappa^2\frac{\omega_h^2}{k^2}\hat{M}\mathbf{u}^K + \left(\hat{K} + \sum_{f\in\mathcal{F}_K} e^{i(2\pi)(p+1)\varkappa(\mathbf{d}\cdot\mathbf{e}_f)}\hat{F}^f\right)\mathbf{u}^K = 0.$$

We note that $\dfrac{\omega_h^2}{k^2}$ is an eigenvalue of the generalized eigenvalue problem:

$$\frac{\omega_h^2}{k^2}\hat{M}V = AV,$$

where

$$A = \frac{1}{(2\pi)^2(p+1)^2\varkappa^2}\left(\hat{K} + \sum_{f\in\mathcal{F}_K} e^{i(2\pi)(p+1)\varkappa(\mathbf{d}\cdot\mathbf{e}_f)}\hat{F}^f\right).$$

At this point we need to identify which modes correspond to the P-waves and S-waves, since the number of eigenvalues exceeds the number of physical modes

$$v_h = \frac{\omega_h}{k}.$$

We define the dispersion error as follow:

$$e_p = \left| \frac{v_h}{v_p} - 1 \right|, \quad e_s = \left| \frac{v_h}{v_s} - 1 \right|,$$

where v_h is the numerical velocity of the mode given by the eigenvalue, and v_p and v_s are the expected velocities associated to P- and S-waves.

The physical modes are (most of time) the two values for which e_p and e_s are the closest to zero. Sometimes, a non-physical mode is closer to the physical mode than the approximated mode, but this happens only for some angles of incidence as we shall see in the next section.

The dispersions e_p and e_s depend on the parameters p the polynomial order of the approximating space, κ the number of points per wavelength and $\mathbf{d}$ the direction of the waves.

1.5.2 Dispersion analysis

In this section, we apply a dispersion analysis to show the numerical properties of the dispersion. We used $v_p = 2600 m.s^{-1}$, $v_s = 1300 m.s^{-1}$ and $\rho = 2300 kg.m^{-3}$ and a penalty parameter $\delta_N = \delta_T = 2$.

On Figure 1.4 and 1.5 we display the convergence of the maximal angular dispersion error ($\max_{\mathbf{u}} |e_p|$ and $\max_{\mathbf{u}} |e_s|$) according to the number of points per wavelength $\frac{1}{\varkappa}$ for different polynomial spaces. We observe that the convergence rates of the dispersion errors are $|e_p| = \mathcal{O}(h^{2k})$ and $|e_s| = \mathcal{O}(h^{2k})$, where k is the polynomial order of the space Q_k.

Remark 1.8. *For NIPDG and IIPDG methods the dispersion errors convergence rates are $k+1$ for odd orders and k for even orders [**?**].*

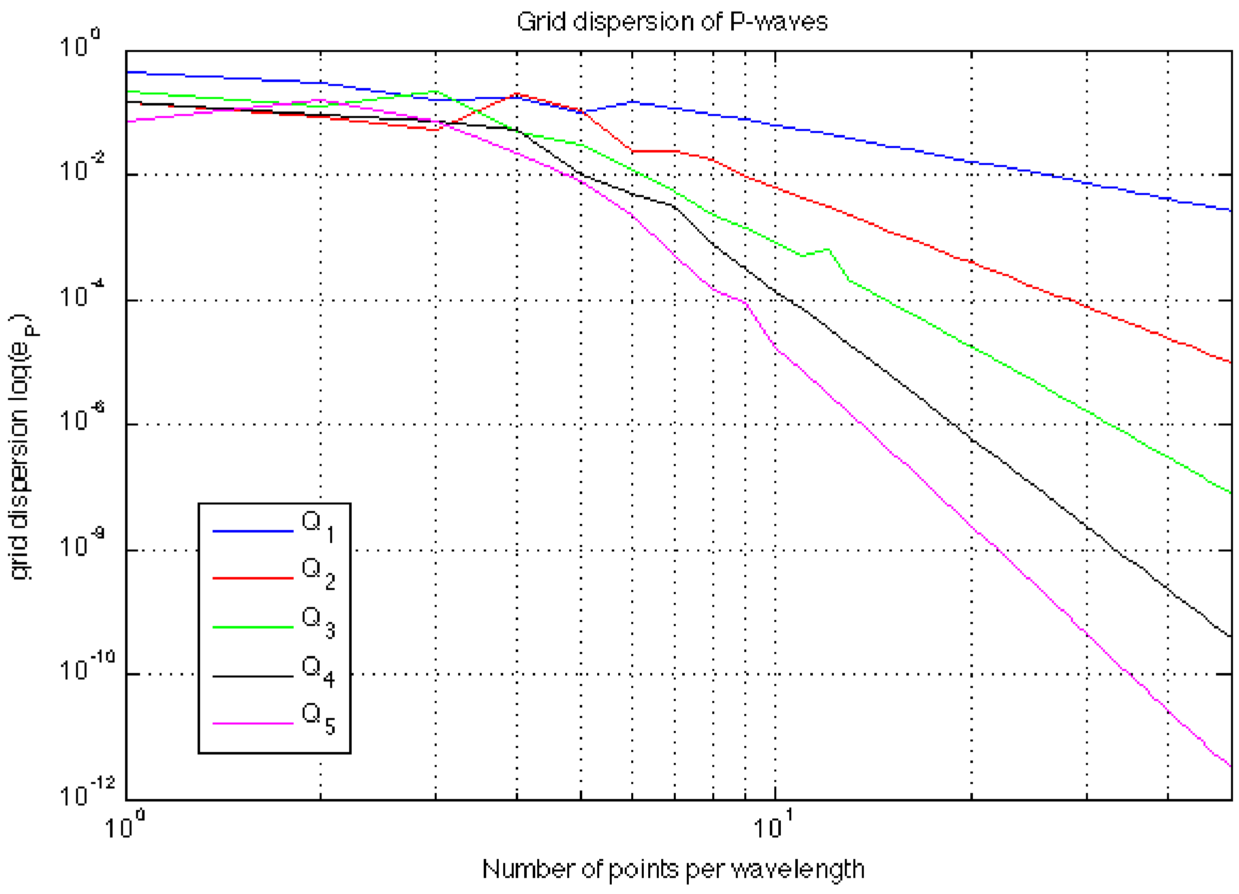


Figure 1.4: Dispersion convergence for P-waves for different polynomial bases according to the number of points per wavelength $\frac{1}{\varkappa}$.

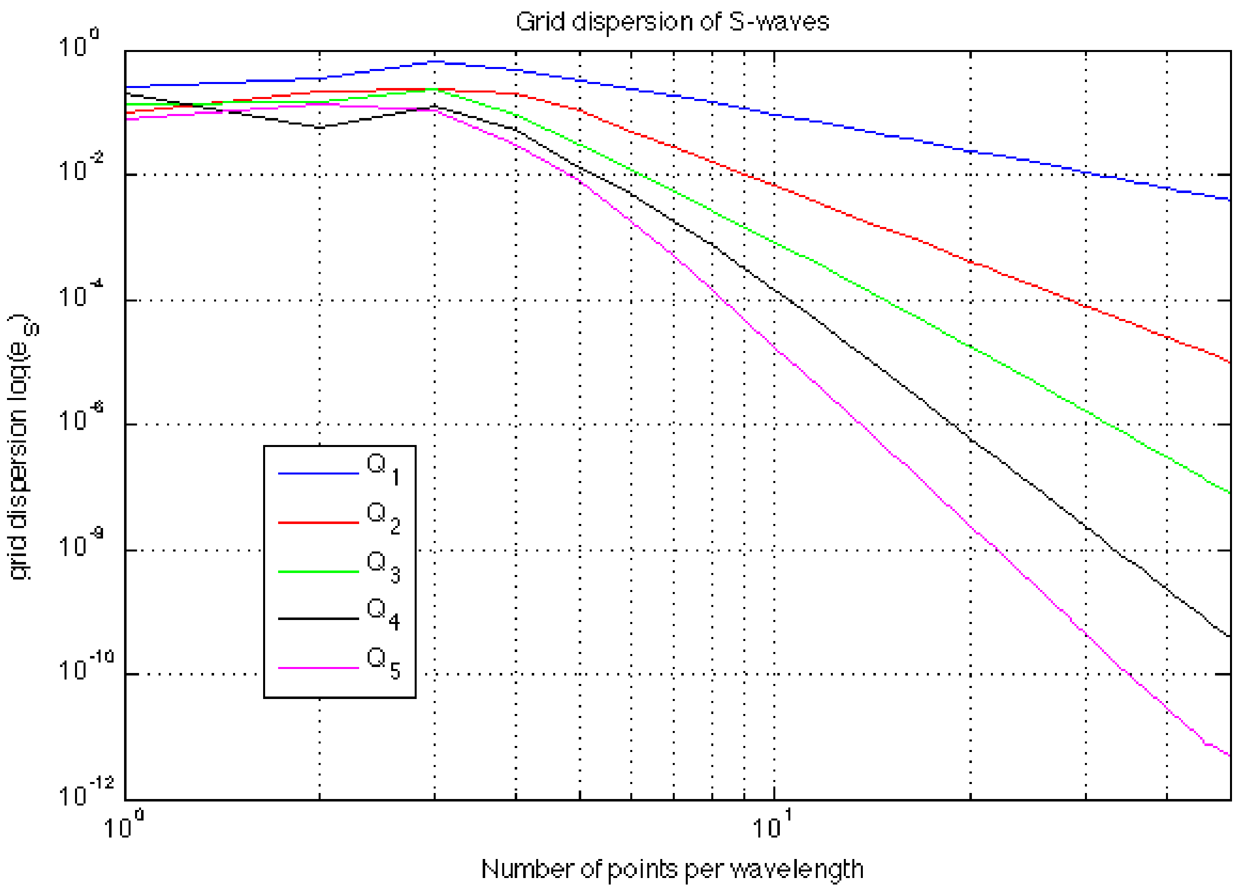


Figure 1.5: Dispersion convergence for S-waves for different polynomial basis according to the number of points per wavelength $\dfrac{1}{\varkappa}$.

On Figure 1.6 to 1.10 we display the dispersion for Gauss-Legendre bases with optimized penalty and roughly the same maximal angular dispersion ($e_s \simeq 1 \times 10^{-2}$) deduced from Figure 1.4 and 1.5. As we can see, we need 30 points per wavelength in Q_1, 9 points per wavelength in Q_2, 6 points per wavelength in Q_3, 5 points per wavelength in Q_4 and 5 points per wavelength in Q_5 to achieve a dispersion error of 10^{-2}. Discontinuities that we can see in Figure 1.9 and 1.10 are due to a non-physical mode being closer to the real physical mode than the approximated mode in a specific direction. For this reason, these discontinuities should not be interpreted as a discontinuity in the shape of the dispersion.

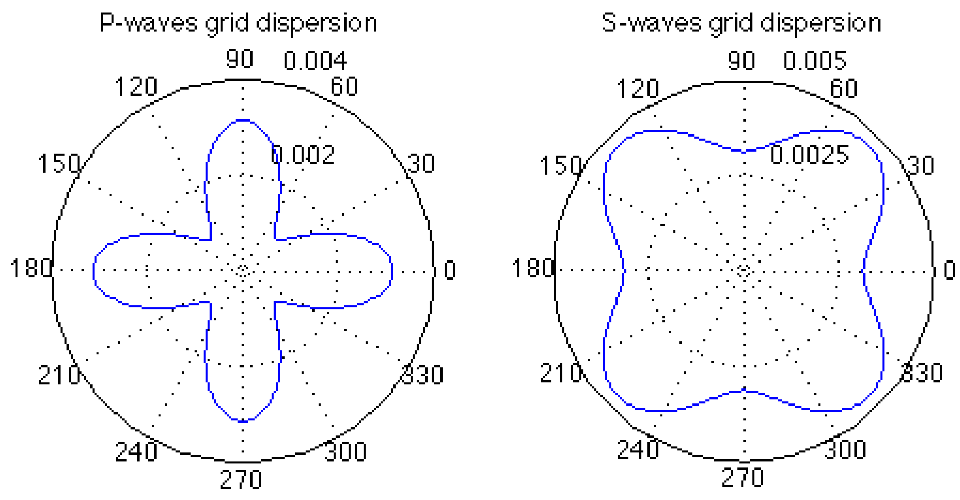


Figure 1.6: Dispersion anisotropy for Q_1 elements with 30 points per wavelength.

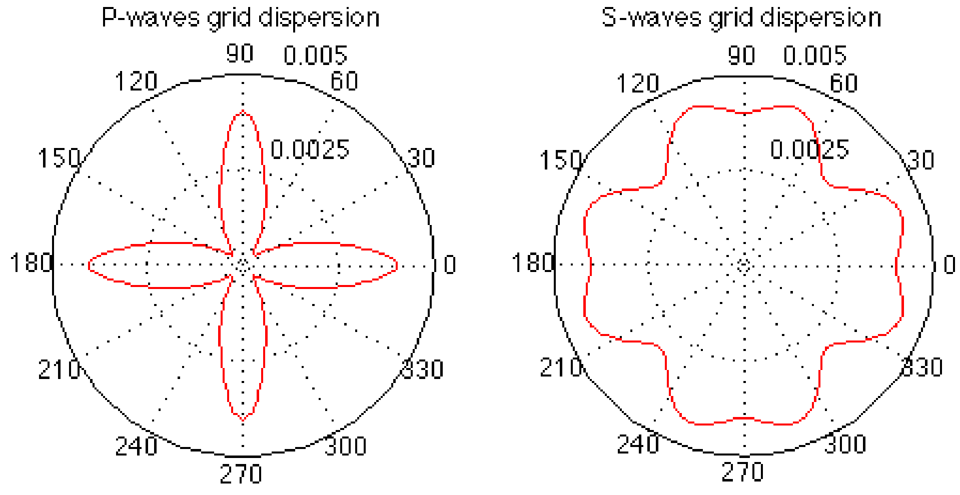


Figure 1.7: Dispersion anisotropy for Q_2 elements with 9 points per wavelength.

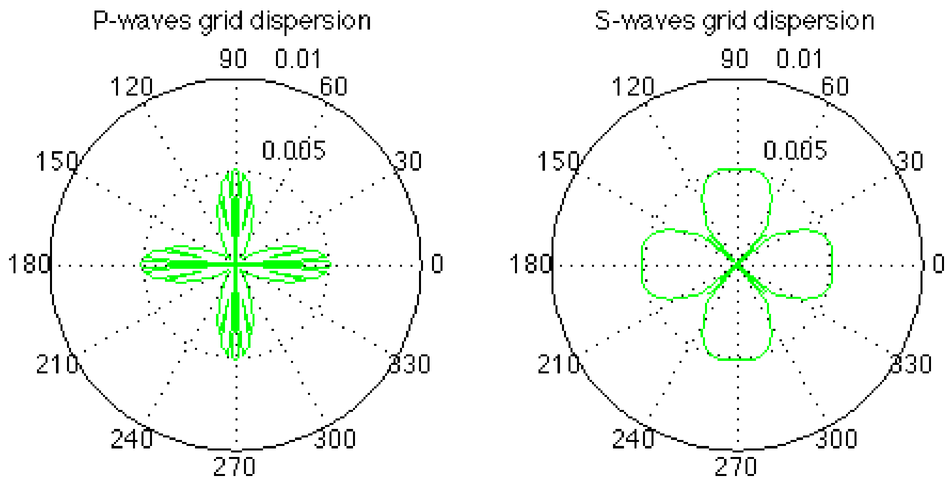


Figure 1.8: Dispersion anisotropy for Q_3 elements with 6 points per wavelength.

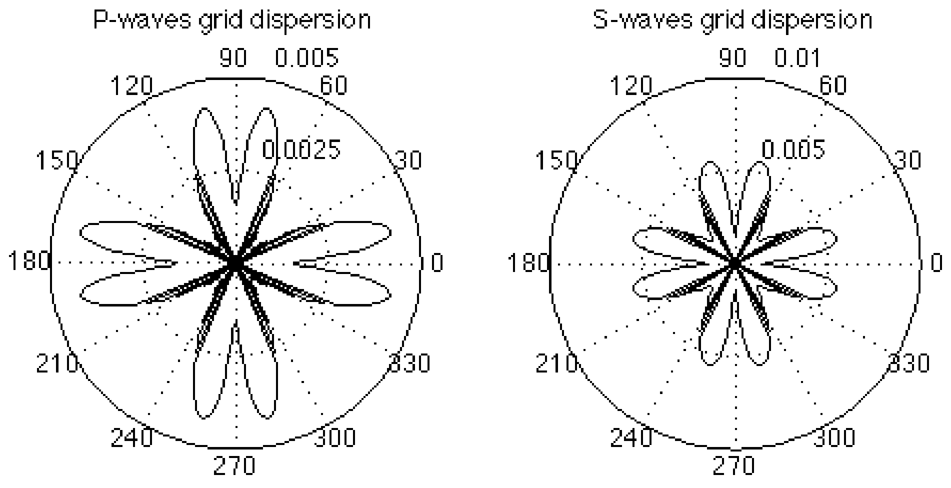


Figure 1.9: Dispersion anisotropy for Q_4 elements with 5 points per wavelength.

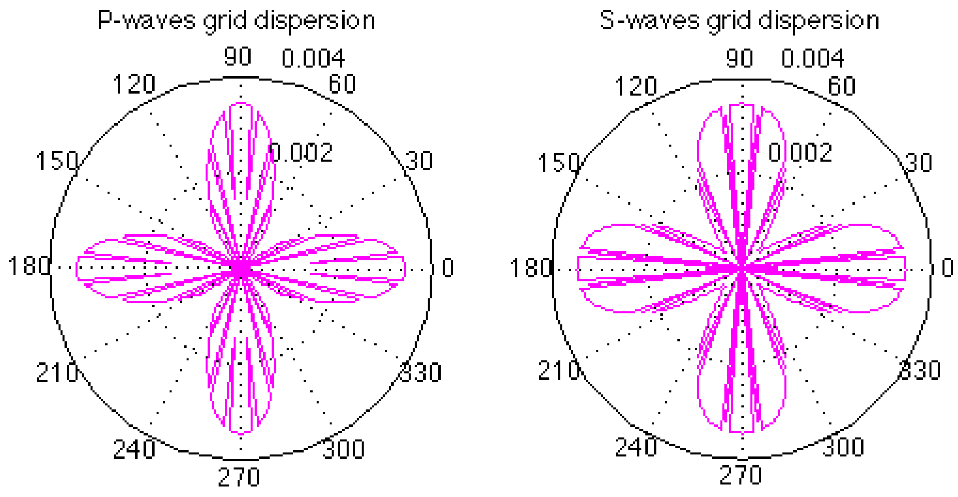


Figure 1.10: Dispersion anisotropy for Q_5 elements with 5 points per wavelength.

On Figure 1.11 we compare the dispersion error for Q_3 elements with 10 points per wavelength with standard and optimized penalty. As we can see the results are almost the same for the P-waves dispersion and slightly better for the optimized penalty for the S-waves dispersion.

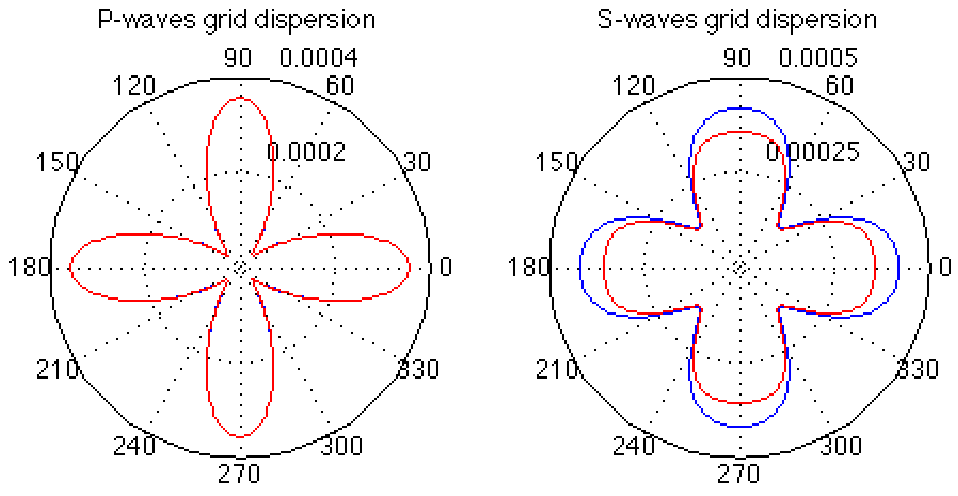


Figure 1.11: Comparison of the dispersion error with standard (in blue) and optimized penalty (in red) for Q_3 elements with 10 points per wavelength.

1.5.3 Stability condition formulation

We can use the previous analysis to derive the stability condition associated with our fully-discrete scheme. For that, we have to introduce the time discretization in the definition of the numerical plane wave. We thus have to inject the plane waves into the fully discretized DG approximation. By doing so, we get a relation between Δt and h.

The fully discrete local DG approximation is

$$M\frac{\mathbf{u}_{n+1}^K - 2\mathbf{u}_n^K + \mathbf{u}_{n-1}^K}{\Delta t^2} + K\mathbf{u}_n^K + \sum_{f\in\mathcal{F}_K} F^f \mathbf{u}_n^{K_f} = 0.$$

Since the displacement is a plane wave we have

$$u_{n,j}^K = A_j e^{-i(\mathbf{k}\cdot\mathbf{x}_j^K - \omega_h n\Delta t)}.$$

Injecting this relation in the DG approximation yields

$$M\frac{e^{-i\omega_h\Delta t} - 2 + e^{i\omega_h\Delta t}}{\Delta t^2}\mathbf{u}_n^K + \left(K + \sum_{f\in\mathcal{F}_K} e^{i\mathbf{k}\cdot\mathbf{x}_f} F^f\right)\mathbf{u}_n^K = 0.$$

We reformulate the temporal term with some trigonometric relations

$$\frac{e^{-i\omega_h\Delta t} - 2 + e^{i\omega_h\Delta t}}{\Delta t^2} = \frac{2(\cos(\omega_h\Delta t) - 1)}{\Delta t^2} = -\frac{4\sin^2(\frac{\omega_h\Delta t}{2})}{\Delta t^2}.$$

Hence, we get the following generalized eigenvalue problem

$$-\frac{4h^2\sin^2(\frac{\omega_h\Delta t}{2})}{\Delta t^2}\hat{M}\mathbf{u}_n^K + \left(\hat{K} + \sum_{f\in\mathcal{F}_K} e^{i\mathbf{k}\cdot\mathbf{x}_f}\hat{F}^f\right)\mathbf{u}_n^K = 0, \tag{1.35}$$

where $\hat{M} = \frac{1}{h^d}M$, $\hat{K} = \frac{1}{h^{2-d}}K$ and $\hat{F}^f = \frac{1}{h^{2-d}}F^f$. We note that $\lambda = \dfrac{4h^2\sin^2(\frac{\omega_h\Delta t}{2})}{\Delta t^2}$ is an eigenvalue of our generalized eigenvalue problem. In order to have stability $\frac{4h^2\sin^2(\frac{\omega_h\Delta t}{2})}{\Delta t^2}$ has to be below all eigenvalues, yielding the following stability relation

$$\frac{\Delta t}{h} \leq \min_{1\leq j\leq N_K}\min_{0\leq\theta\leq 2\pi}\frac{2}{\sqrt{\Lambda_j(\theta)}}, \tag{1.36}$$

where $\{\Lambda_j\}_{1\leq j\leq N_K}$ are the eigenvalues of (1.35) according to the angle of incidence θ. We recall that N_K is the number of degrees of freedom per element.

Remark 1.9. *This study is done on an infinite medium and therefore do not take into account the impact on the CFL condition of Dirichlet or Neumann boundary conditions. It is noteworthy that boundary conditions weaken slightly the CFL condition as we will see with the energy analysis in Section 1.6.*

1.5.4 CFL conditions

The CFL stability condition is a relation of the form

$$v_p\frac{\Delta t}{h} < C_{cfl}(k), \tag{1.37}$$

where $C_{cfl}(k)$ is the CFL constant depending on the polynomial order k of the polynomial spaces Q_k.

From the relation (1.36) we immediatly get the value of the CFL constant:

$$C_{cfl}(k) = \frac{1}{v_p}\min_{1\leq j\leq N_K}\min_{0\leq\theta\leq 2\pi}\frac{2}{\sqrt{\Lambda_j(\theta)}}.$$

1.5.4.1 Comparing optimized and standard penalties

In this section, we compare the impact on the CFL condition of the optimized penalty introduced in Section 1.3.4 with the standard penalty. We note on Table 1.1 that the optimized penalty grants a gain for any polynomial degree of 33% in the CFL condition. These CFL constants have been calculated with the same velocities and penalty as the dispersion.

Space	Standard	Optimized	gain
Q_1	0.150	0.199	33%
Q_2	0.0953	0.121	27%
Q_3	0.0420	0.0561	34%
Q_4	0.0319	0.0417	31%
Q_5	0.0194	0.0259	34%
Q_6	0.0158	0.0207	31%
Q_7	0.0111	0.0148	33%
Q_8	0.00941	0.0123	31%
Q_9	0.00724	0.00962	33%
Q_{10}	0.00622	0.00821	32%

Table 1.1: CFL conditions for different polynomial spaces Q_k for Gauss-Legendre basis functions with optimized and standard penalties.

1.5.4.2 Dependency of the CFL condition with the penalty parameter

In this section we want to show the impact of the penalty parameter on the CFL condition; this result is known for the acoustic equation [?]. We remark on Figure 1.12 that the dependency of the CFL condition is $\mathcal{O}(\alpha^{-\frac{1}{2}})$. It is therefore quite interesting to get the optimal penalty parameter. However, one has to be really careful when trying to find the optimal value, contrary to a CFL condition unstability which is quick and explosive, we observed that too low penalty can take a long time before the unstability is revealed, especially for smooth solutions.

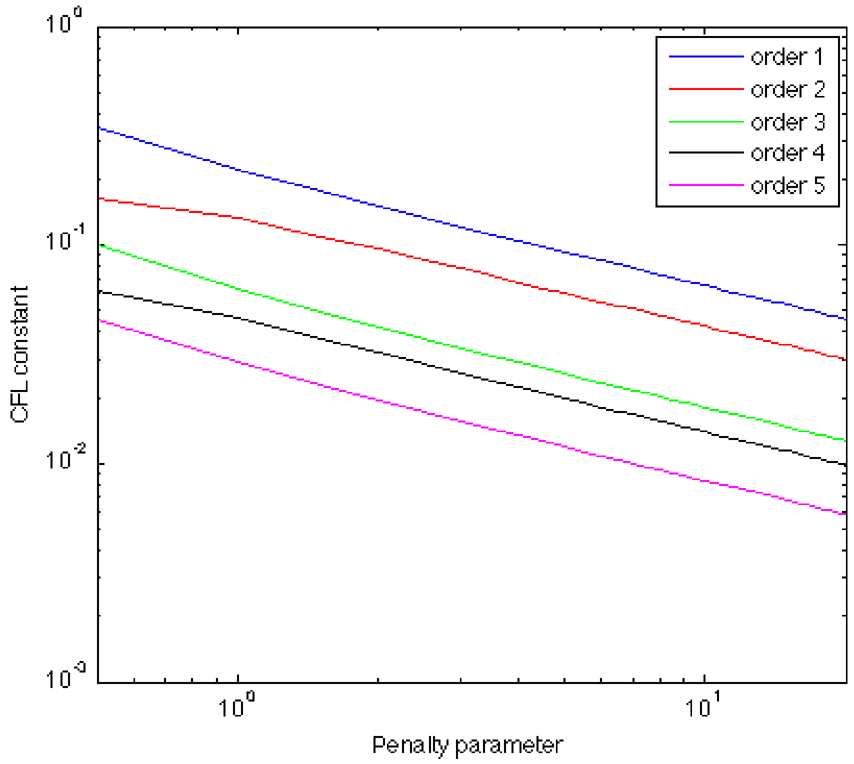


Figure 1.12: Evolution of the CFL condition according to the penalty parameter α and of the polynomial order.

1.5.5 Considerations on the computational and memory costs of DG methods

In this section, we propose to illustrate the effect on the computational and memory costs of different polynomial order basis functions based on the previous dispersion error and stability analysis. Indeed, different polynomial orders result in different computational and memory costs for the same accuracy. The computational cost C_{comp} and memory cost C_{mem} can be considered as unitary since they do not depend of the size of the domain, we propose to evaluate these costs by the following formulas

$$C_{comp}(k) = \frac{nb^3_{elts}(k)}{C_{cfl}(k)} nb^2_{dof}(k),$$

and

$$C_{mem}(k) = nb^2_{elts}(k) nb_{dof}(k),$$

where nb_{dof} is the number of degrees of freedom for one element, C_{cfl} the CFL constant , nb_{pts} the number of points per wavelength and $nb_{elts}(k) = \dfrac{nb_{pts}(k)}{k+1}$ the number of elements per wavelength.

We chose these formulas since the computation cost is proportional to the inverse of the time step ($\frac{1}{\Delta t} \propto \frac{nb_{elts}}{C_{cfl}}$) which is proportional to the number of iterations needed per unit of time, multiplied by the number of points per wavelength power the dimension which reflects the number of elements needed per unit of space ($\frac{1}{h} \propto nb_{elts}$), multiplied by the size of the elementary matrices $nb_{dof} \times nb_{dof}$. The memory cost is proportional to the number of degrees of freedom per element multiplied by the number of elements per wavelength power the dimension. We have

$$C_{comp}(k) \propto \frac{1}{\Delta t}\frac{1}{h^2} nb_{dof}^2,$$

and

$$C_{mem}(k) \propto \frac{nb_{dof}}{h^2}.$$

We report in Table 1.2 for different polynomial spaces Q_k the different constants to calculate the computational and memory costs to achieve a dispersion error of 10^{-2} and of 10^{-4}.

			Dispersion error $= 10^{-2}$		Dispersion error $= 10^{-4}$	
Space	nb_{dof}	C_{cfl}	nb_{pts}	nb_{elts}	nb_{pts}	nb_{elts}
Q_1	8	0.199	30	15	250	125
Q_2	18	0.121	9	3	28	9.3
Q_3	32	0.0561	6	1.5	15	3.75
Q_4	50	0.0417	5	1	10	2
Q_5	72	0.0259	5	0.85	8	1.3

Table 1.2: Constants to calculate computation and memory costs.

We report in Table 1.3 the different costs for polynomial orders going from 1 to 5. As we can note, the memory cost decrease substantially with the order, Q_1 and Q_2 being way behind. The optimal computational cost for a dispersion error of 10^{-2} is obtained with Q_4 elements and for a dispersion error of 10^{-4} the optimum is obtained with Q_5 elements. We also note that Q_1 elements cost a lot more than other elements. There is a factor 18 in computational cost between Q_1 and Q_4 elements for a dispersion error of 10^{-2}, and a factor 1400 between Q_1 and Q_5 elements for a dispersion error of 10^{-4}. Regarding the memory cost there is a factor 34 between Q_1 and Q_4 elements for a dispersion error of 10^{-2}, and a factor 1000 between Q_1 and Q_5 elements for a dispersion error of 10^{-4}.

Therefore, both on computational and memory costs Q_4 elements is the best choice to achieve a dispersion error of 10^{-2} and Q_5 for a dispersion error of 10^{-4}.

	Dispersion error $= 10^{-2}$		Dispersion error $= 10^{-4}$	
Space	Computational cost	Memory cost	Computational cost	Memory cost
Q_1	1085400	1800	628140000	125000
Q_2	72298	162	2153800	1557
Q_3	61604	72	962570	450
Q_4	59952	50	479620	200
Q_5	122920	52	439740	122

Table 1.3: Comparison of the computational and memory costs for different polynomial order basis for the same dispersion error.

The lesson from this is that we should not think that high order means high computational and memory costs, quite the contrary. However, this remark stands only for smooth solution and smooth medium. But where there are singularities and strong local heterogeneities we should use space-time local mesh refinement.

1.6 Stability results for non-conforming heterogeneous media

In this section we use an energetic approach to establish a general CFL stability condition, *i.e.* isotropic hp non-conforming heterogeneous cases, for the SIPDG method. Since this study makes a great use of upper bounds, it is less accurate than the previous one. However, its locality gives precious information about the dependencies of the stability in heterogeneous and hp non-conforming cases.

We remind that for an explicit scheme of the form

$$m_\rho\Big(\frac{\mathbf{u}_h^{n+1} - 2\mathbf{u}_h^n + \mathbf{u}_h^{n-1}}{\Delta t^2}\,,\,\mathbf{v}_h\Big) + a_h(\mathbf{u}_h^n\,,\,\mathbf{v}_h) = l(\mathbf{v}_h),\ \ \mathbf{v}_h \in V_h,$$

with a_h a symmetric positive definite bilinear form, we have the conservation of the discrete energy

$$\mathcal{E}_h^{n+1/2} := m_\rho(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t}\,,\,\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t}) + a_h(\mathbf{u}_h^{n+1}\,,\,\mathbf{u}_h^n).$$

Using the identity of the parallelogram on a_h, the study of stability boils down to finding a CFL condition on Δt to ensure the positivity of the form on $V_h \times V_h$:

$$b_h(\mathbf{v}_h\,,\,\mathbf{v}_h) := m_\rho(\mathbf{v}_h\,,\,\mathbf{v}_h) - \frac{\Delta t^2}{4} a_h(\mathbf{v}_h\,,\,\mathbf{v}_h)$$

First, we shall give some inverse estimation results:

Lemma 1.2
$\forall \mathbf{v}_h \in V_h$, $\forall K \in \mathcal{T}_h$

$$\begin{aligned}
&\|\mathrm{div}\,(\mathbf{v}_h|_K)\|_{L^2(K)} \leq \frac{C_{\mathrm{div}}(p^K)}{|K|^{1/2}}\|\mathbf{v}_h|_K\|_{L^2(K)},\\
&\|\partial_i \mathbf{v}_{h,i}|_K\|_{L^2(K)} \leq \frac{C_\partial(p^K)}{|K|^{1/2}}\|\mathbf{v}_h|_K\|_{L^2(K)},\\
&\|\partial_1 \mathbf{v}_{h,2}|_K + \partial_2 \mathbf{v}_{h,1}|_K\|_{L^2(K)} \leq \frac{C_{12}(p^K)}{|K|^{1/2}}\|\mathbf{v}_h|_K\|_{L^2(K)},
\end{aligned} \tag{1.38}$$

where

$$C_{\mathrm{div}}(p^K)^2 := \lambda_{\max}\Big(\hat{\mathrm{M}}^{-1/2}\hat{\mathrm{R}}_{div}\hat{\mathrm{M}}^{-1/2}\Big),$$

$$C_\partial(p^K)^2 := \lambda_{\max}\Big(\hat{\mathrm{M}}^{-1/2}\hat{\mathrm{R}}_\partial\hat{\mathrm{M}}^{-1/2}\Big)$$

and

$$C_{12}(p^K)^2 := \lambda_{\max}\Big(\hat{\mathrm{M}}^{-1/2}\hat{\mathrm{R}}_{12}\hat{\mathrm{M}}^{-1/2}\Big)$$

with

$$\hat{\mathrm{R}}_{div} := \left(\int_{\hat{K}} \hat{\mathrm{div}}(\hat{\varphi}_l)\cdot\hat{\mathrm{div}}(\hat{\varphi}_m)\,d\hat{\mathbf{x}}\right)_{l,\,,m=1,\cdots,2(p^K+1)^2},$$

$$\hat{\mathrm{R}}_{\partial} := \left(\int_{\hat{K}} \hat{\partial}_1(\hat{\varphi}_{l,1}) \cdot \hat{\partial}_1(\hat{\varphi}_{m,1})\, d\hat{\mathbf{x}} \right)_{l,,m=1,\cdots,2(p^K+1)^2},$$

$$\hat{\mathrm{R}}_{12} := \left(\int_{\hat{K}} (\hat{\partial}_1(\hat{\varphi}_{l,2}) + \hat{\partial}_2(\hat{\varphi}_{l,1})) \cdot (\hat{\partial}_1(\hat{\varphi}_{m,2}) + \hat{\partial}_2(\hat{\varphi}_{m,1}))\, d\hat{\mathbf{x}} \right)_{l,,m=1,\cdots,2(p^K+1)^2},$$

and

$$\hat{\mathrm{M}} := \left(\int_{\hat{K}} \hat{\varphi}_l \cdot \hat{\varphi}_m\, d\hat{\mathbf{x}} \right)_{l,,m=1,\cdots,2(p^K+1)^2}.$$

Proof. Straightforward since the space is of finite dimension. □

We following theorem state sufficient local stability conditions obtained through the energy analysis:

Theorem 1.5
If Δt verifies the local CFL conditions:
$\forall K \in \mathcal{T}_h$,

$$\frac{\Delta t}{|K|^{1/2}} \leq \frac{2}{\sqrt{C_K}} \tag{1.39}$$

with

$$\begin{aligned} C_K := & 3\frac{\lambda_K}{\rho_K} C_{\mathrm{div}}(p^K)^2 + 4\frac{(2\mu_K)}{\rho_K} C_{\partial}^2(p^K) + 3\frac{\mu_K}{\rho_K} C_{12}(p^K)^2 \\ & + \sum_{\Gamma \in \mathcal{F}_h(K)} (\delta + 2) \max(1, \frac{\rho_{V_\Gamma(K)}}{\rho_K}) \{\!\{ C_{inv}(p)^2 [v_p^2 + v_s^2] \}\!\}_\Gamma \, C_{inv}(p^K)^2 \frac{|K|}{h_\Gamma^2} \end{aligned} \tag{1.40}$$

then the explicit scheme (1.32) is L^2-stable.

Even though the CFL estimation stated in the following theorem is more pessimistic than the one obtained with the plane wave analysis, the fact that it takes into account heterogeneities, boundary conditions and *hp* non-conformities gives us valuable information. This theorem provides all these information because of the locality of the CFL condition stated in the theorem. Indeed, instead of having a global CFL condition as in the plane wave analysis, the following theorem state a local CFL condition for each element.

The first remark we can make concerns how we defined the CFL constant $C_{cfl}(k)$ in (1.37). The dependency of the CFL condition is not linear with v_p, and looks to be more of the form $\sqrt{v_p^2 + v_s^2}$. However, since the ratio $\dfrac{v_p}{v_s}$ usually lies in the interval $[\sqrt{2}, 2]$, and we calculated our CFL constants C_{cfl} in the worst case where $\dfrac{v_p}{v_s} = 2$, these constants are still legitimate but relatively pessimistic.

We see that the dependency of the CFL condition with the penalty in $\mathcal{O}(\alpha^{-1/2})$ observed in Section 1.5.4.2 is confirmed by the following theorem since C_K has a linear dependency with the penalty constant δ. We remind that this result was already observed by Agut and Diaz in [**?**] for the acoustic equation.

Concerning heterogeneities, in most cases the global CFL condition is the one dictated by the most restrictive medium. However, in cases of high contrast, we see that the local CFL conditions might deteriorate the global CFL condition, *e.g.* same velocities in two neighboring elements (we might expect the same local CFL conditions) but different densities ρ, then the term

$$\max(1, \frac{\rho_{V_\Gamma(K)}}{\rho_K})\{\!\{C_{inv}(p)^2[v_p^2 + v_s^2]\}\!\}_\Gamma = \frac{\rho_{max}}{\rho_{min}}(v_p^2 + v_s^2)$$

is obviously greater than $(v_p^2 + v_s^2)$.

In the case of h-adaptivity we see that the CFL condition deteriorates since $\frac{|K|}{h_\Gamma^2} \geq 1$ for non conforming faces. In our Cartesian case, $\frac{|K|}{h_\Gamma^2} = \frac{h}{(\frac{h}{p_s})^2} = p_s^2$, where p_s is the spatial refinement ratio. Thus, the CFL condition deteriorates linearly with the space refinement p_s, this is not a real problem since the refined elements impose the same kind of restriction on the CFL condition. However, in the case of a local time stepping scheme, the coarse element right next to the non-conformity should be included in the local time stepping scheme since its local CFL condition is of the same kind as the small elements.

In the case of p-adaptivity wee see that the CFL condition deteriorates for the elements next to the non-conformity with the lower degree. Indeed, the dependency in polynomial degree is $\{\!\{C_{inv}(p)^2\}\!\}_\Gamma\, C_{inv}(p^K)^2 = \frac{C_{inv}(p_{min})^2 + C_{inv}(p_{max})^2}{2} p_{min}^2 \geq p_{min}^4$. Since, in the Cartesian case $C_{inv}(p)^2 = (p+1)^2$ the CFL condition can be substantially weaken on the element with $Q_{p_{min}}$ right next to the element with $Q_{p_{max}}$. This is relatively troublesome since that means that with local time stepping we most likely will not be able to take the CFL constant $C_{cfl}(p_{min})$ in the local time stepping area.

Proof. To show the stability result, we begin with the estimations used in the proof of the

Theorem 1.4:

$$\begin{aligned}
a_h(\mathbf{v}_h,\mathbf{v}_h) =& \int_\Omega \sigma_h(\mathbf{v}_h) : \nabla_h \mathbf{v}_h \, d\mathbf{x} - 2\int_{\mathcal{F}_h} \{\!\{\sigma_h(\mathbf{v}_h)\mathbf{n}\}\!\} \cdot [\![\mathbf{v}_h]\!] \, d\gamma \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\cup\mathcal{F}_h} \delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}\cup\mathcal{F}_h} \delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)} \\
\leq& \sum_{K\in\mathcal{T}_h} \Big((1+4\xi^2)\|\lambda_K^{1/2}\mathrm{div}\mathbf{v}_K\|^2_{L^2(K)} \\
&+ (1+2\xi^2)\|(2\mu_K)^{1/2}\partial_1\mathbf{v}_{K,1}\|^2_{L^2(K)} \\
&+ (1+2\xi^2)\|(2\mu_K)^{1/2}\partial_2\mathbf{v}_{K,2}\|^2_{L^2(K)} \\
&+ (1+4\xi^2)\|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1}+\partial_1\mathbf{v}_{K,2})\|^2_{L^2(K)} \Big) \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}} (1+\frac{1}{2\xi^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}} (1+\frac{1}{2\xi^2\delta_T})\delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)} \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{b}} (1+\frac{1}{\xi^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{b}} (1+\frac{1}{\xi^2\delta_T})\delta_T \frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_T\|^2_{L^2(\Gamma)}.
\end{aligned} \tag{1.41}$$

Using the inverse estimations of Lemma 1.2, we get

$$\begin{aligned}
&\sum_{K\in\mathcal{T}_h} \Big((1+4\xi^2)\|\lambda_K^{1/2}\mathrm{div}\mathbf{v}_K\|^2_{L^2(K)} + (1+2\xi^2)\|(2\mu_K)^{1/2}\partial_1\mathbf{v}_{K,1}\|^2_{L^2(K)} \\
&+ (1+2\xi^2)\|(2\mu_K)^{1/2}\partial_2\mathbf{v}_{K,2}\|^2_{L^2(K)} + (1+4\xi^2)\|\mu_K^{1/2}(\partial_2\mathbf{v}_{K,1}+\partial_1\mathbf{v}_{K,2})\|^2_{L^2(K)} \Big) \\
\leq& \sum_{K\in\mathcal{T}_h} \Big((1+4\xi^2)\frac{\lambda_K}{\rho_K} C_{\mathrm{div}}(p^K)^2 + 2(1+2\xi^2)\frac{(2\mu_K)}{\rho_K} C^2_\partial(p^K) \\
&+ (1+4\xi^2)\frac{\mu_K}{\rho_K} C_{12}(p^K)^2 \Big) \rho_K \frac{\|\mathbf{v}_K\|^2_{L^2(K)}}{|K|}.
\end{aligned} \tag{1.42}$$

Moreover, using a triangular inequality and an inverse estimation, we get

$$\begin{aligned}
&\sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}} (1+\frac{1}{2\xi^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{b}} (1+\frac{1}{\xi^2\delta_N})\delta_N \frac{\{\!\{C_{inv}(p)^2(\lambda+2\mu)\}\!\}}{h_\Gamma} \|[\![\mathbf{v}_h]\!]_N\|^2_{L^2(\Gamma)} \\
\leq& \sum_{K\in\mathcal{T}_h} \Big(\sum_{\Gamma\in\mathcal{F}_h(K)} a(1+\frac{1}{\xi^2\delta_N})\delta_N \{\!\{C_{inv}(p)^2\frac{(\lambda+2\mu)}{\rho_K}\}\!\}_\Gamma C_{inv}(p^K)^2 \frac{|K|}{h_\Gamma^2} \Big) \rho_K \frac{\|\mathbf{v}_K\|^2_{L^2(K)}}{|K|},
\end{aligned} \tag{1.43}$$

and

$$\begin{aligned}
&\sum_{\Gamma\in\mathcal{F}_h^{\mathcal{I}}}(1+\frac{1}{2\xi^2\delta_N})\delta_T\frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)}\\
&+\sum_{\Gamma\in\mathcal{F}_h^{b}}(1+\frac{1}{\xi^2\delta_N})\delta_T\frac{\{\!\{C_{inv}(p)^2\mu\}\!\}}{h_\Gamma}\|[\![\mathbf{v}_h]\!]_N\ \|^2_{L^2(\Gamma)} \qquad (1.44)\\
&\le\sum_{K\in\mathcal{T}_h}\Big(\sum_{\Gamma\in\mathcal{F}_h(K)}(1+\frac{1}{\xi^2\delta_N})\delta_T\{\!\{C_{inv}(p)^2\frac{\mu}{\rho_K}\}\!\}_\Gamma\,C_{inv}(p^K)^2\frac{|K|}{h_\Gamma^2}\Big)\rho_K\frac{\|\mathbf{v}_K\|^2_{L^2(K)}}{|K|}.
\end{aligned}$$

Using (1.42), (1.43) and (1.44), (1.41) becomes:

$$\begin{aligned}
a_h(\mathbf{v}_h,\mathbf{v}_h)\le\sum_{K\in\mathcal{T}_h}&\Big((1+4\xi^2)\frac{\lambda_K}{\rho_K}C_{\mathrm{div}}(p^K)^2+2(1+2\xi^2)\frac{(2\mu_K)}{\rho_K}C^2_\partial(p_K)\\
&+(1+4\xi^2)\frac{\mu_K}{\rho_K}C_{12}(p^K)^2\\
&+\sum_{\Gamma\in\mathcal{F}_h(K)}(1+\frac{1}{\xi^2\delta})\delta\{\!\{C_{inv}(p)^2[\frac{\mu}{\rho_K}+\frac{(\lambda+2\mu)}{\rho_K}]\}\!\}_\Gamma\,C_{inv}(p^K)^2\frac{|K|}{h_\Gamma^2}\Big)\rho_K\frac{\|\mathbf{v}_K\|^2_{L^2(K)}}{|K|}.
\end{aligned}\qquad (1.45)$$

Let $\xi=1/\sqrt{2}$.

$$\begin{aligned}
a_h(\mathbf{v}_h,\mathbf{v}_h)\le\sum_{K\in\mathcal{T}_h}&\Big(3\frac{\lambda_K}{\rho_K}C_{\mathrm{div}}(p^K)^2+4\frac{(2\mu_K)}{\rho_K}C^2_\partial(p^K)+3\frac{\mu_K}{\rho_K}C_{12}(p^K)^2\\
&+\sum_{\Gamma\in\mathcal{F}_h(K)}(\delta+2)\delta\{\!\{C_{inv}(p)^2[\frac{\mu}{\rho_K}+\frac{(\lambda+2\mu)}{\rho_K}]\}\!\}_\Gamma\,C_{inv}(p^K)^2\frac{|K|}{h_\Gamma^2}\Big)\rho_K\frac{\|\mathbf{v}_K\|^2_{L^2(K)}}{|K|}.
\end{aligned}\qquad (1.46)$$

Finally, we get the following lower bound:

$$\begin{aligned}
b_h(\mathbf{v}_h\,,\,\mathbf{v}_h)\ge\sum_{K\in\mathcal{T}_h}&\Big[1-\frac{\Delta t^2}{4|K|}\Big(3\frac{\lambda_K}{\rho_K}C_{\mathrm{div}}(p^K)^2+4\frac{(2\mu_K)}{\rho_K}C^2_\partial(p^K)+3\frac{\mu_K}{\rho_K}C_{12}(p^K)^2\\
&+\sum_{\Gamma\in\mathcal{F}_h(K)}(\delta+2)\{\!\{C_{inv}(p)^2[\frac{\mu}{\rho_K}+\frac{(\lambda+2\mu)}{\rho_K}]\}\!\}_\Gamma\,C_{inv}(p^K)^2\frac{|K|}{h_\Gamma^2}\Big)\Big]\rho_K\|\mathbf{v}_K\|^2_{L^2(K)}.
\end{aligned}\qquad (1.47)$$

where

$$b_h(\mathbf{v}_h\,,\,\mathbf{v}_h)=m_\rho(\mathbf{v}_h\,,\,\mathbf{v}_h)-\frac{\Delta t^2}{4}a_h(\mathbf{v}_h\,,\,\mathbf{v}_h). \qquad (1.48)$$

□

1.7 Conclusion

We introduced the standard IPDG methods for the second order elastodynamic equation. We proposed a penalty term more suited for this equation than the standard penalty term. This penalty term grants a gain of roughly 30% in the CFL condition and a slightly improved dispersion. Our comparative study showed that SIPDG is the most suited IPDG method for elastodynamic, the main two reasons are, the convergence rate of the error which is optimal, and the convergence of the dispersion which is two times larger for SIPDG than for other IPDG methods. The dispersion error is particularly important in an oil exploration context, since an error on the velocity result in an error in the imaging process. Moreover, the symmetry of the SIPDG method offers many accurate possibilities to study the scheme which are not possible with other IPDG schemes. In particular, we can have a CFL condition in heterogeneous medium, with boundary conditions, non-conforming meshes and with *hp*-adaptivity.

Chapter 2

Perfectly matched layers (PML) for the second order elastodynamic equation

2.1 Introduction

Many problems in the simulation of elastic wave propagation have a medium which is either unbounded or much larger than the area of interest. For reasons of problem tractability we have to bound the medium in these cases. This raises the question to know how to artificially bound our medium to simulate an infinite medium. This is a longstanding problem, many researches have been developed in the past and this still is an active area. There mainly exists two classes of methods to achieve this: *absorbing conditions* [**?** **?** **?** **?** **?** **?**] and *absorbing layers* [**?** **?** **?** **?** **?** **?** **?** **?**]. The main objective of these methods is to have boundaries as transparent as possible, as if the medium was unbounded. Absorbing conditions are also referred to as *non-reflecting boundary conditions*; as their name suggests they are conditions on the boundaries of the medium. The main absorbing layers suitable for unbounded medium simulation are referred to as ***P**erfectly **M**atched **L**ayers (PML)*, which are additional non-physical media surrounding the area of interest. The essential property of a PML which distinguishes it from an ordinary absorbent material is that it is designed in such a way that the outgoing waves from the area of interest reaching the PML are not reflected at the interface. This property allows PML to strongly absorb all the outgoing waves of a computational domain without changing the propagation in this area. Propagation problems require more and more precise methods to get accurate simulations, and thus the absorbing methods must be as perfect as possible; for this reason PML have become more and more popular during the last decade, and are the method we have chosen.

PML were first introduced by J.P. Bérenger [**?**] for Maxwell's equations, this formulation referred to as split PML formulation was proved only weakly well-posed and unstable [**?**]. Later, an unsplit formulation was proposed by L. Zhao and C. Cangellaris in [**?**] and proven strongly well-posed and stable. Most recent formulations of the PML are based on the unsplit formulation. Although PML were first proposed for Maxwell's equations, they have been extended to many wave propagation problems, especially the acoustic and elastodynamic equations [**?** **?** **?**].

PMLs have been developed initially for first order systems and then successfully developed for second order equations [**? ? ? ? ?**]. Recently, in the elastodynamic community, most of the effort has been put on developing PML for more complex formulation of the elastodynamic problem, anisotropic or poroelastic media for instance, that leads to instabilities in the PML.

As a general problem, defining the best PML formulation and discretization in the time domain is still an open question. Although, in our isotropic second order case, we can say that a formulation that does not impact the CFL condition, keeps the second order form, and introduces as few new unknowns as possible would be the best. Many PML formulations of the second order elastodynamic equation reformulate it as a first order system and thereby introduce many additional unknowns. Recent studies [**?**] show that the temporal discretization as a major impact on the stability of the PML, therefore we emphasize our choices that do not impact the CFL condition of our method.

In this chapter, we focus on the application of a second order PML formulation to our interior penalty discontinuous Galerkin approximation of the second order elastodynamic equation. In the Section 2.2, we introduce the main ideas of PML and the PML formulation for the second order elastodynamic equation through the PML coordinate transformation, after writing the equation in the frequency domain. In the Section 2.3, we first introduce the discontinuous Galerkin approximation of the PML formulation we have chosen; then we detail and argue the discretization techniques we have selected. In the Section 2.4, we conclude with numerical results to show the good behavior of the method.

2.2 Perfectly Matched Layers Model

2.2.1 General ideas

First, we need to identify the bounded area of interest Ω_ϕ and the rest of the space is called Ω_{PML} where the absorption takes place. Using PML around an area hides any physical phenomenon that would have taken place in Ω_{PML}. Therefore, the area of interest must be chosen carefully, according to surrounding heterogeneities, especially for comparison with real data.

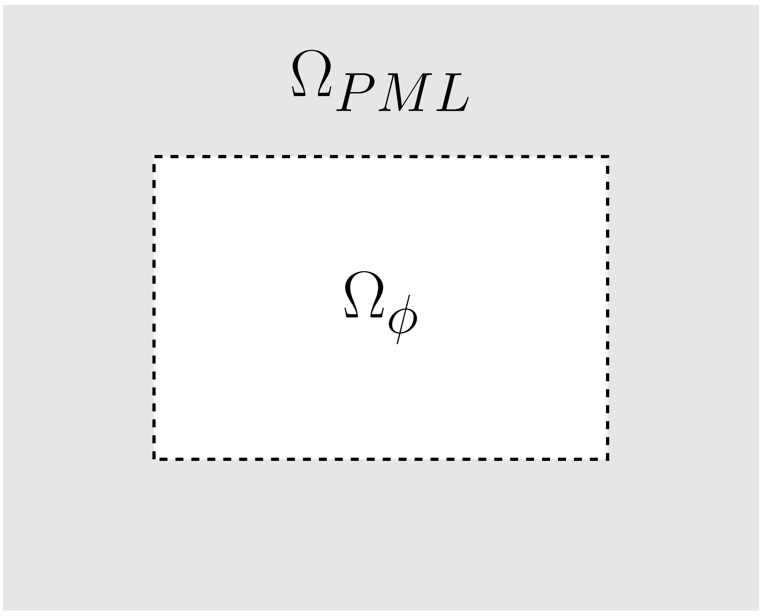


Figure 2.1: $\Omega = \Omega_\phi \cup \Omega_{PML}$.

The PML method can be interpreted as a complex coordinate transformation in the frequency domain:

$$\forall j = 1, 2, \quad x_j \mapsto \tilde{x}_j = x_j + \frac{1}{i\omega}\int_0^{x_j} \zeta_j(\xi) d\xi,$$

where i is the imaginary unit, ω is the pulsation, ζ_j are functions, positive on Ω_{PML} and null on Ω_ϕ, called dumping functions. More technically, it actually is an analytic continuation of the elastodynamic equation in a complex manifold.

To understand why this coordinate transformation creates an absorbing layer, it is interesting to consider plane waves solutions to the elastodynamic equation on an unbounded homogeneous domain:

$$\mathbf{u}(t, \mathbf{x}) = \mathbf{u_0} e^{i(\mathbf{k}\cdot\mathbf{x} - \omega t)},$$

where the pulsation ω and the wave vector $\mathbf{k}$ verify the dispersion relation (B.7). Then if we introduce: $\mathbf{v}(t, \mathbf{x}) = \mathbf{u}(t, \tilde{\mathbf{x}})$, we have:

$$\mathbf{v}(t, \mathbf{x}) = \mathbf{u_0} e^{i(\mathbf{k}\cdot\mathbf{x} - \omega t)} e^{-\sum_{j=1}^{d} k_j \cdot \int_0^{x_j} \zeta_j(s)\, ds}.$$

Thus, we have $\forall t \in [0, T]$ $\forall \mathbf{x} \in \Omega_\phi$, $\mathbf{v}(t, \mathbf{x}) = \mathbf{u}(t, \mathbf{x})$ and therefore no reflections, and $\mathbf{v}$ decreases exponentially in Ω_{PML} which characterizes the absorption.
In order to get the PML formulation, we will proceed as follows:

1. As the PML coordinate transformation is defined in the frequency domain, we first apply a Fourier-Laplace transform to the elastodynamic equation to obtain an equation in the frequency domain.
2. We can then apply the PML coordinate transformation. As we will show, it introduces some difficulties that will be overcome by performing a few algebraic manipulations and introducing new unknowns.
3. Apply an inverse Fourier-Laplace transform to get a system of equations in the time domain.

2.2.2 PML formulation

In this section we introduce the continuous PML formulation we used.

We consider the linear elastodynamic propagation problem in an unbounded domain. We assume for the sake of simplicity of exposure and without loss of generality, no sources and no initial conditions. We further assume the propagation velocities v_s and v_p to be constant in the direction of absorption in Ω_{PML}. Hence, in Ω the displacement $\mathbf{u}$ satisfies

$$\rho \frac{\partial^2}{\partial t^2}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 u_1}{\partial x^2} + \mu\frac{\partial^2 u_1}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 u_2}{\partial x \partial y} \\ \mu\frac{\partial^2 u_2}{\partial x^2} + (\lambda+2\mu)\frac{\partial^2 u_2}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 u_1}{\partial x \partial y} \end{pmatrix}. \tag{2.1}$$

Step 1: Fourier-Laplace transform in the time domain.
Applying the Fourier-Laplace transform in time to Equation (2.1) yields the following equation in the frequency domain

$$\rho s^2 \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} = \begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 \hat{u}_1}{\partial x^2} + \mu\frac{\partial^2 \hat{u}_1}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_2}{\partial x \partial y} \\ \mu\frac{\partial^2 \hat{u}_2}{\partial x^2} + (\lambda+2\mu)\frac{\partial^2 \hat{u}_2}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_1}{\partial x \partial y} \end{pmatrix}, \tag{2.2}$$

where $s = i\omega$, and $\hat{\mathbf{u}}$ is the Fourier-Laplace transform of $\mathbf{u}$.

Step 2: PML Equation (2.3) seen as a perturbation of the initial problem (2.2).
We extend Equation (2.2) in the PML coordinate system

$$\rho s^2 \begin{pmatrix} \hat{v}_1 \\ \hat{v}_2 \end{pmatrix} = \begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 \hat{v}_1}{\partial \tilde{x}^2} + \mu\frac{\partial^2 \hat{v}_1}{\partial \tilde{y}^2} + (\lambda+\mu)\frac{\partial^2 \hat{v}_2}{\partial \tilde{x} \partial \tilde{y}} \\ \mu\frac{\partial^2 \hat{v}_2}{\partial \tilde{x}^2} + (\lambda+2\mu)\frac{\partial^2 \hat{v}_2}{\partial \tilde{y}^2} + (\lambda+\mu)\frac{\partial^2 \hat{v}_1}{\partial \tilde{x} \partial \tilde{y}} \end{pmatrix}, \tag{2.3}$$

where $\hat{\mathbf{v}}$ is the solution of this new equation, and one can prove

$$\forall t \in [0,T],\ \forall x \in \Omega_\phi,\ \hat{\mathbf{v}}(t,x) = \hat{\mathbf{u}}(t,x)$$

and

$$\forall t \in [0,T],\ \forall x \in \Omega_{PML},\ \hat{\mathbf{v}}(t,x) \neq \hat{\mathbf{u}}(t,x).$$

By abusing the notation, we will write $\hat{\mathbf{u}}$ instead of $\hat{\mathbf{v}}$ thereafter.
Then, we interpret $\frac{\partial}{\partial \tilde{x}_i}$ based on $\frac{\partial}{\partial x_i}$, we have

$$\forall i = 1,2, \quad \frac{\partial}{\partial \tilde{x}_i} = \frac{s}{s+\zeta_i}\frac{\partial}{\partial x_i} = \frac{1}{\nu_i}\frac{\partial}{\partial x_i}. \tag{2.4}$$

Applying the relation defined by Equation (2.4) to Equation (2.3) yields

$$\rho s^2 \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} = \begin{pmatrix} (\lambda+2\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) \\ \mu\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) \end{pmatrix}.$$

Hence, by multiplying by $\nu_1\nu_2$ we get

$$\rho s^2 \nu_1\nu_2 \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} = \begin{pmatrix} (\lambda+2\mu)\frac{\partial}{\partial x}\left(\frac{\nu_2}{\nu_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{\partial}{\partial y}\left(\frac{\nu_1}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) + (\lambda+\mu)\frac{\partial^2 \hat{u}_2}{\partial x\partial y} \\ \mu\frac{\partial}{\partial x}\left(\frac{\nu_2}{\nu_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{\partial}{\partial y}\left(\frac{\nu_1}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) + (\lambda+\mu)\frac{\partial^2 \hat{u}_1}{\partial x\partial y} \end{pmatrix}.$$

In addition we have,

$$\begin{cases} \frac{\nu_1}{\nu_2} = \frac{s+\zeta_1}{s+\zeta_2} = \frac{s+\zeta_2-\zeta_2+\zeta_1}{s+\zeta_2} = 1 + \frac{\zeta_1-\zeta_2}{s+\zeta_2}, \\ \frac{\nu_2}{\nu_1} = 1 + \frac{\zeta_2-\zeta_1}{s+\zeta_1}. \end{cases}$$

So that, we obtain

$$\rho s^2 \nu_1\nu_2 \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} =$$
$$\begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 \hat{u}_1}{\partial x^2} + \mu\frac{\partial^2 \hat{u}_1}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_2}{\partial x\partial y} + (\lambda+2\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{\partial}{\partial y}\left(\frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_1}{\partial y}\right) \\ \mu\frac{\partial^2 \hat{u}_2}{\partial x^2} + (\lambda+2\mu)\frac{\partial^2 \hat{u}_2}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_1}{\partial x\partial y} + \mu\frac{\partial}{\partial x}\left(\frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{\partial}{\partial y}\left(\frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_2}{\partial y}\right) \end{pmatrix}.$$

Finally, we end-up with the following modified equation

$$\rho(s^2 + s(\zeta_1+\zeta_2) + \zeta_1\zeta_2) \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} =$$
$$\begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 \hat{u}_1}{\partial x^2} + \mu\frac{\partial^2 \hat{u}_1}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_2}{\partial x\partial y} \\ \mu\frac{\partial^2 \hat{u}_2}{\partial x^2} + (\lambda+2\mu)\frac{\partial^2 \hat{u}_2}{\partial y^2} + (\lambda+\mu)\frac{\partial^2 \hat{u}_1}{\partial x\partial y} \end{pmatrix} + \begin{pmatrix} (\lambda+2\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{\partial}{\partial y}\left(\frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_1}{\partial y}\right) \\ \mu\frac{\partial}{\partial x}\left(\frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{\partial}{\partial y}\left(\frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_2}{\partial y}\right) \end{pmatrix}.$$

This PML equation appears now as a modification of the elastodynamic equation. Unfortunately, we cannot apply the inverse Fourier-Laplace transform directly on this equation to go back to the time domain because of the algebraic fraction of s.

Step 3: Writing the PML system in its final form.
The coordinate transformation leads to powers of $i\omega$, positive and negative powers corresponding in the time domain to time derivatives and time integrations, respectively. If the time derivatives are not too troublesome in practice, we seek to get rid of the time integrations by introducing new unknowns. We will try to minimize the number of these new unknowns to limit the computational and memory cost they introduce. It is also worth noting that the PML formulation is second order as our original equation. For these reasons we decided to use Sim's formulation introduced in [**?**] over other formulations that are not second order in time or introduce more unknowns.

At this point we define auxiliary variables in order to get rid of the negative powers of s:

$$\tilde{\phi}_{11} = \frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_1}{\partial x}, \quad \tilde{\phi}_{12} = \frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_1}{\partial y},$$
$$\tilde{\phi}_{21} = \frac{\zeta_2-\zeta_1}{s+\zeta_1}\frac{\partial \hat{u}_2}{\partial x}, \quad \tilde{\phi}_{22} = \frac{\zeta_1-\zeta_2}{s+\zeta_2}\frac{\partial \hat{u}_2}{\partial y}.$$

We can rewrite the previous equations as the following equations

$$\begin{array}{ll} (s+\zeta_1)\tilde{\phi}_{11} = (\zeta_2-\zeta_1)\frac{\partial \hat{u}_1}{\partial x}, & (s+\zeta_2)\tilde{\phi}_{12} = (\zeta_1-\zeta_2)\frac{\partial \hat{u}_1}{\partial y}, \\ (s+\zeta_1)\tilde{\phi}_{21} = (\zeta_2-\zeta_1)\frac{\partial \hat{u}_2}{\partial x}, & (s+\zeta_2)\tilde{\phi}_{22} = (\zeta_1-\zeta_2)\frac{\partial \hat{u}_2}{\partial y}. \end{array}$$

Thus, we get the following system with only positive powers of s, but with four new unknowns:

$$\left\{\begin{array}{l} \rho(s^2+s(\zeta_1+\zeta_2)+\zeta_1\zeta_2)\begin{pmatrix}\hat{u}_1\\ \hat{u}_2\end{pmatrix} = \begin{pmatrix}(\lambda+2\mu)\frac{\partial^2\hat{u}_1}{\partial x^2}+\mu\frac{\partial^2\hat{u}_1}{\partial y^2}+(\lambda+\mu)\frac{\partial^2\hat{u}_2}{\partial x\partial y}\\ \mu\frac{\partial^2\hat{u}_2}{\partial x^2}+(\lambda+2\mu)\frac{\partial^2\hat{u}_2}{\partial y^2}+(\lambda+\mu)\frac{\partial^2\hat{u}_1}{\partial x\partial y}\end{pmatrix} \\ \qquad\qquad\qquad\qquad\qquad\qquad + \begin{pmatrix}(\lambda+2\mu)\frac{\partial\phi_{11}}{\partial x}+\mu\frac{\partial\phi_{12}}{\partial y}\\ \mu\frac{\partial\phi_{21}}{\partial x}+(\lambda+2\mu)\frac{\partial\phi_{22}}{\partial y}\end{pmatrix}, \\ (s+\zeta_1)\tilde{\phi}_{11} = (\zeta_2-\zeta_1)\frac{\partial \hat{u}_1}{\partial x}, \\ (s+\zeta_2)\tilde{\phi}_{12} = (\zeta_1-\zeta_2)\frac{\partial \hat{u}_1}{\partial y}, \\ (s+\zeta_1)\tilde{\phi}_{21} = (\zeta_2-\zeta_1)\frac{\partial \hat{u}_2}{\partial x}, \\ (s+\zeta_2)\tilde{\phi}_{22} = (\zeta_1-\zeta_2)\frac{\partial \hat{u}_2}{\partial y}. \end{array}\right.$$

Step 4: Inverse Fourier-Laplace transform.
Finally, we apply the inverse Fourier-Laplace transform and obtain the PML system of equations for the second order linear elastodynamic problem

$$\left\{\begin{array}{l} \rho\dfrac{\partial^2\mathbf{u}}{\partial t^2}+\rho(\zeta_1+\zeta_2)\dfrac{\partial\mathbf{u}}{\partial t}+\rho\zeta_1\zeta_2\mathbf{u} = div(\sigma(\mathbf{u}))+div(\Phi:\underline{\underline{\phi}}), \\ \dfrac{\partial\underline{\underline{\phi}}}{\partial t} = \Psi_1:\underline{\underline{\phi}}+\Psi_2:\nabla\mathbf{u}, \end{array}\right. \tag{2.5}$$

where $\underline{\underline{\phi}} = \begin{pmatrix}\phi_{11} & \phi_{12}\\ \phi_{21} & \phi_{22}\end{pmatrix}$ is a second order tensor, $.:.$ is a component wise product and

$$\Phi = \begin{pmatrix}\lambda+2\mu & \mu\\ \mu & \lambda+2\mu\end{pmatrix}, \quad \Psi_1 = \begin{pmatrix}-\zeta_1 & -\zeta_2\\ -\zeta_1 & -\zeta_2\end{pmatrix}, \quad \Psi_2 = \begin{pmatrix}\zeta_2-\zeta_1 & \zeta_1-\zeta_2\\ \zeta_2-\zeta_1 & \zeta_1-\zeta_2\end{pmatrix}.$$

Remark 2.1. *The tensor $\underline{\underline{\phi}}$ introduces four new unknowns, with the two unknowns of the displacement this leads to a memory cost three times higher in the PML domain. Fortunately, these unknowns only exist in the PML domain.*

Property 2.1
The PML formulation is stable and strongly well-posed. We refer to [**?** **?**] for the proof of these properties.

2.2.3 Truncation of the PML domain

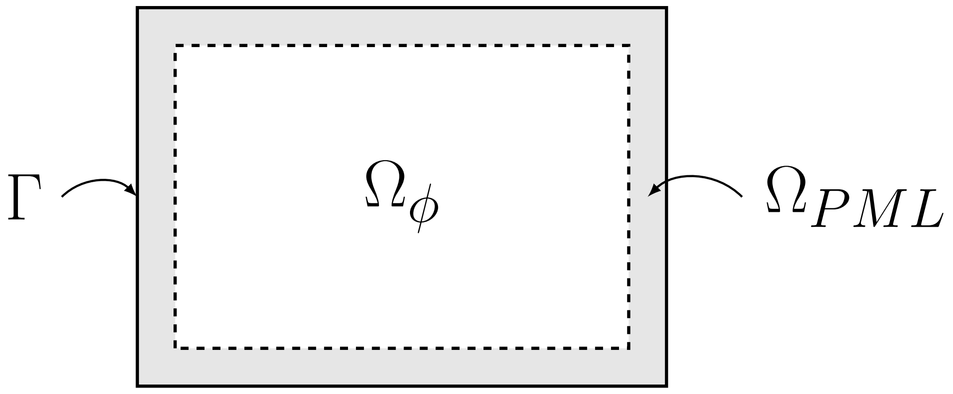


Figure 2.2: $\Omega = \Omega_\phi \cup \Omega_{PML}$.

It is important to keep in mind that PML are perfectly matched only for the continuous problem in an unbounded domain or with dumping functions that tend to infinity in a finite distance. The purpose of PML is to have a simulated unbounded domain within a bounded domain, thus Ω_{PML} needs to be truncated. Truncating at a finite thickness the PML makes them no longer perfectly absorbing, and reflected waves appear. However, PML are nevertheless very attractive as these reflections can be controlled easily to achieve the desired accuracy through appropriate dumping functions. Moreover, the quality of the absorption is not very dependent on the angle of incidence of waves contrary to *absorbing boundary conditions* [**?**].

Truncating the PML consists in adding Dirichlet boundary conditions to bound our PML domain. The thickness of the PML and the dumping functions must be chosen together to get the desired absorption. However, as we shall see in the numerical experiments the thickness and the dumping functions should be chosen carefully according to the discretization in order to avoid a poor absorption. For this reason we discarded the option of using dumping functions that tend to infinity in a finite distance.

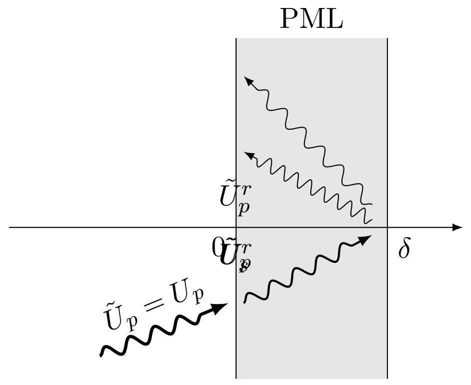


Figure 2.3: Reflection of a plane wave P in a finite PML.

As we mentioned earlier, waves decrease exponentially in the PML, thus the reflection coefficient becomes quickly very small. Through a plane wave analysis F.Collino and C. Tsogka showed in [**?**] that the reflection coefficients $r^\delta_{pp}, r^\delta_{ps}, r^\delta_{ss}, r^\delta_{sp}$ for plane waves solutions, which is the ratio between the amplitude of a P- or S-wave entering the PML (denoted by the first letter p or s in subscript) and the amplitude of the corresponding P- or S-wave outgoing the PML (denoted by the second letter p or s in subscript) in $x = 0$ (see Figure 2.3) after reflecting on the Dirichlet boundaries, are

$$r_{pp}^{\delta} = r_{pp} e^{-2\frac{\cos\theta}{v_p}\int_0^{\delta}\zeta(s)\,ds}, \tag{2.6}$$

$$r_{ps}^{\delta} = r_{ps} e^{-2\frac{\cos\theta}{v_p}\int_0^{\delta}\zeta(s)\,ds}, \tag{2.7}$$

$$r_{ss}^{\delta} = r_{ss} e^{-2\frac{\cos\theta}{v_s}\int_0^{\delta}\zeta(s)\,ds}, \tag{2.8}$$

$$r_{sp}^{\delta} = r_{sp} e^{-2\frac{\cos\theta}{v_s}\int_0^{\delta}\zeta(s)\,ds}, \tag{2.9}$$

where $r_{pp}, r_{ps}, r_{ss}, r_{sp}$ are the reflection coefficients on a Dirichlet boundary condition, θ is the angle of incidence and δ the thickness of the PML.

The truncated PML system for the second order can be written as follows
Find $(\mathbf{u}, \underline{\underline{\phi}})$ such that

$$\begin{cases}
\rho\dfrac{\partial^2\mathbf{u}}{\partial t^2} + \rho(\zeta_1+\zeta_2)\dfrac{\partial\mathbf{u}}{\partial t} + \rho\zeta_1\zeta_2\mathbf{u} = div(\sigma(\mathbf{u})) + div(\Phi : \underline{\underline{\phi}}) + f, & \text{in } \Omega,\\
\dfrac{\partial\underline{\underline{\phi}}}{\partial t} = \Psi_1 : \underline{\underline{\phi}} + \Psi_2 : \nabla\mathbf{u}, & \text{in } \Omega,\\
\mathbf{u} = 0, & \text{on } \Gamma,\\
\mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}), & \text{in } \Omega_{\phi},\\
\mathbf{u}(0,\mathbf{x}) = 0, & \text{in } \Omega_{PML},\\
\dfrac{\partial\mathbf{u}}{\partial t}(0,\mathbf{x}) = \mathbf{v}_0(\mathbf{x}), & \forall\mathbf{x}\in\Omega_{\phi},\\
\dfrac{\partial\mathbf{u}}{\partial t}(0,\mathbf{x}) = 0, & \forall\mathbf{x}\in\Omega_{PML},\\
\underline{\underline{\phi}}(0,\mathbf{x}) = 0, & \text{in } \Omega_{PML},\\
\dfrac{\partial\underline{\underline{\phi}}}{\partial t}(0,\mathbf{x}) = 0, & \forall\mathbf{x}\in\Omega_{PML},
\end{cases} \tag{2.10}$$

where $\underline{\underline{\phi}}$ is a second order tensor and

$$\Phi = \begin{pmatrix}\lambda+2\mu & \mu\\ \mu & \lambda+2\mu\end{pmatrix}, \quad \Psi_1 = \begin{pmatrix}-\zeta_1 & -\zeta_2\\ -\zeta_1 & -\zeta_2\end{pmatrix}, \quad \Psi_2 = \begin{pmatrix}\zeta_2-\zeta_1 & \zeta_1-\zeta_2\\ \zeta_2-\zeta_1 & \zeta_1-\zeta_2\end{pmatrix}.$$

2.3 Numerical schemes for the PML model

2.3.1 Discontinuous Galerkin approximation

We now introduce the Discontinuous Galerkin approximation of the PML system that we have considered. We use the same approach as previously, an interior penalty discontinuous Galerkin, to build this approximation.

Theorem 2.1
The PML system (2.10) is equivalent to the following discontinuous Galerkin system
Find $\forall t \in [0,T]$, $(\mathbf{u}(t,.), \underline{\underline{\phi}}(t,.)) \in H^{1+s}(\mathcal{T}_h)^2 \times H^s(\mathcal{T}_h)^4$, $s > \frac{1}{2}$, such that

$$\begin{cases} \forall \mathbf{v} \in H^s(\mathcal{T}_h)^2, & (\rho \partial_{tt}\mathbf{u}, \mathbf{v})_\Omega + (\rho(\zeta_1+\zeta_2)\partial_t \mathbf{u}, \mathbf{v})_\Omega + (\rho\zeta_1\zeta_2\mathbf{u}, \mathbf{v})_\Omega = a(\mathbf{u},\mathbf{v}) + b(\underline{\underline{\phi}}, \mathbf{v}), \\ \forall \varphi \in H^s(\mathcal{T}_h)^4, & (\partial_t \underline{\underline{\phi}}, \varphi)_\Omega = c(\underline{\underline{\phi}}, \varphi) + d(\mathbf{u}, \varphi), \end{cases}$$

where

$$\begin{aligned} a(\mathbf{u},\mathbf{v}) &= -\sum_{K\in\mathcal{T}_h} \int_K \underline{\underline{\sigma}}(u)\cdot\nabla v\,dx + \sum_{F\in\mathcal{F}_h}\int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\}\cdot[\![v]\!]\,ds \\ &\quad + \sum_{F\in\mathcal{F}_h}\int_F [\![u]\!]\cdot\{\!\{\underline{\underline{\sigma}}(v)n\}\!\}\,ds - \sum_{F\in\mathcal{F}_h}\int_F \alpha_F[\![u]\!]\cdot[\![v]\!]\,ds, \\ b(\underline{\underline{\phi}},\mathbf{v}) &= \sum_{K\in\mathcal{T}_h}\int_K div(\Phi:\underline{\underline{\phi}})\cdot v\,dx + \sum_{F\in\mathcal{F}_h}\int_F [\![(\Phi:\underline{\underline{\phi}})n]\!]\cdot\{\!\{v\}\!\}\,ds, \\ c(\underline{\underline{\phi}},\varphi) &= \sum_{K\in\mathcal{T}_h}\int_K (\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx, \\ d(\mathbf{u},\varphi) &= \sum_{K\in\mathcal{T}_h}\int_K (\Psi_2:\nabla u)\cdot\varphi\,dx + \sum_{F\in\mathcal{F}_h}\int_F [\![u]\!]\cdot\{\!\{(\Psi_2:\varphi)n\}\!\}\,ds. \end{aligned}$$

and

$$\Phi = \begin{pmatrix} \lambda+2\mu & \mu \\ \mu & \lambda+2\mu \end{pmatrix}, \quad \Psi_1 = \begin{pmatrix} -\zeta_1 & -\zeta_2 \\ -\zeta_1 & -\zeta_2 \end{pmatrix}, \quad \Psi_2 = \begin{pmatrix} \zeta_2-\zeta_1 & \zeta_1-\zeta_2 \\ \zeta_2-\zeta_1 & \zeta_1-\zeta_2 \end{pmatrix}.$$

Proof. The steps to follow in order to get the discontinuous Galerkin approximation are standard:

1. Multiply each equation by a suited test function,
2. Integrate each equation on Ω,
3. Use Green's formula in order to obtain the flux terms and relax constraints on derivatives to obtain the so called weak formulation.

Step 1: Multiply all equations by test functions.
We multiply the first equation of the system (2.5) by a sufficiently smooth test function v and the second equation by a sufficiently smooth test function φ. We obtain the system

$$\begin{cases} \rho\dfrac{\partial^2 u}{\partial t^2}\cdot v + \rho(\zeta_1+\zeta_2)\dfrac{\partial u}{\partial t}\cdot v + \rho\zeta_1\zeta_2 u\cdot v = \\ div(\underline{\underline{\sigma}}(u))\cdot v + div(\Phi:\underline{\underline{\phi}})\cdot v, \\ \dfrac{\partial \underline{\underline{\phi}}}{\partial t}\cdot\varphi = (\Psi_1:\underline{\underline{\phi}})\cdot\varphi + (\Psi_2:\nabla u)\cdot\varphi. \end{cases}$$

Step 2: Integration on the domain Ω.

$$\begin{cases} \displaystyle\int_\Omega \rho\frac{\partial^2 u}{\partial t^2}\cdot v\,dx + \int_\Omega \rho(\zeta_1+\zeta_2)\frac{\partial u}{\partial t}\cdot v\,dx + \int_\Omega \rho\zeta_1\zeta_2 u\cdot v\,dx = \\ \displaystyle\int_\Omega div(\underline{\underline{\sigma}}(u))\cdot v\,dx + \int_\Omega div(\Phi:\underline{\underline{\phi}})\cdot v\,dx, \\ \displaystyle\int_\Omega \frac{\partial \underline{\underline{\phi}}}{\partial t}\cdot\varphi\,dx = \int_\Omega (\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx + \int_\Omega (\Psi_2:\nabla u)\cdot\varphi\,dx. \end{cases}$$

As $\Omega = \bigcup\limits_{K\in\mathcal{T}_h} K$, we write previous integrals on Ω as a sum of integrals on each element

$$\begin{cases}
\displaystyle\sum_{K\in\mathcal{T}_h}\left(\int_K \rho\frac{\partial^2 u}{\partial t^2}\cdot v\,dx + \int_K \rho(\zeta_1+\zeta_2)\frac{\partial u}{\partial t}\cdot v\,dx + \int_K \rho\zeta_1\zeta_2 u\cdot v\,dx\right) = \\
\displaystyle\sum_{K\in\mathcal{T}_h}\left(\int_K div(\underline{\underline{\sigma}}(u))\cdot v\,dx + \int_K div(\Phi:\underline{\underline{\phi}})\cdot v\,dx\right), \\
\displaystyle\sum_{K\in\mathcal{T}_h}\int_K \frac{\partial\underline{\underline{\phi}}}{\partial t}\cdot\varphi\,dx = \sum_{K\in\mathcal{T}_h}\left(\int_K(\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx + \int_K(\Psi_2:\nabla u)\cdot\varphi\,dx\right).
\end{cases}$$

Step 3: Application of Green's formula.
We first recall Green's formula for our problem:

$$\int_K div(\underline{\underline{\sigma}}(u))\cdot v\,dx = -\int_K \underline{\underline{\sigma}}(u)\cdot\nabla v\,dx + \int_{\partial K}(\underline{\underline{\sigma}}(u)n)\cdot v\,ds.$$

As for classical IPDG formulation we have

$$\sum_{K\in\mathcal{T}_h}\int_{\partial K}(\underline{\underline{\sigma}}(u)n)\cdot v\,ds = \sum_{F\in\mathcal{F}_h}\int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\}\cdot[\![v]\!]\,ds.$$

Thus, we obtain

$$\begin{cases}
\displaystyle\sum_{K\in\mathcal{T}_h}\left(\int_K \rho\frac{\partial^2 u}{\partial t^2}\cdot v\,dx + \int_K \rho(\zeta_1+\zeta_2)\frac{\partial u}{\partial t}\cdot v\,dx + \int_K \rho\zeta_1\zeta_2 u\cdot v\,dx\right) = \\
\displaystyle -\sum_{K\in\mathcal{T}_h}\int_K \underline{\underline{\sigma}}(u)\cdot\nabla v\,dx + \sum_{F\in\mathcal{F}_h}\int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\}\cdot[\![v]\!]\,ds \\
\displaystyle +\sum_{K\in\mathcal{T}_h}\int_K div(\Phi:\underline{\underline{\phi}})\cdot v\,dx + \sum_{F\in\mathcal{F}_h}\int_F [\![(\Phi:\underline{\underline{\phi}})n]\!]\cdot\{\!\{v\}\!\}\,ds, \\
\displaystyle\sum_{K\in\mathcal{T}_h}\int_K \frac{\partial\underline{\underline{\phi}}}{\partial t}\cdot\varphi\,dx = \sum_{K\in\mathcal{T}_h}\int_K(\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx + \sum_{K\in\mathcal{T}_h}\int_K(\Psi_2:\nabla u)\cdot\varphi\,dx \\
\displaystyle +\sum_{F\in\mathcal{F}_h}\int_F [\![u]\!]\cdot\{\!\{(\Psi_2:\varphi)n\}\!\}\,ds.
\end{cases}$$

We add the classical SIPDG symmetric term $\displaystyle\int_F [\![u]\!]\cdot\{\!\{\underline{\underline{\sigma}}(v)n\}\!\}\,ds$ and the penalty term $\displaystyle -\int_F \alpha_F[\![u]\!]\cdot[\![v]\!]\,ds$, thus, we finally obtain the weak PML formulation

$$\begin{cases}
\displaystyle\sum_{K\in\mathcal{T}_h}\left(\int_K \rho\frac{\partial^2 u}{\partial t^2}\cdot v\,dx + \int_K \rho(\zeta_1+\zeta_2)\frac{\partial u}{\partial t}\cdot v\,dx + \int_K \rho\zeta_1\zeta_2 u\cdot v\,dx\right) = \\
\displaystyle -\sum_{K\in\mathcal{T}_h}\int_K \underline{\underline{\sigma}}(u)\cdot\nabla v\,dx + \sum_{F\in\mathcal{F}_h}\int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\}\cdot[\![v]\!]\,ds + \sum_{F\in\mathcal{F}_h}\int_F [\![u]\!]\cdot\{\!\{\underline{\underline{\sigma}}(v)n\}\!\}\,ds \\
\displaystyle -\sum_{F\in\mathcal{F}_h}\int_F \alpha_F[\![u]\!]\cdot[\![v]\!]\,ds + \sum_{K\in\mathcal{T}_h}\int_K div(\Phi:\underline{\underline{\phi}})\cdot v\,dx + \sum_{F\in\mathcal{F}_h}\int_F [\![(\Phi:\underline{\underline{\phi}})n]\!]\cdot\{\!\{v\}\!\}\,ds, \\
\displaystyle\sum_{K\in\mathcal{T}_h}\int_K \frac{\partial\underline{\underline{\phi}}}{\partial t}\cdot\varphi\,dx = \sum_{K\in\mathcal{T}_h}\int_K(\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx + \sum_{K\in\mathcal{T}_h}\int_K(\Psi_2:\nabla u)\cdot\varphi\,dx \\
\displaystyle +\sum_{F\in\mathcal{F}_h}\int_F [\![u]\!]\cdot\{\!\{(\Psi_2:\varphi)n\}\!\}\,ds.
\end{cases}$$

where

$$\Phi = \begin{pmatrix} \lambda+2\mu & \mu \\ \mu & \lambda+2\mu \end{pmatrix} \quad \Psi_1 = \begin{pmatrix} -\zeta_1 & -\zeta_2 \\ -\zeta_1 & -\zeta_2 \end{pmatrix} \Psi_2 = \begin{pmatrix} \zeta_2-\zeta_1 & \zeta_1-\zeta_2 \\ \zeta_2-\zeta_1 & \zeta_1-\zeta_2 \end{pmatrix}$$

□

2.3.2 Spatial semi-discrete formulation

In order to get the spatial discretization we need to choose approximating subspaces of $H^s(\mathcal{T}_h)^2$ and $H^s(\mathcal{T}_h)^4$ called finite element spaces. We still take polynomial approximating spaces.

For a given partition $\mathcal{T}_h$ of Ω and an approximation order $k \geq 1$, we wish to approximate $\mathbf{u}(t,.)$ in the finite element space

$$V_h := \{\mathbf{v} \in L^2(\Omega)^2 \ : \ \forall K \in \mathcal{T}_h \ \mathbf{v}|_K \in \mathcal{Q}^k(K)^2\},$$

and $\underline{\underline{\phi}}(t,.)$ in the finite element space

$$W_h := \{\varphi \in L^2(\Omega)^4 \ : \ \forall K \in \mathcal{T}_h \ \varphi|_K \in \mathcal{Q}^k(K)^4\},$$

where $\mathcal{Q}^k(K)$ are spaces of polynomials of degree at most k in each variable on K.

Remark 2.2. *Here again we could use any approximating subspace of $H^s(\Omega)^2$, $s > \frac{3}{2}$ instead.*

Let $K \in \mathcal{T}_h$, we denote by $\{\varphi_i^K\}$ and $\{\psi_i^K\}$ a basis of $V_h(K)$ and $W_h(K)$, respectively. Let $N^K = |\hat{A}\ \{\varphi_i\}|$ and $N_\phi^K = \{\psi_i\}|$ denote the number of degrees of freedom associated with the displacement $\mathbf{u}$ and the PML unknowns on the element K, respectively.

We shall now express the approximated solutions $\mathbf{u}_h(t,\mathbf{x})$ and $\underline{\underline{\phi}}_h(t,\mathbf{x})$ in these spaces.

2.3.2.1 Global formulation of the spatial discretization

The semi-discrete solution can be expanded in the local basis functions by

$$\forall t \in [0,T],\ \forall \mathbf{x} \in \Omega, \quad \mathbf{u}_h(t,\mathbf{x}) = \sum_{K\in\mathcal{T}_h} \sum_{i=1}^{N^K} U_{ind_K(i)}(t)\varphi_i^K(\mathbf{x}),$$

and

$$\forall t \in [0,T],\ \forall \mathbf{x} \in \Omega, \quad \underline{\underline{\phi}}_h(t,\mathbf{x}) = \sum_{K\in\mathcal{T}_h} \sum_{i=1}^{N_\phi^K} \Phi_{ind_{\phi,K}(i)}(t)\psi_i^K(\mathbf{x}),$$

where $ind_K(i)$ and $ind_{\phi,K}(i)$ are global indexing functions on $[\![1,N]\!]$ and $[\![1,N_\phi]\!]$ respectively.

The global space discretization of the PML is

$$\begin{cases} M\dfrac{\partial^2 U}{\partial t^2} + M_{\zeta_1+\zeta_2}\dfrac{\partial U}{\partial t} + M_{\zeta_1\zeta_2}U = K_\sigma U + K_\Phi \Phi, \\ M_\Phi \dfrac{\partial \Phi}{\partial t} = K_{\Psi_1}\Phi + K_{\Psi_2}U. \end{cases} \tag{2.11}$$

where

$$\begin{aligned}
&\forall i,j \in [\![1,N]\!] \quad M_{ij} = (\rho\varphi_j,\varphi_i)_\Omega, \quad M_{\zeta_1+\zeta_2,ij} = (\rho(\zeta_1+\zeta_2)\varphi_j,\varphi_i)_\Omega,\\
&\forall i,j \in [\![1,N]\!] \quad M_{\zeta_1\zeta_2,ij} = (\rho\zeta_1\zeta_2\varphi_j,\varphi_i)_\Omega, \quad K_{\sigma,ij} = a_h(\varphi_j,\varphi_i),\\
&\forall i \in [\![1,N]\!]\ \forall j \in [\![1,N_\phi]\!] \quad K_{\Phi,ij} = b_h(\psi_j,\varphi_i),\\
&\forall i,j \in [\![1,N_\phi]\!] \quad M_{\Phi,ij} = (\psi_j,\psi_i)_\Omega, \quad K_{\Psi_1,ij} = c_h(\psi_j,\psi_i),\\
&\forall i \in [\![1,N_\phi]\!]\ \forall j \in [\![1,N]\!] \quad K_{\Psi_2,ij} = d_h(\phi_j,\psi_i).
\end{aligned}$$

2.3.2.2 Local formulation of the spatial discretization

The global formulation is simple to read, but has a major drawback, it hides all the locality of the discontinuous Galerkin and consequently all the attractiveness and difficulties of the method. For this reason, we prefer to rewrite these equations in a local form.

The semidiscrete solution can also be expanded in the local basis functions by

$$\forall t \in [0,T],\ \forall x \in \Omega, \quad \mathbf{u}_h(t,\mathbf{x}) = \sum_{K\in\mathcal{T}_h}\sum_{i=1}^{N^K} u_i^K(t)\varphi_i^K(\mathbf{x}),$$

and

$$\forall t \in [0,T],\ \forall x \in \Omega, \quad \underline{\underline{\phi}}_h(t,\mathbf{x}) = \sum_{K\in\mathcal{T}_h}\sum_{i=1}^{N_\phi^K} \phi_i^K(t)\psi_i^K(\mathbf{x}).$$

First, we define local operators as follow

$$\begin{aligned}
a^K(\mathbf{u},\mathbf{v}) &:= -\int_K \sigma(\mathbf{u})\cdot\nabla\mathbf{v}\,dx + \sum_{F\in\mathcal{F}_K}\int_F \{\!\{\sigma(\mathbf{u})n\}\!\}\cdot\mathbf{v}|_K\,ds\\
&\quad + \sum_{F\in\mathcal{F}_K}\int_F [\![\mathbf{u}]\!]\cdot\sigma^K(\mathbf{v}|_K)n_K\,ds - \sum_{F\in\mathcal{F}_K}\int_F \alpha_F[\![\mathbf{u}]\!]\cdot\mathbf{v}|_K\,ds,\\
b^K(\underline{\underline{\phi}},\mathbf{v}) &:= \int_K div(\Phi:\underline{\underline{\phi}})\cdot\mathbf{v}\,dx + \sum_{F\in\mathcal{F}_K}\int_F [\![(\Phi:\underline{\underline{\phi}})n]\!]\cdot\mathbf{v}|_K\,ds,\\
c^K(\underline{\underline{\phi}},\varphi) &:= \int_K (\Psi_1:\underline{\underline{\phi}})\cdot\varphi\,dx,\\
d^K(\mathbf{u},\varphi) &:= \int_K (\Psi_2:\nabla\mathbf{u})\cdot\varphi\,dx + \sum_{F\in\mathcal{F}_K}\int_F [\![\mathbf{u}]\!]\cdot(\Psi_2:\varphi|_K)n_K\,ds.
\end{aligned}$$

To obtain the local formulation of the variational formulation we have to consider test functions which are not null only on the considered element K, thus we obtain the following local variational formulation

$$\begin{cases} \rho_K M^K \dfrac{\partial^2 u^K}{\partial t^2} + \rho_K M^K_{\zeta_1+\zeta_2} \dfrac{\partial u^K}{\partial t} + \rho_K M^K_{\zeta_1\zeta_2} u^K = K^K_\sigma u^K + \displaystyle\sum_{F\in\mathcal{F}_K} F^{V_F(K)}_\sigma u^{V_F(K)} \\ + K^K_\Phi \phi^K + \displaystyle\sum_{F\in\mathcal{F}_K} F^{V_F(K)}_\Phi \phi^{V_F(K)}, \\ M^K \dfrac{\partial \phi^K}{\partial t} = K^K_{\Psi_1}\phi^K + K^K_{\Psi_2} u^K + \displaystyle\sum_{F\in\mathcal{F}_K} F^{V_F(K)}_{\Psi_2} u^{V_F(K)}, \end{cases} \tag{2.12}$$

where
$\forall i,j \in [\![1, N^K]\!]$

$$M^K_{ij} = (\rho_K \varphi^K_j, \varphi^K_i)_K,$$

$$M^K_{\zeta_1+\zeta_2,ij} = (\rho_K(\zeta_1+\zeta_2)\varphi^K_j, \varphi^K_i)_K,$$

$$M^K_{\zeta_1\zeta_2,ij} = (\rho_K \zeta_1\zeta_2 \varphi^K_j, \varphi^K_i)_K,$$

$$K^K_{\sigma,ij} = a^K_h(\varphi^K_j, \varphi^K_i),$$

$\forall i \in [\![1, N^K]\!]\ \forall j \in [\![1, N^{V_F(K)}]\!]$

$$F^{V_F(K)}_{\sigma,ij} = a^{V_F(K)}_h(\varphi^{V_F(K)}_j, \varphi^K_i),$$

$\forall i \in [\![1, N^K]\!]\ \forall j \in [\![1, N^K_\phi]\!]$

$$K^K_{\Phi,ij} = b^K_h(\psi^K_j, \varphi^K_i),$$

$\forall i \in [\![1, N^K]\!]\ \forall j \in [\![1, N^{V_F(K)}_\phi]\!]$

$$F^{V_F(K)}_{\Phi,ij} = b^{V_F(K)}_h(\psi^{V_F(K)}_j, \varphi^K_i),$$

$\forall i,j \in [\![1, N^K_\phi]\!]$

$$K^K_{\Psi_1,ij} = c^K_h(\psi^K_j, \psi^K_i),$$

$\forall i \in [\![1, N^K_\phi]\!]\ \forall j \in [\![1, N^K]\!]$

$$K^K_{\Psi_2,ij} = d^K_h(\psi^K_j, \varphi^K_i),$$

$\forall i \in [\![1, N^K_\phi]\!]\ \forall j \in [\![1, N^{V_F(K)}]\!]$

$$F^{V_F(K)}_{\Psi_2,ij} = d^{V_F(K)}_h(\psi^{V_F(K)}_j, \varphi^K_i),$$

2.3.3 Full discretization

There are plenty of ways to achieve the temporal discretization in order to get the full discretization. Different discretizations result in different stabilities and sensibilities to the PML parameters. Especially, the CFL condition might be weakened by the absorption strength [**?**]. We decided to take inspiration from a temporal discretization described in [**?**] for first order hyperbolic systems in order to have a CFL condition as little affected as possible. As we mention later in the numerical results (see Section 2.4.1.2), we use exactly the same CFL as without PML.

For the second order derivative we take a standard second order centered scheme

$$\frac{d^2\mathbf{u}}{dt^2}(t_n) \simeq \frac{u_{n+1} - 2u_n + u_{n-1}}{\Delta t^2}.$$

For the first order derivative we take centered scheme:

$$\frac{d\mathbf{u}}{dt}(t_n) \simeq \frac{u_{n+1} - u_{n-1}}{2\Delta t},$$

this choice is led by the desire to have a discrete scheme symmetric in time. Since there is a second order time derivative, this first order centered scheme does not lead to an unstable scheme as we shall see in the numerical result section short after. Finally, inspired by [?], and to continue having a symmetric scheme in time, we used

$$\mathbf{u}(t_n) \simeq \frac{u_{n+1} + 2u_n + u_{n-1}}{4}.$$

Moreover, we center the first equation of the system (2.12) in time n and the second in time $n + \frac{1}{2}$.

These temporal discretizations lead to

$$\begin{cases} \rho_K M^K \dfrac{u^K_{n+1} - 2u^K_n + u^K_{n-1}}{\Delta t^2} + \rho_K M^K_{\zeta_1+\zeta_2} \dfrac{u^K_{n+1} - u^K_{n-1}}{2\Delta t} \\ + \rho_K M^K_{\zeta_1\zeta_2} \dfrac{u^K_{n+1} + 2u^K_n + u^K_{n-1}}{4} = \Theta_1(u_n, \phi_n), \\ M^K \dfrac{\phi^K_{n+1} - \phi^K_n}{\Delta t} = \Theta_2(\dfrac{u_{n+1} + u_n}{2}, \dfrac{\phi_{n+1} + \phi_n}{2}), \end{cases}$$

where Θ_1 and Θ_2 represent the "spatial" parts of the PML system.

Hence, if we write the recurrence induced by this temporal discretization, we get

$$\begin{cases} \rho_K(M^K + 2\Delta t M^K_{\zeta_1+\zeta_2} + \dfrac{\Delta t^2}{4} M^K_{\zeta_1\zeta_2}) u^K_{n+1} = \\ \rho_K(2M^K - \dfrac{\Delta t^2}{2} M^K_{\zeta_1\zeta_2}) u^K_n \\ + \rho_K(-M^K + \Delta t M^K_{\zeta_1+\zeta_2} - \dfrac{\Delta t^2}{4} M^K_{\zeta_1\zeta_2}) u^K_{n-1} \\ + \Delta t^2 \Theta_1(u_n, \phi_n), \\ M^K \phi^K_{n+1} = M^K \phi^K_n + \Delta t \Theta_2(\dfrac{u_{n+1} + u_n}{2}, \dfrac{\phi_{n+1} + \phi_n}{2}). \end{cases}$$

2.4 Numerical results

In this section, we investigate the main numerical features of the PML scheme we have defined. We will present several examples of elastic wave propagation simulations.

When adding PML around a physical domain to simulate an unbounded domain, we need to choose the dumping functions. For the continuous truncated PML it has been shown in [?] that the theoretical reflection coefficient $r = r^\delta_{pp}$ (see relation (2.6) for $\theta = 0$) for plane wave solutions, which is the ratio between the amplitude of an incident wave and the amplitude of the corresponding reflected wave, is

$$r = e^{-\frac{2}{v}\int_0^\delta \zeta(s)\, ds},$$

where v is the velocity of the considered plane wave and δ the thickness of the PML. Thus, for quadratic dumping functions, we have

$$r = e^{-\frac{2}{v}\int_0^\delta s^2\, ds} = e^{-\frac{2\delta^3}{3v}}.$$

We now consider standard "normalized" quadratic dumping functions ζ_i defined in [? ?] as follows:

$$\zeta_i(\mathbf{x}) = \begin{cases} \frac{3c}{2\delta^3} log(\frac{1}{r})(x_i^{min} - x_i)^2 & ,\forall x_i \leq x_i^{min}, \\ 0 & ,\forall x_i^{min} \leq x_i \leq x_i^{max}, \\ \frac{3c}{2\delta^3} log(\frac{1}{r})(x_i^{max} - x_i)^2 & ,\forall x_i \geq x_i^{max}, \end{cases}$$

where c is the largest velocity in Ω_{PML} and r the theoretical reflection coefficient becomes the desired absorption.

The profile of the dumping functions ζ_i must not be too steep, otherwise it results in a bad discretization of the PML causing spurious effects and even unstabilities. Thus, the thickness of the PML δ and the desired absorption r have to be chosen carefully as they rule the slope of the dumping functions. δ has to be chosen according to the desired absorption that depends on the *largest* velocity, and of the *smallest* wave length in order to have a good discretization of the PML. For a chosen absorption r, if δ is too large then we have unnecessary memory and computation cost, if δ is too small then the PML will not be efficient and can even be unstable.

For our numerical experiments we use an explosive source located at the point $\mathbf{x}_S$, that is

$$\mathbf{f}(\mathbf{x}, t) = h(t) g(|\mathbf{x} - \mathbf{x}_S|) \frac{\overrightarrow{\mathbf{x} - \mathbf{x}_S}}{|\mathbf{x} - \mathbf{x}_S|}$$

where $h(t)$ is a second order Ricker, with central frequency $f_0 = 40Hz$,

$$h(t) = (2\pi^2(f_0 t - 1)^2 - 1)e^{-\pi^2(f_0 t - 1)^2},$$

and $g(|\mathbf{x} - \mathbf{x}_S|)$ is a regularization of a Dirac by a Gaussian centered in $\mathbf{x}_S = (300m, 300m)$ and distributed over a disk of radius $r_0 = 8m$,

$$g(|\mathbf{x} - \mathbf{x}_S|) = \frac{e^{-7\frac{|\mathbf{x} - \mathbf{x}_S|^2}{r_0^2}}}{r_0^2}.$$

2.4.1 Homogeneous medium test case

In this first experiment we want to show that PML have the expected behavior in an homogeneous medium under some constraints.

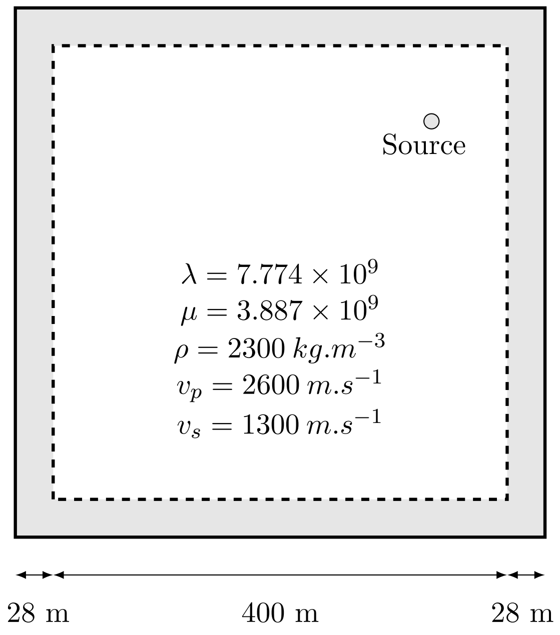

Source
$\lambda = 7.774 \times 10^9$
$\mu = 3.887 \times 10^9$
$\rho = 2300\, kg.m^{-3}$
$v_p = 2600\, m.s^{-1}$
$v_s = 1300\, m.s^{-1}$
28 m
400 m
28 m

Figure 2.4: Homogeneous medium characteristics.

We consider an homogeneous medium with $\rho = 2300\, kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$ ($v_s = 1300\, m.s^{-1}$ and $v_p = 2600\, m.s^{-1}$). The physical domain is of size $400m \times 400m$. The initial conditions are null. We used $\mathcal{Q}_3$ elements of size $4m$ for these simulations, with Legendre-Gauss function basis (see figure 1.3.3). We add PML around our physical computation domain, as the longest wavelength is $\lambda_{max} = \frac{c}{2.5 f_0} = 26m$, we take PML of thickness $\delta = 28m = 7cells \simeq \lambda_{max}$ and $r = 10^{-4}$ as often suggested [? ?].

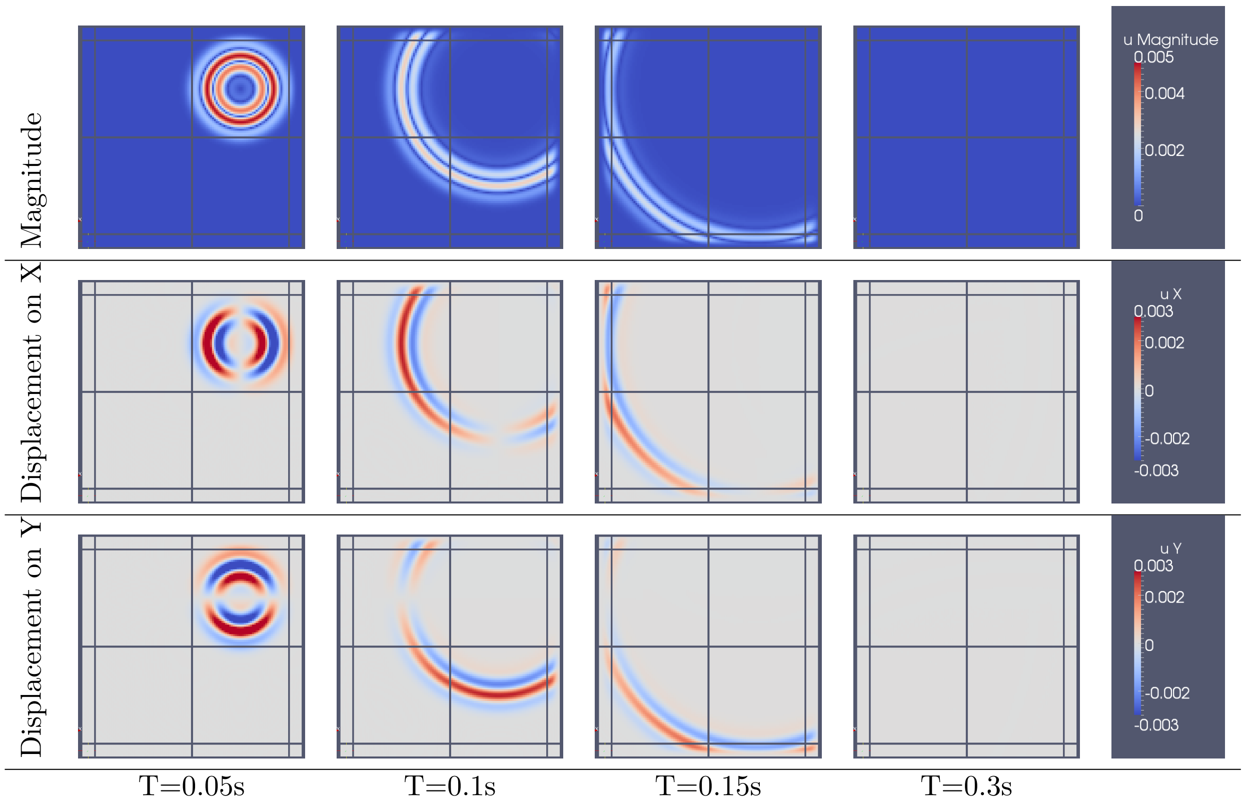
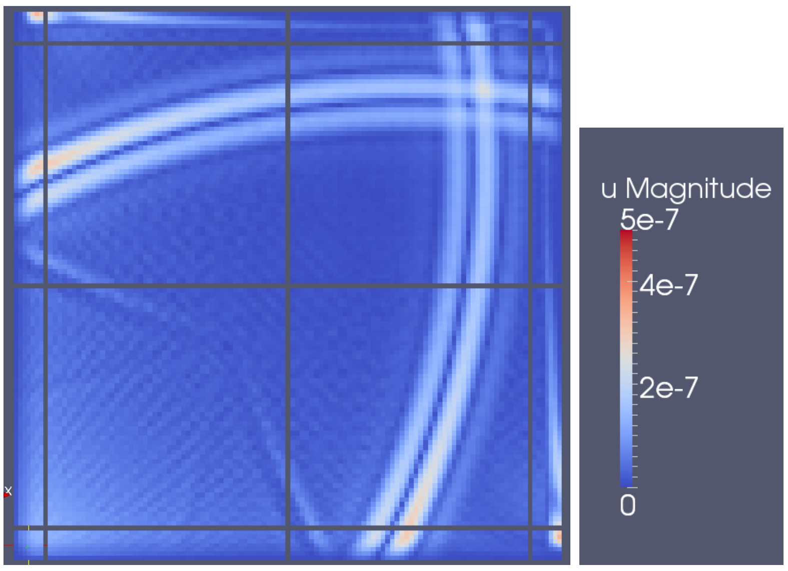


Figure 2.5: Magnitude and amplitude of the displacement at different times, for an homogeneous medium with $\rho = 2300kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$.



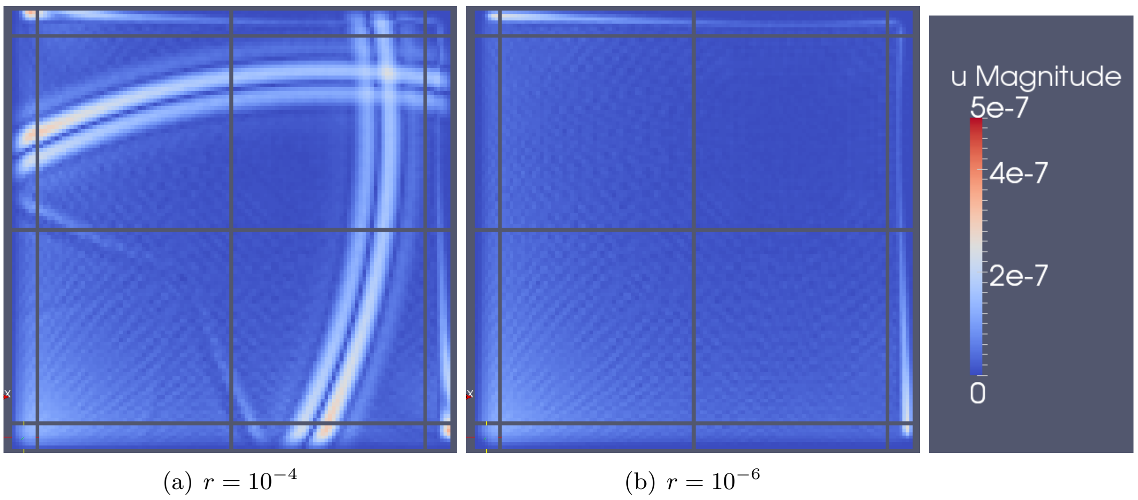
(a) $r = 10^{-4}$

Figure 2.6: Magnitude of the displacement for a color scale divided by a factor of $1/r = 10^4$ at $T = 0.3s$, for an homogeneous medium with $\rho = 2300kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$.

We first present some snapshots of the solution on the normal scale in Figure 2.5. We can remark in the snapshots presented in 2.5 that we cannot see any reflection on the normal scale. To see some reflections we have to magnify the results. We present in Figure 2.6 the results magnified by the invert of the desired absorption, that is $1/r = 10^4$, and we remark that the reflections are of the expected amplitude. We can also see that the PML are well discretized has no other spurious effects than the expected reflected waves are noticeable.
This first result on an homogeneous medium is interesting. Nothing in the study of reflection coefficient achieved for plane waves suggested that the theoretical reflection coefficients would be correct for other kinds of waves.

2.4.1.1 Impact of the absorption coefficient and of the thickness of the PML

Here we want to show what can be the impact of both the absorption coefficient r and the thickness δ. To show the effects we take a constant thickness for the PML of $\delta = 7cells = 28m$.

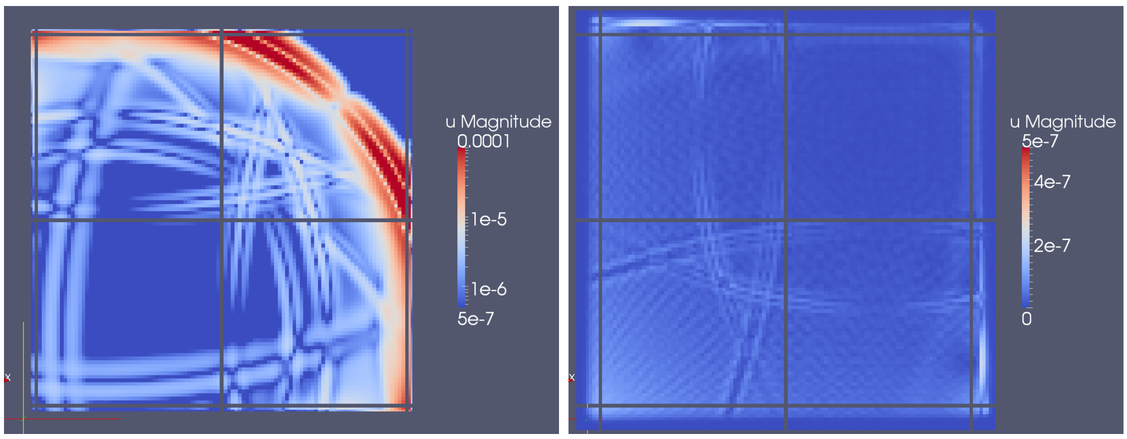


(a) $r = 10^{-4}$ (b) $r = 10^{-6}$

Figure 2.7: Magnitude of the displacement for a color scale divided by a factor of 10^4 for different absorption coefficients $r = 10^{-4}$ and $r = 10^{-6}$ at $T = 0.3s$, for an homogeneous medium with $\rho = 2300kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$.

As we can see in Figure 2.7(b) with an absorption coefficient of $r = 10^{-6}$ the reflected waves amplitude is now below the dispersion amplitude, and thus the PML become numerically perfectly matched and perfectly absorbing.

We show now in Figure 2.8 what happens if we push too hard these parameters, either with too thin PML or too large absorption.



(a) $\delta = 8m, r = 10^{-8}$ (b) $\delta = 28m, r = 10^{-20}$

Figure 2.8: Magnitude of the displacement amplified by a factor of 1000 for different PML thickness at $T = 0.30s$, for an homogeneous medium with $\rho = 2300kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$.

These last parameters taken for Figure 2.8 might seem exaggerated, but when PML are used in heterogeneous materials, we must keep in mind that the number of point per wavelength to be taken depends on the *shortest wavelength*, whereas the thickness of the PML depends on the *highest velocity*. Thus one might quickly end up with one of the cases introduced here.
However, we remark in Figure 2.8 that having too short PML thickness δ has a much more serious effect on spurious reflection than having a too high absorption coefficient r. It is easy to see why, as the dependency of our dumping functions ζ on δ is cubic whereas the dependency on r is logarithmic, shortening the PML has much more impact on the slope of our dumping functions than increasing the desired reflection r.
We also remark on Figure 2.8(b) that spurious S-waves are the first to appear, this can be explained by the fact that they are the one with the shortest wavelength and consequently suffer the most of a bad discretization.

2.4.1.2 Stability and impact on the CFL condition

In this section, we study the behavior of our PML scheme on the time step and on the stability on long time simulations.
E. Bécache and A. Prieto in [**?**] emphasized the impact of the choice of the time discretization on the maximum time step allowed for a chosen absorption. We compare the optimal time step allowed with PML to the optimal time step on the same computational domain with Dirichlet conditions called Δt_{CFL}. Through all the numerical experiments we performed, the optimal time step was never impacted by the PML.

Remark 2.3. *We saw that PML become unstable for the smallest penalty values. However, we recommend not using the smallest values of the penalty as it has an important impact on the stability of the error, even if the penalty has an impact on the CFL condition.*

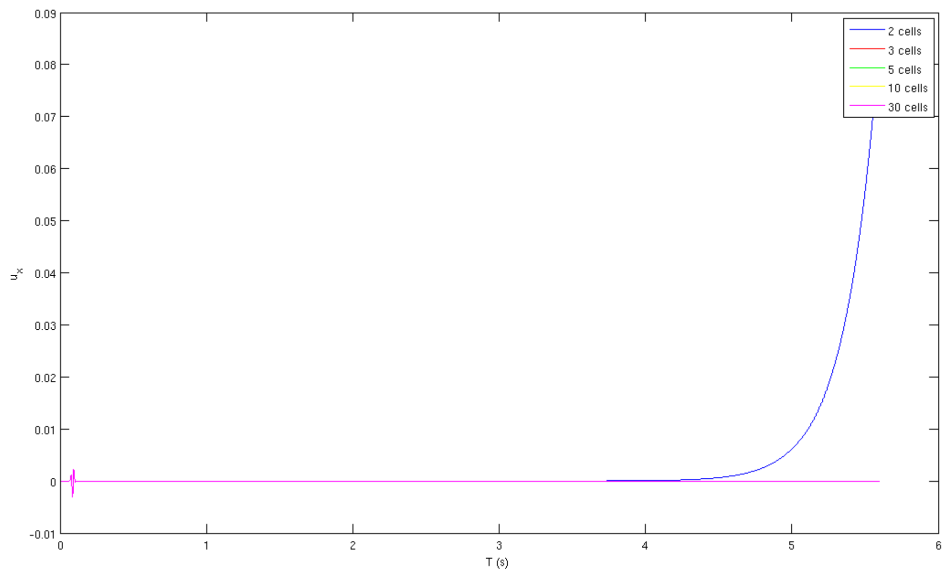


Figure 2.9: Sismos on long time simulations for different PML thickness δ.

As shown on Figure 2.9 the PML might become unstable for too violent absorption requirements, however these cases correspond to cases where PML are not well discretized and thus do not absorb correctly.

2.4.2 Simple heterogeneous medium test case

In this second experiment we want to show that the PML behave well in a simple heterogeneous medium.

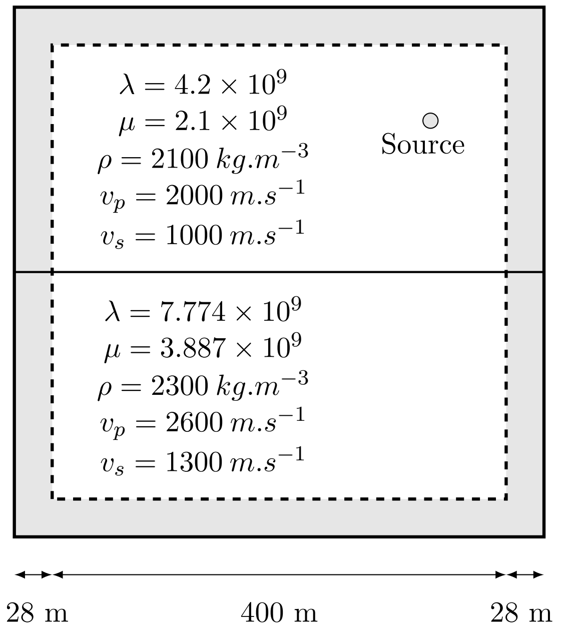
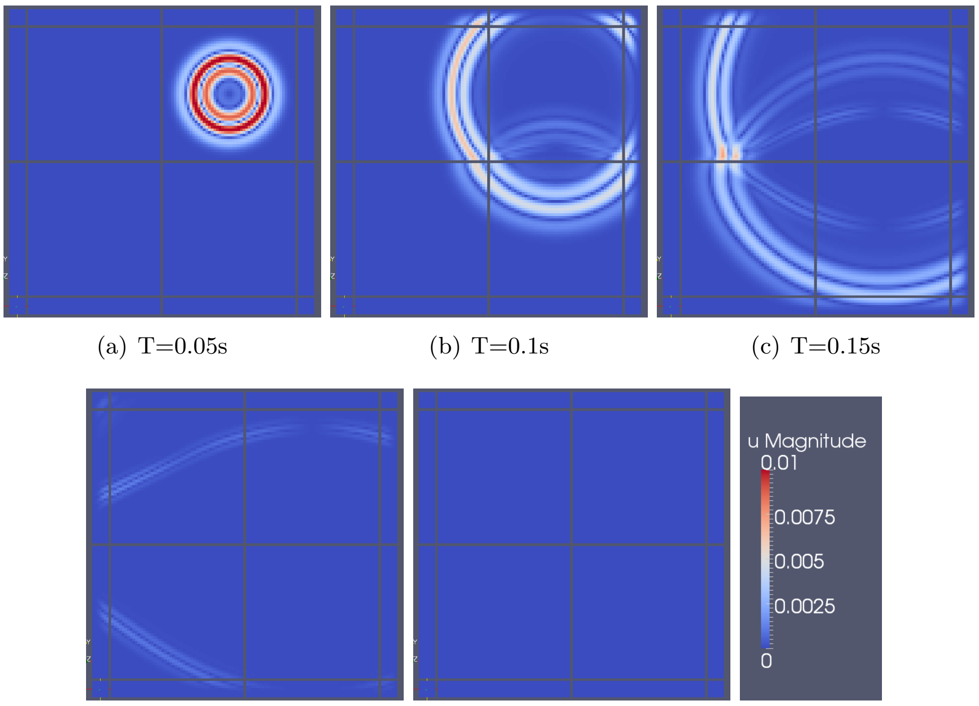


Figure 2.10: Heterogeneous medium characteristics.

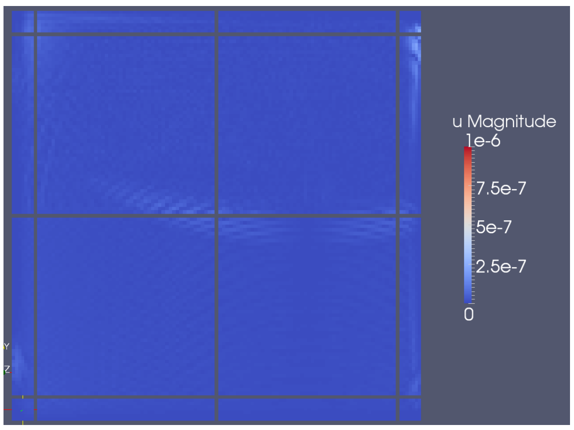
We consider an heterogeneous medium with $\rho = 2300\,kg.m^{-3}$, $\lambda = 7.774 \times 10^9$ and $\mu = 3.887 \times 10^9$ ($v_s = 1300\,m.s^{-1}$ and $v_p = 2600\,m.s^{-1}$) in the bottom half space, and with $\rho = 2100\,kg.m^{-3}$, $\lambda = 4.2 \times 10^9$ and $\mu = 2.1 \times 10^9$ ($v_s = 1000\,m.s^{-1}$ and $v_p = 2000\,m.s^{-1}$) in the top half space. The physical domain is of size $400m \times 400m$. The initial conditions are null. We used $\mathcal{Q}_3$ elements of size $4m$ for these simulations, with Legendre-Gauss function basis (see Figure 1.3.3).



(a) T=0.05s (b) T=0.1s (c) T=0.15s

(d) T=0.3s (e) T=0.5s

Figure 2.11: Magnitude of the displacement at different times, for an heterogeneous medium (see Figure 2.10).



(a) $r = 10^{-4}$

Figure 2.12: Magnitude of the displacement for a color scale divided by a factor of $1/r = 10^4$ at $T = 0.5s$, for an heterogeneous medium (see figure 2.10).

We can remark in the snapshots presented on Figure 2.11 that we cannot see any reflections. To see some reflection, we again have to magnify the scale by a factor equal to the invert of the absorption $1/r = 10^4$. As we can see on Figure 2.12, the reflected waves are of the expected amplitudes, and thus the PML behave well for an heterogeneous medium too.

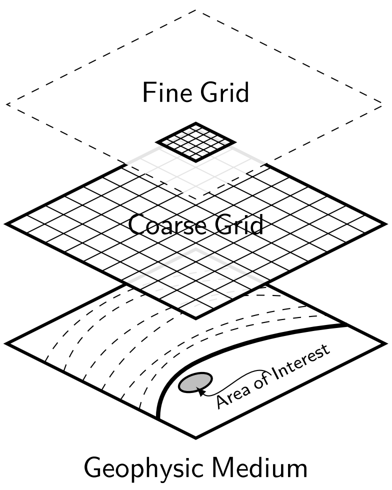
2.5 Conclusion

We have presented a second order PML formulation and his discontinuous Galerkin approximation for the second order elastodynamic equation. We introduced a time discretization that overcome the problem of having a CFL condition weakened by the PML absorption, therefore the CFL condition in the PML is identical to the CFL condition of the discrete scheme in the physical domain.

Chapter 3

Using local time stepping with non-conforming Cartesian space refinement

3.1 Introduction



In some parts of the physical domain, if we want to take into account the geometrical details or capture a singularity of the solution, it is tempting to use techniques of spatial local mesh refinement. Since the time step is conditioned by the smallest spatial element, it results in a substantial increase in computation cost. It is therefore natural to want to use a local time-stepping method in such configurations.

At this point, we shall remind the constraint that meshes are Cartesian grids as well as each refined area. This leads to the need to be able to deal with non-conforming meshes. Since our meshes were only Cartesian grids we focused first on finite difference refinement methods. Most methods to achieve space-time local mesh refinement were interpolation techniques [? ? ? ?], but they encountered stability problems. Then came the conservative methods [? ?] that first seek to ensure the stability of the scheme through the conservation of a discrete energy. The first approaches that used this idea were coupling two problems, the coarse and the fine, through transmission conditions imposed by a Lagrange multiplier. This approach has several drawbacks, first the Lagrange multiplier requires the solution of a linear system, whereas we wanted to stay fully explicit (we did not want to solve any linear system), and some local high frequency spurious effects can appear [?]. The need to have non-conforming Cartesian grids is one of the main reasons why we decided to use discontinuous Galerkin methods since they can naturally handle these configurations and yield really low spurious effects on the non-conforming interfaces as we shall see. This important requirement also made us discard the interesting so called ADER local time-stepping method [?], since this last method is more suited for highly heterogeneous elements as each element can have its own time step. Our approach is different, we see the local time-stepping and the local mesh refinement as two independent problems. For this reason, we decided to use the conservative local time-stepping scheme introduced by J. Diaz and M. Grote in [**?**], and a conservative non-conforming local space refinement through discontinuous Galerkin finite elements methods.

In Section 3.2 we introduce J. Diaz and M. Grote's strategy to overcome the stability constraint imposed by the smallest element. This results in a first scheme which we call the "$\tilde{z}$-exact" scheme. We found interesting to study this $\tilde{z}$-exact scheme because of the numerical properties it shares with J. Diaz and M. Grote's local time stepping scheme. Then, we introduce J. Diaz and M. Grote's local time stepping algorithm as an approximation of the $\tilde{z}$-exact scheme. In Section 3.3, we investigate and discuss the cost of local space-time refinement. We emphasize our non-conforming and p-adaptive approach that enables a number of optimizations to control the rapid growth of the computational cost. In Section 3.4 we perform experiments to illustrate the appealing numerical features of the proposed schemes and its flexibility.

3.2 Local time stepping method: Diaz-Grote's formulation

The usual way to achieve local mesh refinement is to discretize independently in time the coarse and fine parts, and then find coupling transmission conditions. Those transmission conditions have been either interpolation (which is unstable), or the introduction of a Lagrange multiplier (that leads to a linear system). On the contrary, in Diaz-Grote's scheme, the time discretization is applied regardless of the fine part. The main idea of J. Diaz and M. Grote is to get a "better" approximation of the second order time derivative in such a way that the global stability of the scheme is unchanged. Then, the remaining problem is how to calculate this improved approximation of the time derivative. In that respect, a new problem is introduced to get a better approximation of the fine part time derivative. At this point there still is no local fine time step, and we can solve this new problem analytically as we will see. Unfortunately the analytical solution is unrealistic in practice. This new problem is thus discretized to be solved numerically. Using a similar time discretization as for the original problem, a leap-frog scheme, at a smaller time step. Put in another way, we have a first problem that we temporally discretize regardless of the fine area, and a second problem concerning only the fine area which is solved at a smaller time step.

3.2.1 Construction of Diaz-Grote's $\tilde{z}$-exact scheme

The classic way to approximate the second order derivative is to take an order 0 approximation that is $u''(t + \Delta t) \simeq u''(t)$. Since J. Diaz and M. Grote's idea is to propose a better approximation of this term, we will first introduce the improved approximation of the second order time derivative. This choice leads to a new scheme which we call the "$\tilde{z}$-exact" formulation, we emphasize that this scheme will not contain any local time step. Diaz-Grote's algorithm is based on a discrete approximation of this scheme. Then, we study the numerical behavior of this approximation and deduct some numerical properties. This preliminary study reveals that most of the numerical properties of the "$\tilde{z}$-exact" scheme will be similar to those of Diaz-Grote's local time stepping scheme.

First of all, we want to give the intuition why improving the second order derivative allows a larger time step Δt than the one dictated by the smallest element.

We shall begin with the time discretization we use, the leap-frog scheme, with the integral form of the remainder:

$$u(t_{n+1}) - 2u(t_n) + u(t_{n-1}) = \Delta t^2 \int_{-1}^{1} (1 - |\theta|) u''(t_n + \theta\Delta t) d\theta. \tag{3.1}$$

Using $u''(t) + Au(t) = 0$, we get

$$u(t_{n+1}) - 2u(t_n) + u(t_{n-1}) = -\Delta t^2 \int_{-1}^{1} (1 - |\theta|) Au(t_n + \theta\Delta t) d\theta,$$

where $A = M^{-1}K$ in our case.

The construction of the leap-frog scheme is done considering the second order derivative constant on the interval $[t_{n-1}, t_{n+1}]$, *i.e.* $\forall\theta \in [-1, 1], \quad Au(t_n + \theta\Delta t) \simeq Au(t_n)$. In this case, if we denote by u_n the approximation of $u(t_n)$ we get the following standard leap-frog scheme:

$$u_{n+1} - 2u_n + u_{n-1} = -\Delta t^2 Au_n. \tag{3.2}$$

Unfortunately, the stability condition of this scheme, $\Delta t \leq 2/\sqrt{\lambda_{\max}(A)}$, is linked to the largest eigenvalue of the matrix A, $\lambda_{\max}(A)$, which is proportional to $1/h_f^2$ with h_f the space step of the fine part. In other words, Δt is globally penalized by the fine part, which is very binding and unrealistic.

The principle of the construction of Diaz-Grote's scheme is to improve the previous scheme by considering the following approximation:

$$\begin{aligned} Au(t_n + \theta\Delta t) &= Au^{[coarse]}(t_n + \theta\Delta t) + Au^{[fine]}(t_n + \theta\Delta t) \\ &\simeq A(I - P)u(t_n) + APu(t_n + \theta\Delta t), \end{aligned} \tag{3.3}$$

where P is the canonical restriction to the fine part, we also note $Q = (I - P)$.

We assume that the degrees of freedom are sorted in the following form

$$u = \begin{pmatrix} u^{[coarse]} \\ u^{[fine]} \end{pmatrix}. \tag{3.4}$$

We have not done much yet since we cannot do anything of the approximation (3.3), we now have to define an approximation of $APu(t_n + \theta\Delta t)$.

A classic way to do this is to add unknowns $u^{n+m/p}$ (for $m = -p + 1, ..., p - 1$) approximation of $u(t_n + m/p\Delta t)$ in the fine part. We then calculate the unknowns in the coarse and fine parts using a leap-frog scheme and a specific treatment must be done to link the two parts, while ensuring the stability and order of consistency of the overall scheme. This task is difficult to achieve optimally.

To avoid this difficulty, J. Diaz and M. Grote use a different approach. Their idea is to construct directly a global approximation of $APu(t_n + \theta\Delta t)$ and not to make a global connection afterwards. To do this, they introduced the following second order ordinary

differential equation:
Find $\tilde{z} : [-\Delta t, \Delta t] \to \mathbb{R}^d$ solution of

$$\begin{cases} \tilde{z}''(\tau) &= -A(I-P)u(t_n) - AP\tilde{z}(\tau), \\ \tilde{z}(0) &= u(t_n), \\ \tilde{z}'(0) &= \nu, \end{cases} \tag{3.5}$$

where $\nu \in \mathbb{R}^n$ is a free constant vector parameter to be precised later on.

Remark 3.1. *It is important to remember for later that $\tilde{z}$ is only intended to provide $\tilde{z}''(\tau) = u''(t_n + \tau)$ and not to give an approximation of u! It should be noted that $\tilde{z}(\tau)$ is generally not a "good" approximation of $u(t_n+\tau)$. This is usually an order 1 approximation because of the first initial condition. We can nevertheless improve things by taking $\nu = u'(t_n)$ but we see later that this choice is not the most appropriate.*

3.2.1.1 The $\tilde{z}$-exact formulation

Let us recall that the purpose of $\tilde{z}$ defined by the differential problem (3.5) is to give a better approximation of u''. We expect that the stability condition would be less constrained by the area with a specific treatment, or even unconstrained. We shall note that there is not any sort of local time stepping scheme at this moment. We only have a scheme with two different approximations of u''.

First, we give some properties on the differential problem (3.5), which will help us defining what we call the $\tilde{z}$-exact formulation, and finally we give some properties about this $\tilde{z}$-exact formulation.

We have the following results:

Property 3.1
- The expression of $\tilde{z}$ in the fine part is:
  $\forall \tau \in [-\Delta t, \Delta t]$

$$\begin{aligned} P\tilde{z}(\tau) &= \cos((PAP)^{1/2}\tau)Pu(t_n) + \sin((PAP)^{1/2}\tau)(PAP)^{-1/2}P\nu \\ &\quad + (\cos((PAP)^{1/2}\tau) - 1)(PAP)^{-1}PA(I-P)u(t_n). \end{aligned} \tag{3.6}$$

- $\tilde{z}''(\tau)$ is an approximation of $u''(t_n)$:
  $\forall \tau \in [-\Delta t, \Delta t]$

$$\begin{aligned} \tilde{z}''(\tau) &= u''(t_n) - \sum_{n=1}^{+\infty} \frac{(-1)^n}{(2n)} A(PAP)^n Pu(t_n)\tau^{2n} \\ &\quad - \sum_{n=0}^{+\infty} \frac{(-1)^n}{(2n+1)!} A(PAP)^n P\nu\tau^{2n+1} \\ &\quad + \sum_{n=1}^{+\infty} \frac{(-1)^n}{(2n)} A(PAP)^{n-1} PA(I-P)Pu(t_n)\tau^{2n}. \end{aligned} \tag{3.7}$$

- The expression of $\Theta_n := \Delta t^2 \int_{-1}^{1} (1-|\theta|)\tilde{z}''(\theta\Delta t)d\theta$ is:

$$\begin{aligned} \Theta_n =& 2A(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t) - I)Pu(t_n) \\ &+ 2A(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t) - I)(PAP)^{-1}PAQu(t_n) \\ &- \Delta t^2 AQu(t_n) + \Delta t^2 A(PAP)^{-1}PAQu(t_n). \end{aligned} \tag{3.8}$$

Remark 3.2. $(PAP)^{-1}$ *is the Moore-Penrose pseudoinverse since a large part of this matrix is null, but the non null block is invertible, this is why we kept this notation.*

Proof. To obtain the expression of $P\tilde{z}$, we have to solve the ordinary differential equation with constant coefficient (3.5) projected on the fine part by applying P.
First, we define the solution of the homogeneous problem:

$$P\tilde{z}_0''(\tau) \quad = \quad -PAP\tilde{z}_0(\tau). \tag{3.9}$$

Hence, we immediately get:

$$\tilde{z}_0(\tau) \quad = \quad \cos((PAP)^{1/2}\tau)\beta + \sin((PAP)^{1/2}\tau)\alpha, \tag{3.10}$$

where $\alpha,\, \beta \in \mathbb{R}^N$.

The matrices $\cos((PAP)^{1/2}\tau)$ and $\sin((PAP)^{1/2}\tau)$ are defined as follows. We diagonalize the symmetric positive definite matrix PAP *i.e.* $PAP = VDV^T$ where $D = diag(\lambda_i, i = 1, ..., N)$ and λ_i are the eigenvalues of PAP. We thus have $\cos((PAP)^{1/2}\tau) = V\, diag(\cos(\sqrt{\lambda_i}\tau))V^T$ (*idem* for sin).

Then, we can easily verify that $\tilde{z}_1 := -(PAP)^{-1}PAQu(t_n)$ is a particular solution of (3.5) (without the initial conditions). Finally, we seek $P\tilde{z}$ in the form $\tilde{z}_0 + \tilde{z}_1$ and the initial conditions determine the two values (α, β) such that

$$\tilde{z}_0(0) + \tilde{z}_1(0) = \beta - (PAP)^{-1}PAQu(t_n) = Pu(t_n),$$

which implies

$$\beta = Pu(t_n) + (PAP)^{-1}PAQu(t_n).$$

Besides,

$$\tilde{z}_o'(\tau) = -(PAP)^{1/2}\sin((PAP)^{1/2}\tau)\beta + (PAP)^{1/2}\cos((PAP)^{1/2}\tau)\alpha,$$

thus

$$\tilde{z}_0'(0) = (PAP)^{1/2}\alpha = P\nu,$$

which implies

$$\alpha = (PAP)^{-1/2}P\nu.$$

Finally,

$$\begin{aligned} P\tilde{z}(\tau) = &\cos((PAP)^{1/2}\tau)Pu(t_n) + \sin((PAP)^{-1/2}\tau)(PAP)^{-1/2}P\nu \\ &+ (\cos((PAP)^{1/2}\tau) - 1)(PAP)^{-1}PAQu(t_n). \end{aligned} \tag{3.11}$$

To prove (3.7), we simply use $\tilde{z}''(\tau) = -A(I-P)u(t_n) - AP\tilde{z}(\tau)$ and the power series of cos and sin:

$$\begin{aligned} \cos((PAP)^{1/2}\tau) &= \sum_{n=0}^{+\infty}(-1)^n \frac{\tau^{2n}}{(2n)!}(PAP)^n \\ \sin((PAP)^{1/2}\tau) &= \sum_{n=0}^{+\infty}(-1)^n \frac{\tau^{2n+1}}{(2n+1)!}(PAP)^{n+1/2} \end{aligned}$$

Finally, to prove (3.8), we use again $\tilde{z}''(\tau) = -A(I-P)u(t_n) - AP\tilde{z}(\tau)$, the expression (3.11) and the following integrals:

$$\begin{aligned}
\int_{-1}^{1} (1-|\theta|)\cos((PAP)^{1/2}\theta\Delta t)d\theta &= -\tfrac{2}{\Delta t^2}(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t)-1), \\
\int_{-1}^{1} (1-|\theta|)\sin((PAP)^{1/2}\theta\Delta t)d\theta &= 0.
\end{aligned}$$

□

We define a first temporal scheme (Diaz-Grote's scheme will be an approximation of this one) using the previous approximation $\tilde{z}''(\theta\Delta t)$ of $u''(t_n+\theta\Delta t)$. We have:

$$\boxed{u_{n+1} - 2u_n + u_{n-1} \;=\; \Delta t^2 \int_{-1}^{1}(1-|\theta|)\tilde{z}''(\theta\Delta t)d\theta.} \tag{3.12}$$

Using (3.8), the scheme (3.12) takes the form of the following leap-frog scheme:

$$\boxed{u_{n+1} - 2u_n + u_{n-1} \;=\; -\Delta t^2 \mathrm{B}u^n,} \tag{3.13}$$

where the matrix B is

$$B \;=\; \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

where

$$B_{11} = -\frac{2}{\Delta t^2}(\cos((PAP)^{1/2}\Delta t) - I)P,$$

$$B_{12} = B_{21}^T = -\frac{2}{\Delta t^2}(\cos((PAP)^{1/2}\Delta t) - I)(PAP)^{-1}PAQ,$$

$$B_{22} = -\frac{2}{\Delta t^2}QA(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t) - I)(PAP)^{-1}PAQ + (QAQ) - QA(PAP)^{-1}PAQ.$$

Remark 3.3. *There is an abuse of notation in the matrices B_{11}, B_{12}, B_{21} and B_{22}, they are the restriction to the corresponding non null matrix block since the size of B is the same as the size of A. Moreover, to write B in this form we used the assumption that the degrees of freedom of u are sorted as defined in* (3.4)*.*

Remark 3.4. *We note that the matrix B does not depend of the initial condition ν of the differential equation* (3.5)*.*

Remark 3.5. *It is noteworthy that the matrix B is symmetric.*

Property 3.2
The scheme (3.13) is consistent at the order 2 in time.

Proof. Using the power series of the cosine function, we immediately get:

$$\begin{aligned}
&-\frac{2}{\Delta t^2}(\cos((PAP)^{1/2}\Delta t) - I)Pu(t_n) \\
&= (PAP)u(t_n) - \frac{2}{\Delta t^2}\sum_{n=2}^{+\infty}\frac{(-1)^n\Delta t^{2n}}{(2n)!}(PAP)^n u(t_n),
\end{aligned} \tag{3.14}$$

$$\begin{aligned}&-\frac{2}{\Delta t^2}(\cos((PAP)^{1/2}\Delta t) - I)(PAP)^{-1}PAQu(t_n)\\&= (PAQ)u(t_n) - \frac{2}{\Delta t^2}\sum_{n=2}^{+\infty}\frac{(-1)^n\Delta t^{2n}}{(2n)!}(PAP)^{n-1}PAQu(t_n),\end{aligned} \tag{3.15}$$

$$\begin{aligned}&-\frac{2}{\Delta t^2}QA(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t) - I)P\\&= (QAP)u(t_n) - \frac{2}{\Delta t^2}\sum_{n=2}^{+\infty}\frac{(-1)^n\Delta t^{2n}}{(2n)!}QA(PAP)^{n-1}Pu(t_n),\end{aligned} \tag{3.16}$$

$$\begin{aligned}&-\frac{2}{\Delta t^2}QA(PAP)^{-1}(\cos((PAP)^{1/2}\Delta t) - I)(PAP)^{-1}PAQ\\&= (QA)(PAP)^{-1}(PAQ)u(t_n) - \frac{2}{\Delta t^2}\sum_{n=2}^{+\infty}\frac{(-1)^n\Delta t^{2n}}{(2n)!}QA(PAP)^{n-2}AQu(t_n).\end{aligned} \tag{3.17}$$

By grouping (3.14), (3.15), (3.16) and (3.17), we get:

$$\begin{aligned}Bu(t_n) &= \begin{bmatrix} PAP & PAQ \\ QAP & QAQ \end{bmatrix} u(t_n) + O(\Delta t^2),\\ &= Au(t_n) + O(\Delta t^2) = -u''(t_n) + O(\Delta t^2).\end{aligned} \tag{3.18}$$

We define the consistency error of the scheme by:

$$\Lambda_n := \frac{u(t_{n+1}) - 2u(t_n) + u(t_{n-1})}{\Delta t^2} + Bu(t_n). \tag{3.19}$$

Using the consistency result of the leap-frog scheme and (3.18), we get $\Lambda_n = O(\Delta t^2)$ and the scheme (3.13) is consistent at the order 2 in time. □

3.2.1.2 Stability analysis

We shall now investigate the main question of this method: the stability of the scheme (3.13).

Since the scheme (3.13) can be written as a leap-frog scheme, we have the following discrete energy:

$$E^{n+1/2} = \frac{1}{2}\left(\left\langle\left(I - \frac{\Delta t^2}{4}B\right)\frac{u_{n+1}-u_n}{\Delta t}, \frac{u_{n+1}-u_n}{\Delta t}\right\rangle + \left\langle B\frac{u_{n+1}+u_n}{2}, \frac{u_{n+1}+u_n}{2}\right\rangle\right). \tag{3.20}$$

A sufficient condition of stability is the positiveness of the energy.

Let $(\lambda_i^B)_{i=1,\cdots,N}$ denote the eigenvalues of B. The stability condition is:

$$\boxed{\text{Find } \Delta t_{\text{opt}} > 0 \quad \text{such that} \quad \forall \Delta t \le \Delta t_{\text{opt}}, \forall i = 1, \cdots, N,\ 0 \le \frac{\Delta t^2}{4}\lambda_i^B \le 1.} \tag{3.21}$$

One of the difficulties to do this analysis comes from the fact that the λ_i^B have a non-linear dependency with Δt. In order to give some answer, we perform a numerical analysis

of the 1D problem:
Find $u :]0, L[\to \mathbb{R}$ solution of

$$\left\{ \begin{array}{rcl} \rho \dfrac{\partial^2 u}{\partial t^2} - \mathrm{div}(\mu \nabla u) & = & f, \\ u(0,t) = u(L,t) & = & 0, \\ u(x,0) & = & u_0(x), \\ \dfrac{\partial u}{\partial t}(x,0) & = & v_0(x). \end{array} \right. \tag{3.22}$$

This problem has been discretized using the SIPDG method described in Chapter 1. The mesh used is described on Figure 3.1. On this figure, the intermediate area corresponds to coarse cells that will possibly be included in the fine area through the projection matrix P. We will see that adding this area will improve the spectral behavior of B and thus the stability leading to a larger time step $\Delta t_{\max}$. For this study, we denote by $\Delta t_{\min} := 2/\sqrt{\lambda_{\max}(A)}$ the time step that corresponds to the CFL condition when the standard leap-frog scheme is used. Finally, we define $\Delta t_{\max} := 2/\sqrt{\lambda_{\max}(QAQ)}$.

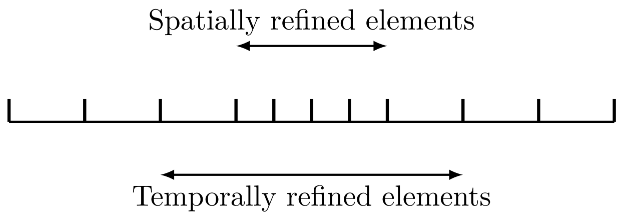


Figure 3.1: Description of the 1D mesh.

We display on Figure 3.2 the largest eigenvalues of $\frac{\Delta t^2}{4} B$ with a fine part spatially refined by $p_s = 2$. We note that the maximum time step is roughly at 60% of the desired time step. However, the Theorem 1.5 suggested that the coarse element right next to a fine element has a local CFL condition of the same kind as the small elements. Therefore, it is natural to include coarse elements right next to fine elements in the refined area. We call halo-n the set of coarse elements at a distance lower than or equal to n elements of a fine element which we include in the refined area.

Remark 3.6. *We study non exhaustively the stability of the scheme by an energy method. If we study the stability of the scheme (3.13) from an energetic point of view, the condition to be verified is*
$\forall U \in \mathbb{R}^N$,

$$\left\langle (I - \frac{\Delta t^2}{4} B) U, U \right\rangle \geq 0 \quad \textit{and} \quad \langle BU, U \rangle \geq 0.$$

We observe that if $QU = 0$ then this condition becomes:

$$\|U\|^2 - \frac{1}{2} \left\langle (I - \cos((PAP)^{1/2} \Delta t)) U, U \right\rangle \quad \geq \quad 0.$$

This condition is then verified for any time step Δt.

Now, if we take $PU = 0$ then the condition becomes:

$$\begin{aligned} \|U\|^2 - \frac{\Delta t^2}{4} \langle QAQU, U \rangle + \frac{\Delta t^2}{4} \Big\langle (PAP)^{-1} PAQU, PAQU \Big\rangle & \\ -\frac{1}{2} \Big\langle (I - \cos((PAP)^{1/2}\Delta t))(PAP)^{-1}PAQU, (PAP)^{-1}PAQU \Big\rangle & \geq 0. \end{aligned}$$

Using the power series of cos*, we can show that:*
$\forall \Delta t \geq 0$

$$\begin{aligned} \frac{\Delta t^2}{4} \Big\langle (PAP)^{-1} PAQU, PAQU \Big\rangle & \\ -\frac{1}{2} \Big\langle (I - \cos((PAP)^{1/2}\Delta t))(PAP)^{-1}PAQU, (PAP)^{-1}PAQU \Big\rangle & \geq 0. \end{aligned}$$

Hence, we have $\forall \Delta t \leq \Delta t_{max} := \dfrac{2}{\sqrt{\lambda_{max}(QAQ)}}$ *is verified for* $U \in \mathbb{R}^N$ *such that* $PU = 0$.

This quick investigation revealed that the fine part is always stable, the stability condition in the coarse part is exactly the same, thus if we observe any stability difference it will be coming from the transition between the coarse and the fine part.

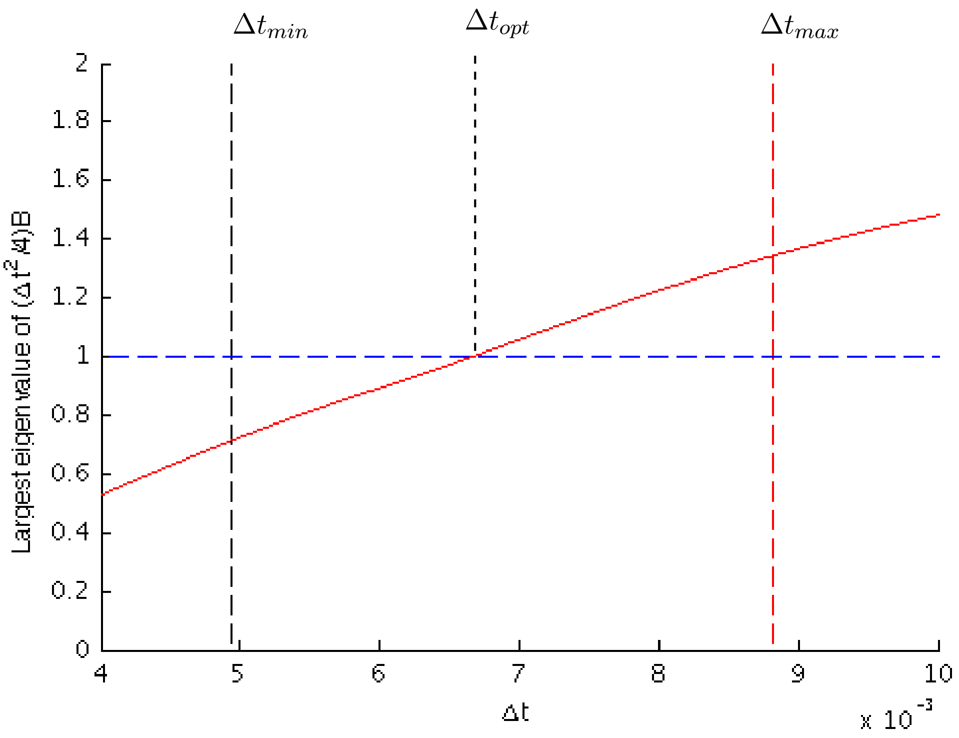


Figure 3.2: Evolution of the largest eigenvalue of $\frac{\Delta t^2}{4} B$ according to Δt for a spatial refinement $p_s = 2$.

We display on Figure 3.3 and 3.4 the largest eigenvalues (and a zoom around 1) of $\frac{\Delta t^2}{4} B$ with a fine part spatially refined by $p_s = 2$ with different size of halo. We note that as soon as the halo is superior or equal to 1 (halo-1 and halo-2) the scheme is stable at the optimal time step. Nevertheless, we note on Figure 3.4 that there are few intervals below the optimal value for which the scheme is unstable, and rising the size of the halo from 1 to 2 does improve greatly the situation with only really small intervals of unstability left.

We display on Figure 3.5 the smallest eigenvalues of $\frac{\Delta t^2}{4}B$ with a fine part spatially refined by $p_s = 2$. We note that the smallest eigenvalues behave well in all situations, and have no impact on the stability of the method.

We display on Figure 3.6 and 3.7 the largest eigenvalues (and a zoom around 1) of $\frac{\Delta t^2}{4}B$ with a fine part spatially refined by $p_s = 8$ this time. We note that this time without halo (halo-0) the maximum time step allowed is only about 20% of the optimal time step desired. However, once again using a halo size superior to 1 greatly improves the situation, and we recover an optimal time step. We also note on Figure 3.7 that with a halo size of 1, the number of unstable intervals is greater than with $p_s = 2$, however, with a halo size of 2, there only are few unstable intervals.

We display on Figure 3.8 the smallest eigenvalues of $\frac{\Delta t^2}{4}B$ with $p_s = 8$. We note that the size of the halo slightly impacts the smallest eigenvalues, however the smallest eigenvalue never leads to an unstable scheme in any case.

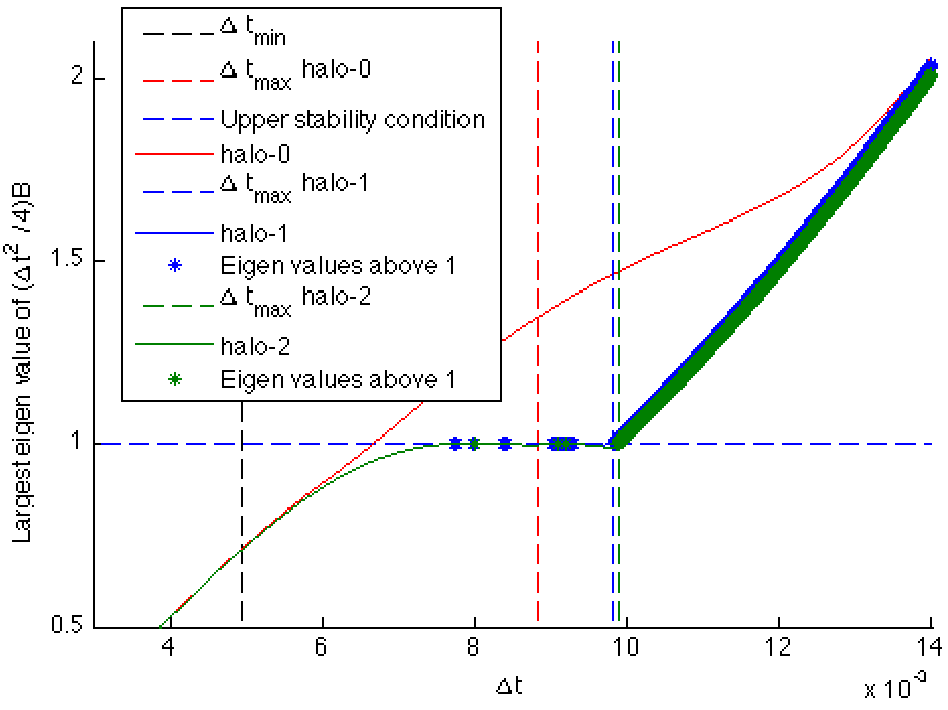


Figure 3.3: Evolution with or without halo of the largest eigenvalue of $\frac{\Delta t^2}{4}B$ according to Δt for a spatial refinement $p_s = 2$.

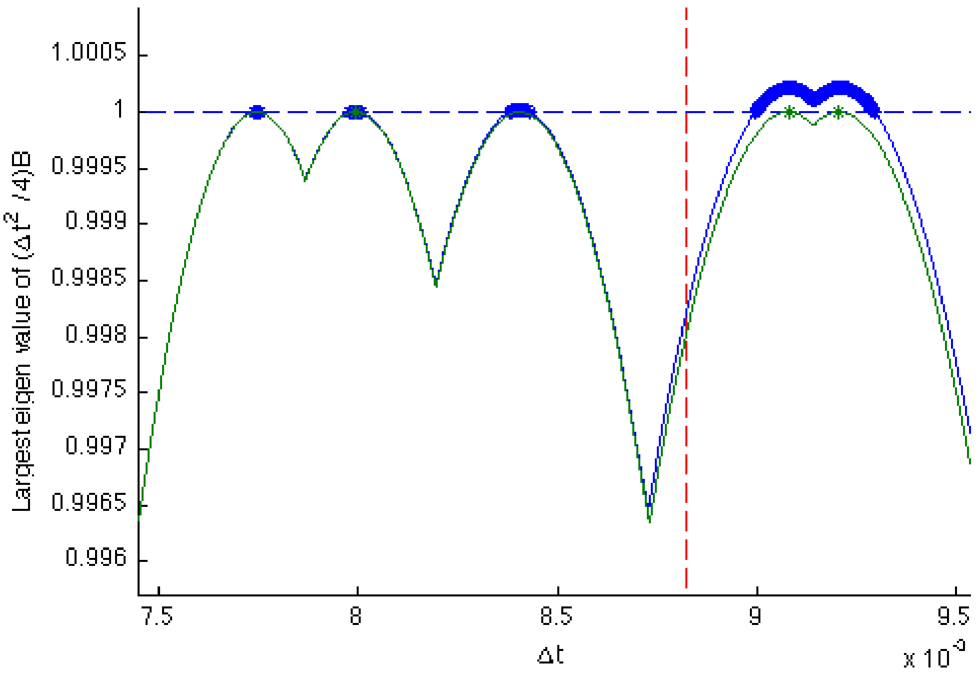


Figure 3.4: Zoom on Figure 3.3.

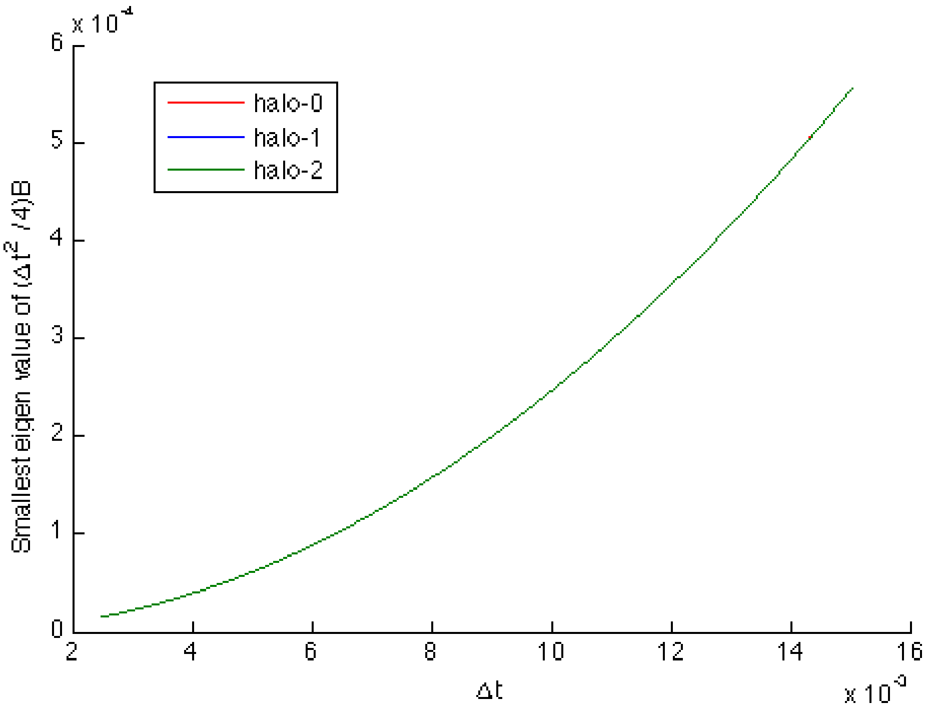


Figure 3.5: Evolution with or without halo of the smallest eigenvalue of $\frac{\Delta t^2}{4}B$ according to Δt for a spatial refinement $p_s = 2$.

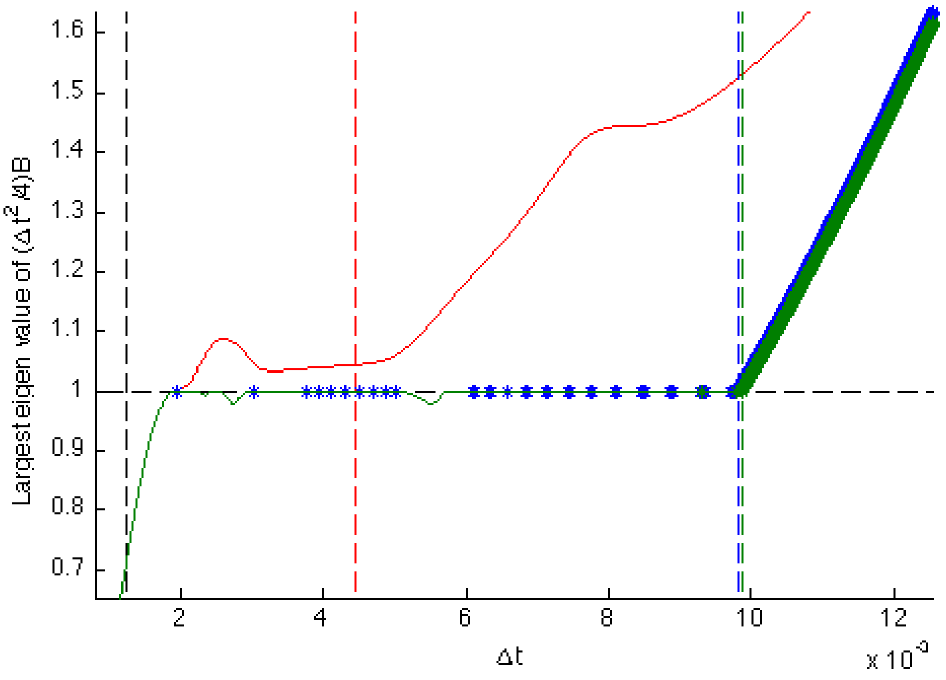


Figure 3.6: Evolution with or without halo of the largest eigenvalue of $\frac{\Delta t^2}{4}B$ according to Δt for a spatial refinement $p_s = 8$ (the legend is the same as for Figure 3.3).

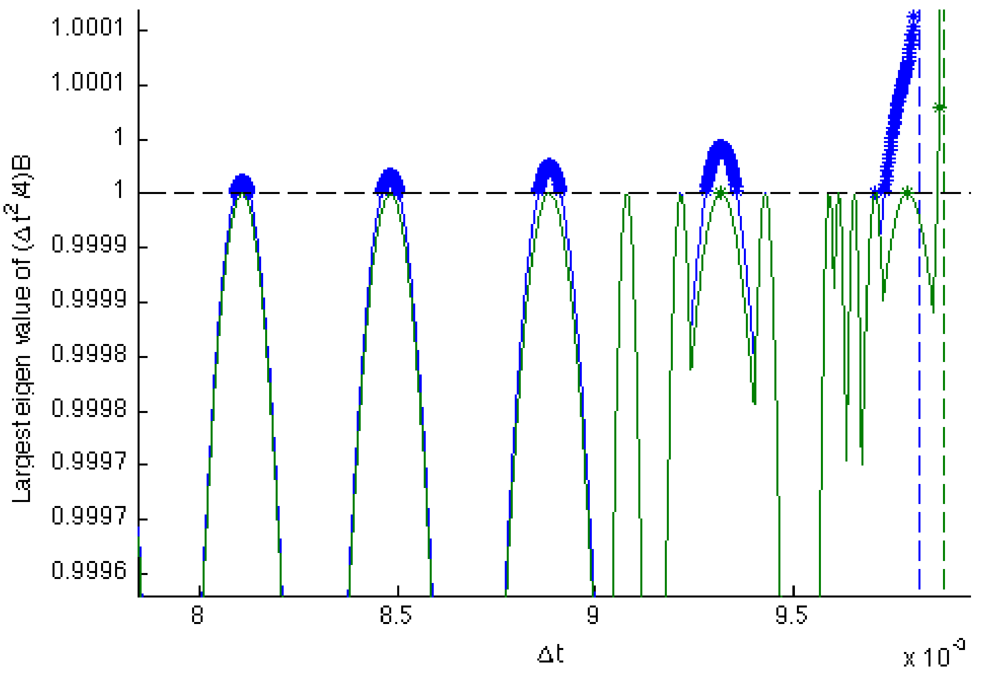


Figure 3.7: Zoom on Figure 3.6.

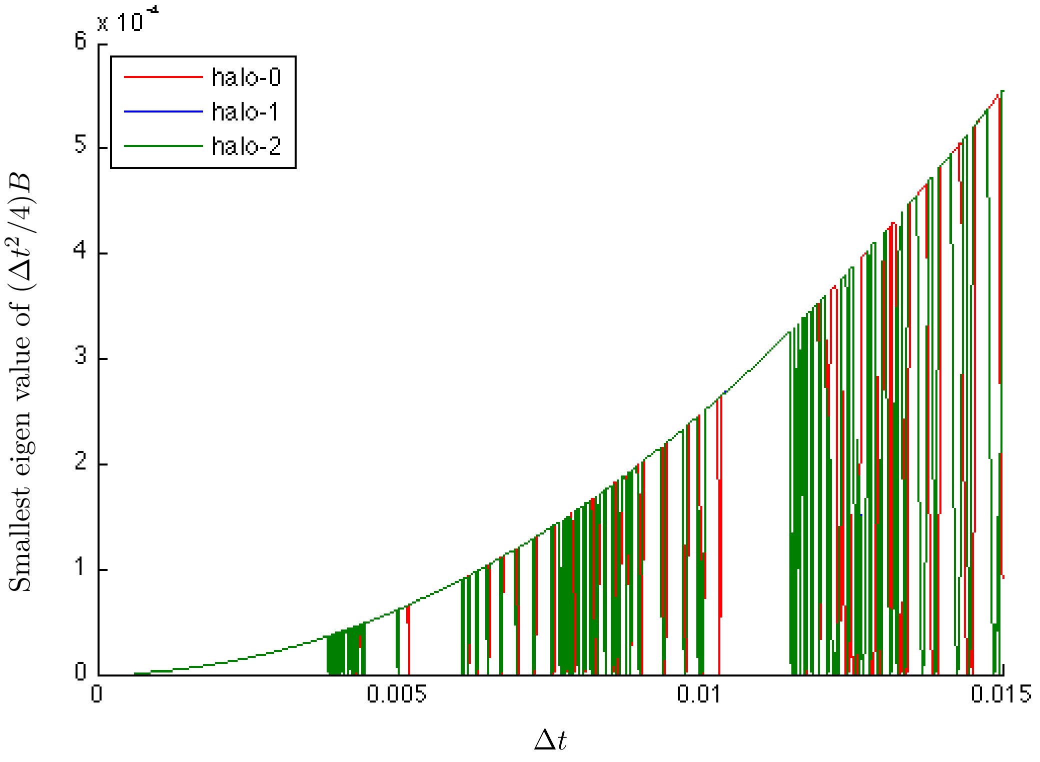


Figure 3.8: Evolution with or without halo of the smallest eigenvalue of $\frac{\Delta t^2}{4}B$ according to Δt for a spatial refinement $p_s = 8$.

This study of the $\tilde{z}$-exact scheme revealed the necessity to have a halo of coarse elements, otherwise the stability condition is almost as bad as without the $\tilde{z}$-exact scheme. However, as soon as the depth of the halo is superior to 1 we have $\Delta t_{opt} \simeq \Delta t_{max}$. We observed that the depth of halo needed to have the optimal time step is not dependent of spatial refinement.

3.2.2 Diaz-Grote's local time stepping algorithm

J. Diaz and M. Grote observed the same behavior on the scheme they proposed, introduced short-after, which is one of the reason why we wanted to introduce this intermediary scheme. This shows that this behavior is due to the choice of the approximation of $u''(t)$, and in no cases of the discretization of the solution of (3.5) involved in the their formulation.

The scheme (3.13) using the $\tilde{z}$-exact operator B is mathematically appealing, however in most situation computing B would be really expensive since it arises from the exact solution of (3.5). The strategy proposed by J. Diaz and M. Grote is to approximate B in a leap-frog manner. Solving (3.5) with a leap frog scheme leads to the local time stepping algorithm proposed in [**?**]. We shall emphasize again, that it is only when approximating the scheme (3.13) that a local time stepping appears, and until now there was no notion of local time step.

3.2.2.1 From the $\tilde{z}$-exact to Diaz-Grote's scheme: the local time stepping algorithm

In order to get a local time-stepping algorithm, we have to use a relation between u and $\tilde{z}$ since we will not use the exact solution described previously. By construction $\tilde{z}$ verifies

$$\tilde{z}(\Delta t) - 2\tilde{z}(0) + \tilde{z}(-\Delta t) = \Delta t^2 \int_{-1}^{1} (1 - |\theta|)\tilde{z}''(\theta \Delta t) d\theta,$$

and since u verifies (3.12) we have

$$u_{n+1} - 2u_n + u_{n-1} = \tilde{z}(\Delta t) - 2\tilde{z}(0) + \tilde{z}(-\Delta t).$$

The first step is thus to approximate $\tilde{z}(\Delta t)$ and $\tilde{z}(-\Delta t)$ where, we recall, $\tilde{z}$ solves the following differential problem:

$$\begin{cases} \frac{\mathrm{d}^2 \tilde{z}}{\mathrm{d}\tau^2}(\tau) = -A(I-P)u(t) - AP\tilde{z}(\tau), \\ \tilde{z}(0) = u(t), \quad \frac{\mathrm{d}\tilde{z}}{\mathrm{d}\tau}(0) = \nu. \end{cases} \tag{3.23}$$

It is important to note that we need to solve the equation forward and backward in time to get $\tilde{z}(\Delta t)$ and $\tilde{z}(-\Delta t)$.

Contrary to the $\tilde{z}$-exact scheme (3.13) where the parameter ν was of absolutely no use, for the Diaz-Grote's scheme we need to choose a value for the initial condition ν since we are going to use this value to initialize the algorithm. It is tempting to choose $\nu = \frac{\mathrm{d}u}{\mathrm{d}t}(t)$ since $\tilde{z}(\tau)$ becomes a second order approximate of $u(t+\tau)$. However, for the approximations of $\frac{\mathrm{d}u}{\mathrm{d}t}(t)$ (we have tried $\frac{u_{n+1}-u_{n-1}}{2\Delta t}$ or $\frac{u_{n+1}-u_n}{\Delta t}$) the local-time stepping scheme had bad numerical properties (dissipation, bad CFL condition), moreover having $\tilde{z}(\tau) = u(t+\tau)$ is only interesting for clarity reasons.

On the other hand, taking $\nu = 0$ leads to a very convenient algorithm since we then have $\tilde{z}(\tau) = \tilde{z}(-\tau)$, thus we do not need any more to solve (3.23) both forward and backward. Noting that $\tilde{z}(0) = u(t)$, the temporal scheme becomes

$$u(t+\Delta t) + u(t-\Delta t) \simeq 2\tilde{z}(\Delta t). \tag{3.24}$$

The previous relation leads to the discrete relation

$$\mathbf{u}_{n+1} + \mathbf{u}_{n-1} \simeq 2\tilde{\mathbf{z}}_{p/p},$$

where $\tilde{\mathbf{z}}_{m/p}$ is an approximation of $\tilde{z}(\frac{m}{p}\Delta t)$ achieved by solving (3.23) with a leap-frog scheme at the *fine time step* $\dfrac{\Delta t}{p}$.

Hence, we get the following algorithm

Algorithm 3.1 Diaz-Grote's local time stepping algorithm.

```
1: Set $\mathbf{w} = A(I-P)\mathbf{u}_n$ and $\tilde{\mathbf{z}}_0 = \mathbf{u}_n$
2: $\tilde{\mathbf{z}}_{1/p} = \tilde{\mathbf{z}}_0 - \frac{1}{2}\left(\frac{\Delta t}{p}\right)^2 (AP\tilde{\mathbf{z}}_0 + \mathbf{w})$
3: For $m = 1, .., p-1$, compute
       $\tilde{\mathbf{z}}_{(m+1)/p} = 2\tilde{\mathbf{z}}_{m/p} + \tilde{\mathbf{z}}_{(m-1)/p} - \left(\frac{\Delta t}{p}\right)^2 \left(AP\tilde{\mathbf{z}}_{m/p} + \mathbf{w}\right)$
4: Compute $\mathbf{u}_{n+1} = -\mathbf{u}_{n-1} + 2\tilde{\mathbf{z}}_{p/p}$
```

Solving the differential problem (3.23) in order to get $\tilde{z}(\Delta t)$ corresponds to the steps 1–3 in the Algorithm 3.1. As we can see on step 3, the differential equation (3.23) is solved with a leap-frog scheme and a time step $\Delta\tau = \frac{\Delta t}{p}$. The step 2 is also a leap-frog scheme, slightly hidden. The standard way to write the step 2 in a leap-frog manner is

$$\tilde{\mathbf{z}}_{1/p} = 2\tilde{\mathbf{z}}_0 - \tilde{\mathbf{z}}_{-1/p} - \left(\frac{\Delta t}{p}\right)^2 (AP\tilde{\mathbf{z}}_0 + \mathbf{w}),$$

but we know that $\tilde{\mathbf{z}}_{-1/p} = \tilde{\mathbf{z}}_{1/p}$. Hence, we get

$$\tilde{\mathbf{z}}_{1/p} = \tilde{\mathbf{z}}_0 - \frac{1}{2}\left(\frac{\Delta t}{p}\right)^2 (AP\tilde{\mathbf{z}}_0 + \mathbf{w}).$$

We can also see this first step as a second order Taylor decomposition with $\tilde{\mathbf{z}}_0' = 0$. Finally, we just have to apply the formula (3.24) to finish the coarse time step which corresponds to the step 4.

3.2.2.2 Properties of the local time stepping algorithm

So far, we have introduced an algorithm that evolves locally with a smaller time step. However, the whole point of a local time-stepping method is that the numerical properties outside the refined area are unchanged, and the numerical properties within the refined area are as close as possible to those of a global small time step. The properties we want are that this algorithm be second order accurate in time, and that the CFL conditions inside and outside the refined area behave as if the two parts were numerically independent.

We refer to [**?**] for a proof of the following properties.

Property 3.3
The local time-stepping Algorithm 3.1 is equivalent to

$$u_{n+1} = 2u_n - u_{n-1} - \Delta t^2 A_p u_n,$$

where A_p is defined by

$$A_p = A - \frac{2}{p^2}\sum_{j=1}^{p-1}\left(\frac{\Delta t}{p}\right)^{2j} \alpha_j^p (AP)^j A,$$

where the constants α_j^m are given by

$$\begin{cases} \alpha_1^2 = \frac{1}{2}, \quad \alpha_1^3 = 3, \quad \alpha_2^3 = -\frac{1}{2}, \\ \alpha_1^{m+1} = \frac{m^2}{2} + 2\alpha_1^m - \alpha_1^{m-1}, \\ \alpha_j^{m+1} = 2\alpha_j^m - \alpha_j^{m-1} - \alpha_{j-1}^m, \quad j = 2, .., m-2, \\ \alpha_{m-1}^{m+1} = 2\alpha_{m-1}^m - \alpha_{m-2}^m, \\ \alpha_m^{m+1} = -\alpha_{m-1}^m. \end{cases} \tag{3.25}$$

This scheme is second order accurate in time. Furthermore, the matrix A_p is symmetric if A is symmetric, consequently A_p has real eigenvalues.

From the previous property, we can directly deduce that the local time-stepping algorithm conserves the same discrete energy as the leap-frog scheme as stated by the following property.

Property 3.4
The second-order local time-stepping scheme conserves the discrete energy

$$E^{n+\frac{1}{2}} = \frac{1}{2}\left(\left\langle\left(I - \frac{\Delta t^2}{4}A_p\right)\frac{u_{n+1}-u_n}{\Delta t}, \frac{u_{n+1}-u_n}{\Delta t}\right\rangle + \left\langle A_p\frac{u_{n+1}+u_n}{2}, \frac{u_{n+1}+u_n}{2}\right\rangle\right).$$

Hence, if λ_{min} and λ_{max} denote the smallest and largest eigenvalues of A_p, the numerical scheme will be stable if and only if

$$0 \leq \frac{\Delta t^2}{4}\lambda_{min} \leq \frac{\Delta t^2}{4}\lambda_{max} \leq 1.$$

Remark 3.7. *When we tried to write a local time stepping with the initial condition $\nu = \dfrac{du}{dt}(t)$ for the problem (3.23), we did not manage to write the global scheme in a leap-frog manner as previously with a matrix A_p, this might explain why we had such bad properties and why Diaz-Grote's scheme has interesting properties, especially the conservation of a discrete energy.*

3.2.2.3 Comparing the $\tilde{z}$-exact and Diaz-Grote's formulation

In this section, we want to compare the spectral behavior of the $\tilde{z}$-exact operator B and its approximation A_p.

Property 3.5
A_p converges to B when p tends to infinity.

We illustrate this property on Figure 3.9 where we see how the maximum eigenvalue of A_p converges to the one of B.

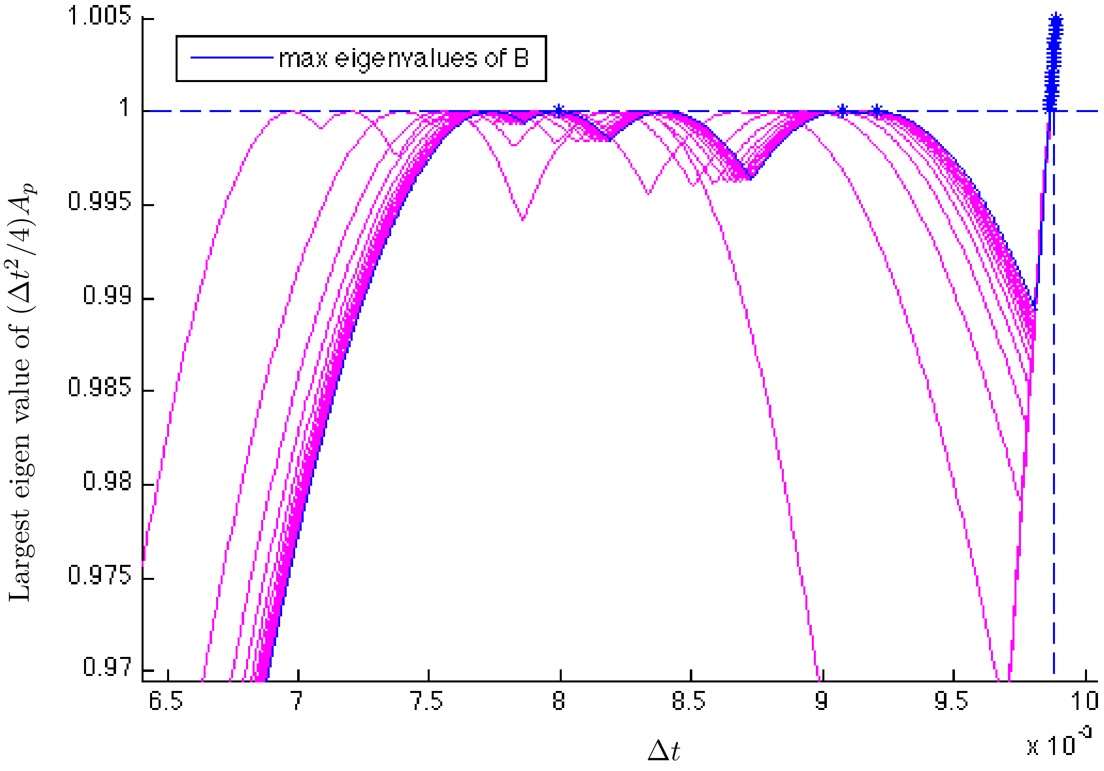


Figure 3.9: Convergence of the maximum eigenvalue of A_p to the one of B.

Proof. I would like to thank Anne Cassier for her help on this proof. We want to show that $\lim_{p\to\infty} A_p = B$. Since we know the power series of A_p and B, this is equivalent to show that

$$\forall j \in \mathbb{N}, \quad \lim_{p\to\infty} \frac{\alpha_j^p}{p^{2(j+1)}} = \frac{(-1)^{j+1}}{(2(j+1))!}, \tag{3.26}$$

where the α_j^p are the constants defined in (3.25).
We proceed by recurrence on $j \in \mathbb{N}$.

Case $j = 1$: We begin by summing the relation $\alpha_1^{k+1} = \frac{k^2}{2} + 2\alpha_1^k - \alpha_1^{k-1}$ for $k = 2, .., m-1$

$$\begin{aligned}
&\sum_{k=2}^{m-1} \alpha_1^{k+1} = \sum_{k=2}^{m-1} \frac{k^2}{2} + 2\sum_{k=2}^{m-1} \alpha_1^k - \sum_{k=2}^{m-1} \alpha_1^{k-1} \\
\Longrightarrow\ &\sum_{k=3}^{m} \alpha_1^k = \sum_{k=2}^{m-1} \frac{k^2}{2} + 2\sum_{k=2}^{m-1} \alpha_1^k - \sum_{k=1}^{m-2} \alpha_1^k \\
\Longrightarrow\ &\alpha_1^m = \sum_{k=2}^{m-1} \frac{k^2}{2} + \alpha_1^2 + \alpha_1^{m-1} + \alpha_1^1 \\
\Longrightarrow\ &\sum_{m=2}^{p} \alpha_1^m = \sum_{m=2}^{p} \sum_{k=2}^{m-1} \frac{k^2}{2} + (p-1)(\alpha_1^2 - \alpha_1^1) + \sum_{m=1}^{p-1} \alpha_1^m
\end{aligned}$$

$$\begin{aligned}
\Longrightarrow \quad \alpha_1^p &= \sum_{m=2}^{p} \sum_{k=2}^{m-1} \frac{k^2}{2} + (p-1)(\alpha_1^2 - \alpha_1^1) + \alpha_1^1 \\
&= \sum_{m=2}^{p} \frac{1}{2}(\frac{(m-1)m(2m-1)}{6} - 1) - \frac{7}{2}(p-1) + 3 \\
&= \sum_{m=2}^{p} \frac{1}{2}\frac{2m^3 - 3m^2 + m - 6}{6} - \frac{7}{2}(p-1) + 3.
\end{aligned}$$

For simplicity, we take an equivalent of α_1^p

$$\begin{aligned}
\alpha_1^p &\sim \sum_{m=2}^{p} \frac{m^3}{6} \\
&\sim \frac{p^4}{24}.
\end{aligned}$$

Finally, we get

$$\lim_{p\to\infty} \frac{\alpha_1^p}{p^4} = \frac{1}{24}.$$

Thus, the result (3.26) is true for the rank 1.

Case $j > 1$: Let $j \in \mathbb{N}$. We assume that the result (3.26) is true for the rank j, and we want to prove it for the rank $j+1$.
$\forall m \in \mathbb{N}$, we define,

$$\begin{cases} v_j^m = \alpha_j^m - \alpha_j^{m-1}, \\ w_j^m = v_j^m - v_j^{m-1}. \end{cases}$$

Then,

$$\begin{aligned}
&\alpha_{j+1}^{m+1} = 2\alpha_{j+1}^m - \alpha_{j+1}^{m-1} - \alpha_j^m, \\
\Longrightarrow\ &\alpha_{j+1}^{m+1} - \alpha_{j+1}^m = \alpha_{j+1}^m - \alpha_{j+1}^{m-1} - \alpha_j^m, \\
\Longrightarrow\ &v_{j+1}^{m+1} = v_{j+1}^m - \alpha_j^m, \\
\Longrightarrow\ &v_{j+1}^{m+1} - v_{j+1}^m = -\alpha_j^m, \\
\Longrightarrow\ &w_{j+1}^{m+1} = -\alpha_j^m,
\end{aligned}$$

$$\boxed{\Longrightarrow \quad w_{j+1}^{m+1} \underset{m\to\infty}{=} \frac{(-1)^{j+2}}{(2(j+1))!} m^{2(j+1)} + o(m^{2(j+1)}).}$$

Furthermore,

$$\begin{aligned}
v_{j+1}^m &= \sum_{k=1}^{m} w_{j+1}^k + v_{j+1}^0, \\
&\underset{m\to\infty}{=} \frac{(-1)^{j+2}}{2(j+1)!} \sum_{k=1}^{m} m^{2(j+1)} + v_{j+1}^0 + \sum_{k=1}^{m} o(m^{2(j+1)}), \\
&\underset{m\to\infty}{=} \frac{(-1)^{j+2}}{2(j+1)!} \frac{m^{2(j+1)+1}}{2(j+1)+1} + v_{j+1}^0 + o(m^{2(j+1)+1}), \quad \text{since} \begin{cases} \displaystyle\sum_{k=1}^{m} m^p = \frac{m^{p+1}}{p+1} + o(m^{p+1}), \\ \text{and} \\ \displaystyle\sum_{k=0}^{m} o(m^p) = o(m^{p+1}). \end{cases} \\
&\underset{m\to\infty}{=} \frac{(-1)^{j+2}}{(2j+3)!} m^{2j+3} + o(m^{2j+3}).
\end{aligned}$$

Finally, we get

$$\begin{aligned}
\alpha_{j+1}^m &\underset{m\to\infty}{=} \sum_{k=0}^{m} v_j^m + \alpha_{j+1}^0,\\
&\underset{m\to\infty}{=} \alpha_{j+1}^0 + \frac{(-1)^{j+2}}{(2j+3)!}\sum_{k=1}^{m} k^{2j+3} + \sum_{k=1}^{m} o(k^{2j+3}),\\
&\underset{m\to\infty}{=} \alpha_{j+1}^0 + \frac{(-1)^{j+2}}{(2j+3)!}\frac{m^{2j+3}}{2j+4} + o(m^{2j+4}),
\end{aligned}$$

$$\boxed{\alpha_{j+1}^m \underset{m\to\infty}{=} \frac{(-1)^{j+2}}{(2(j+2))!} m^{2(j+2)} + o(m^{2(j+2)}).}$$

$\square$

From the previous property, we can define another approximation of B based on its Taylor representation. We define the matrices B_p as follows

$$\forall p > 2, \quad B_p = A + \sum_{j=1}^{p-1} \beta_j \Delta t^{2j} (AP)^j A,$$

where

$$\forall j > 2, \quad \beta_j = 2\frac{(-1)^{2j}}{(2(j+1))!}.$$

The idea behind this last scheme is to show that it is not straightforward to have a good approximation of the $\tilde{z}$-exact scheme as Diaz-Grote's scheme achieves. We still have a leap-frog relation of the form

$$u_{n+1} = 2u_n - u_{n-1} - \Delta t^2 B_p u_n,$$

however we do not have a local time stepping relation in the fine domain as for Diaz-Grote's scheme, which is one of the main advantage of Diaz-Grote's method since we do not have to compute explicitly the matrix A_p.

As we can see on Figures 3.10 and 3.11, the spectral behavior of the matrices B_p for a space refinement p_s of 2 are not as good as for the matrices A_p. Indeed, the minimal eigenvalue of B_2 becomes negative before the optimal time step Δt_{max} is reached contrary to A_2. The spectral behavior is not better for B_3 since it is now the maximum eigenvalue that becomes superior to 1 before the optimal time step. We have to go up to B_4 to get a approximate that has the desired spectral behavior.

Besides, we shall note that the instability of B_p comes from the maximal eigenvalue for odd p and from the minimal eigenvalue for even p. This is a really troublesome property since we saw that the maximum eigenvalues of B are already just below 1. Moreover, there is no correlation between the value of p and the value of p_s for B_p contrary to A_p. This shows one of the major advantage of Diaz-Grote's approximation, which we unfortunately cannot prove but only observe, the approximation A_p really acts as if the relation between p and p_s is of the same nature as a CFL condition. In particular, it means that if we refine in space by p_s it is sufficient to refine in time with $p = p_s$.

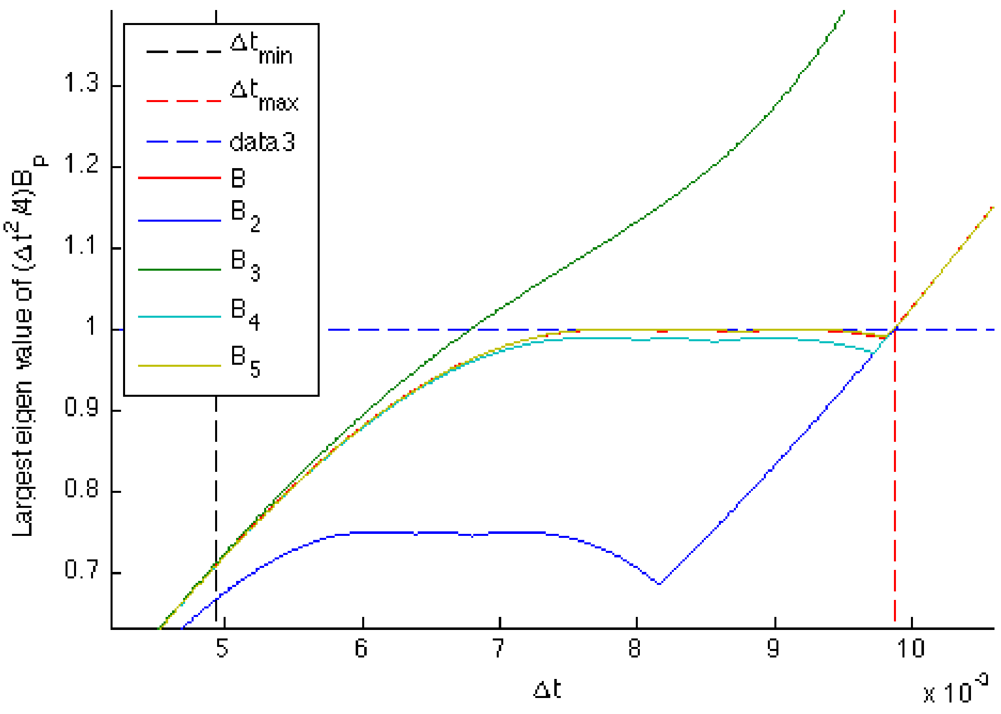


Figure 3.10: Maximum eigenvalues of B_p for $p = 2, 3, 4, 5$ compared to those of B.

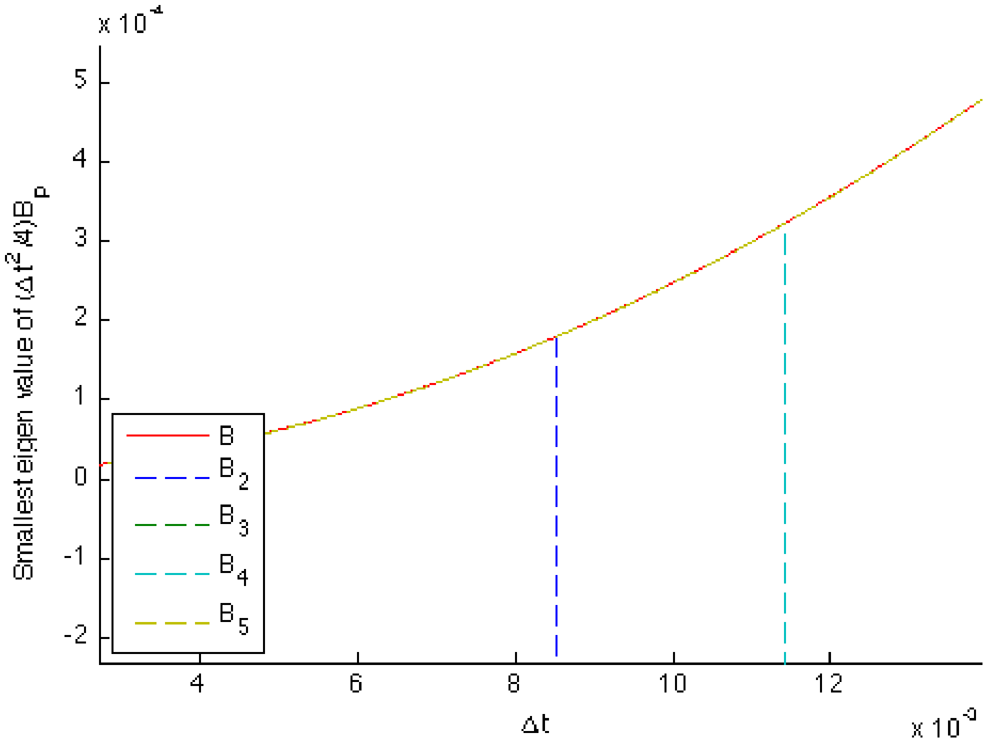
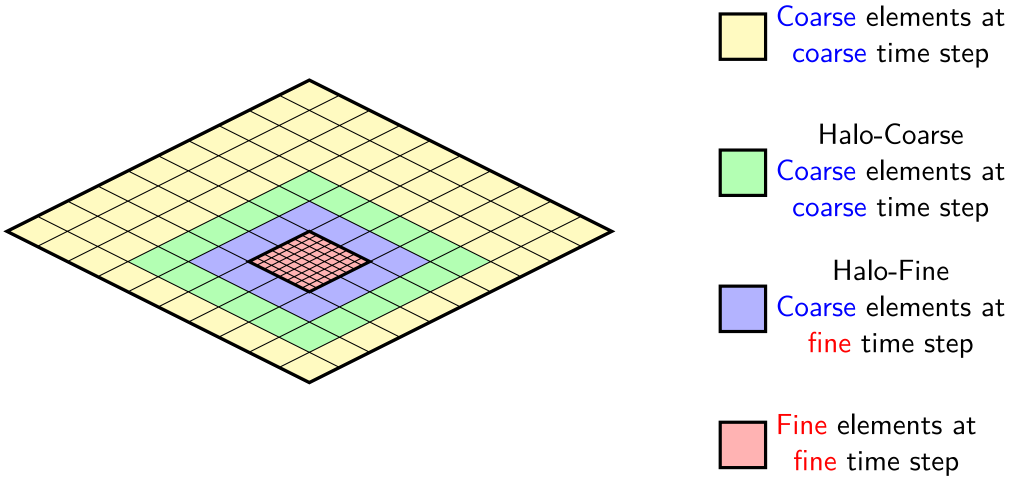


Figure 3.11: Minimum eigenvalues of B_p for $p = 2, 3, 4, 5$ compared to those of B.

3.2.2.4 Introduction to the halo

Another important aspect of this scheme, already mentioned for the $\tilde{z}$-exact scheme, is the idea to *overlap* spatially coarse cells and temporally fines cells. This means that not only spatially fine cells are at fine time steps, but also a certain amount of the surrounding spatially coarse cells. The number of coarse elements overlapped within the fine time step is an important parameter, since it has a significant impact on the stability of the method and thus the global CFL condition. Not overlapping has a major impact on the coarse CFL condition, which is precisely what we do not want of a local time stepping algorithm. On the contrary, the more coarse elements are overlapped with the fine time step, the more stable the method is. However, it is not necessary to have too many coarse elements at fine time step to stabilize the method.
In particular, an overlap by one element for 2D cases (as shown on Figure 3.12) is enough to recover normal CFL conditions.
In [**?**] J. Diaz and M. Grote performed a detailed study of this parameter on their local time stepping scheme. However, as we saw this stability condition already arises with the $\tilde{z}$-exact scheme.



🟨 Coarse elements at coarse time step
🟩 Halo-Coarse Coarse elements at coarse time step
🟦 Halo-Fine Coarse elements at fine time step
🟥 Fine elements at fine time step

Figure 3.12: Representation of different types of elements.

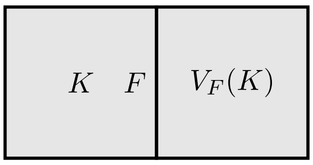
We call halo of a refined area, the set of spatially coarse elements which are affected by AP. We call halo-coarse the set of coarse elements affected by AP not included in the fine part through P, and halo-fine the set of coarse elements included in the fine part through P. (See Figure 3.12 for a graphical illustration).

Remark 3.8. *This distinction between halo-coarse and halo-fine is really important to understand how the fluxes between these elements are exchanged as we shall see in the local formulation of local time stepping scheme.*

3.2.2.5 Local formulation of the local time stepping algorithm

Using the global formulation with the matrix A and the projection matrix P describes well the local time-stepping algorithm but hides the locality of the fine time step, also making it impossible to use the local time-stepping algorithm as it is in an effective computer implementation. Thus, we rewrite here a local version of this algorithm for discontinuous Galerkin methods.

We denote by $\mathcal{F}_K^f$ and $\mathcal{F}_K^c$ the sets of faces of the element K shared with an element at fine and coarse time step respectively. We also denote by $V_F(K)$ the neighbor element of K on a face F.



To have a local description of the algorithm we need to introduce three different algorithms, each corresponding to a specific area, *i.e.* halo, coarse and fine areas (as shown on Figure 3.12). We begin with the algorithm for the elements inside the halo as it is the closest to the Algorithm 3.1.

Algorithm 3.2 Halo element algorithm.

```
1: Set $\mathbf{w}^K = \sum_{F\in\mathcal{F}_K^c} F_F \mathbf{u}_n^{V_F(K)}$ and $\tilde{\mathbf{z}}_0^K = \mathbf{u}_n^K$
2: $\tilde{\mathbf{z}}_{1/p}^K = \tilde{\mathbf{z}}_0^K - \frac{1}{2}\left(\frac{\Delta t}{p}\right)^2 \left(K_K \tilde{\mathbf{z}}_0^K + \mathbf{w}^K + \sum_{F\in\mathcal{F}_K^f} F_F \tilde{\mathbf{z}}_0^{V_F(K)}\right)$
3: For $m = 1, .., p-1$, compute
       $\tilde{\mathbf{z}}_{(m+1)/p}^K = 2\tilde{\mathbf{z}}_{m/p}^K + \tilde{\mathbf{z}}_{(m-1)/p}^K - \left(\frac{\Delta t}{p}\right)^2 \left(K_K \tilde{\mathbf{z}}_{m/p}^K + \mathbf{w}^K + \sum_{F\in\mathcal{F}_K^f} F_F \tilde{\mathbf{z}}_{m/p}^{V_F(K)}\right)$
4: Compute $\mathbf{u}_{n+1}^K = -\mathbf{u}_{n-1}^K + 2\tilde{\mathbf{z}}_{p/p}^K$
```

Remark 3.9. *One might think that halo elements at fine and coarse time steps are performing exactly the same algorithm, whereas fluxes between halo-coarse elements update the vector* $\mathbf{w}^K$ *and thus are computed at every coarse time step while fluxes between halo-fine elements are computed at every fine time step (see Figure 3.12).*

From this halo element algorithm, it is easy to deduce the algorithm for elements both fine in space and time since $\mathbf{w}^K = 0$:

Algorithm 3.3 Fine element algorithm.

```
1: Set    $\tilde{\mathbf{z}}_0^K = \mathbf{u}_n^K$
2: $\tilde{\mathbf{z}}_{1/p}^K = \tilde{\mathbf{z}}_0^K - \frac{1}{2}\left(\frac{\Delta t}{p}\right)^2 \left(K_K \tilde{\mathbf{z}}_0^K + \sum_{F\in\mathcal{F}_K} F_F \tilde{\mathbf{z}}_0^{V_F(K)}\right)$
3: For $m = 1, .., p-1$, compute
       $\tilde{\mathbf{z}}_{(m+1)/p}^K = 2\tilde{\mathbf{z}}_{m/p}^K + \tilde{\mathbf{z}}_{(m-1)/p}^K - \left(\frac{\Delta t}{p}\right)^2 \left(K_K \tilde{\mathbf{z}}_{m/p}^K + \sum_{F\in\mathcal{F}_K} F_F \tilde{\mathbf{z}}_{m/p}^{V_F(K)}\right)$
4: Compute $\mathbf{u}_{n+1}^K = -\mathbf{u}_{n-1}^K + 2\tilde{\mathbf{z}}_{p/p}^K$
```

As we mention earlier, the algorithm for elements both coarse in space and time is a leap-frog algorithm:

Algorithm 3.4 Coarse element algorithm.

```
Compute $\mathbf{u}_{n+1}^K = 2\mathbf{u}_n^K - \mathbf{u}_{n-1}^K - \Delta t^2 \left(K_K \mathbf{u}_n^K + \sum_{F\in\mathcal{F}_K} F_F \mathbf{u}_n^{V_F(K)}\right)$
```

Remark 3.10. *Having all these different algorithms working at different time steps, sending their fluxes to different vectors ($\mathbf{w}^K$ or $\mathbf{u}^K$) reveals some of the difficulties to implement the local time-stepping method without projection matrix.*

3.3 Considerations on the cost of local space-time mesh refinement

Using local space-time refinements where needed is a really appealing feature, however we still have to be careful on the fast increase of the overall computation cost of the refined areas. In two dimensions, any cell refined by a factor p has at least its cost multiplied by a factor of p^3, and in three dimensions by p^4. We propose here an analysis to show how quickly the computation cost grows, even for fairly small refined area.
Let p_t denotes the level of temporal refinement with $\Delta t_f = \dfrac{\Delta t_c}{p_t}$, where Δt_c and Δt_f are the coarse and fine time steps, respectively. Let p_s denotes the level of spatial refinement with $h_f = \dfrac{h_c}{p_s}$, where h_c and h_f are the coarse and fine space steps, respectively. We talk about a refinement by p when $p_t = p_s = p$.

Mesh refinement is usually used to describe complex heterogeneities and singularities of the solution. Therefore, the number of point per wavelength is no longer a relevant criterion to mesh the medium. What matters is the number of elements to represent the heterogeneity or the singularity due to the loss of regularity. For this reason, we can use another interesting feature of discontinuous Galerkin methods, called *p-adaptivity* and reduce the order of our polynomial basis inside the refined area, significantly saving computation and memory without reducing the global accuracy significantly. Furthermore, the CFL condition is different for each polynomial approximation. This can be exploited to use a smaller temporal refinement than the spatial refinement according to the CFL conditions, *i.e* $p_t < p_s$. Note that this is only possible because we both have non-conforming refinement and p-adaptivity. Due to the CFL condition, which is slightly weakened by the non-conforming interface, we cannot strictly use $p_t = \dfrac{C_{cfl}(k_c)}{C_{cfl}(k_f)} p_s$ where k_c and k_f are the polynomial order of the basis in the coarse and fine parts respectively.

We display in Figures 3.13, 3.14, 3.15 and 3.16 the "normalized" computational cost we expect from using these different options for different situations. We mean by "normalized cost" that a cost of 1 is the total cost of a simulation without local refinement. We consider four different refining scenarios that are:

- dashed blue line: a mesh purely refined in space with a global time step limited by the smallest element, thus p_s times smaller,
- continuous blue line: a mesh using local time stepping we the ratio dictated by the spatial refinement ($p_t = p_s$),
- dashed red line: a mesh using a lower polynomial order in the refined area, but not taking account of the larger CFL condition to relax the time refinement,
- continuous red line: a mesh using a lower order in the refined area and using a time refinement according to the CFL condition in this area.

We begin first with some general remarks. If we consider that the interesting window to use local time-stepping is when the continuous blue line is significantly below the dashed blue line. Then, we note that without *p-adaptivity* the interesting window to use local time stepping is only for really small refined areas. We note that it is really important to lower the polynomial degree in the refined area so that the computational cost does not increase too quickly. We also note that if the same polynomial order is used in the coarse and the fine parts, then the local-time stepping is only interesting (in term of computational cost) for really small areas. This is due to the fine part that quickly caries all the computation cost.

We note on Figure 3.13 and 3.14, when using a polynomial basis $Q_f = Q_1$ instead of $Q_f = Q_5$ in the refined area, that the computation cost is reduced by factor of roughly 100 when the refined area reaches at least 1% of the total domain. When using the correct CFL condition inside the refined area, the computation cost is again reduced by a factor of 10. Those two optimizations lead to an overall cost reduction of 10^3 which is significant. When using a polynomial basis $Q_f = Q_1$ instead of $Q_f = Q_{10}$, the gain is even more considerable, as can be seen on Figure 3.16, with an overall cost reduced by a factor of roughly $10^{4.5}$.

We note on Figures 3.14, 3.15 and 3.16, refining by a factor $p_s = 100$ is really demanding. If we compare Figures 3.14 and 3.15, we note that using $Q_f = Q_1$ and $Q_f = Q_2$ already makes a huge difference in terms of computation cost. We note on Figure 3.16 that if we are using the coarse grid in a "spectral" discontinuous Galerkin manner (since $Q_c = Q_{10}$), then it becomes really interesting, if not mandatory, to use all computation cost optimizations, otherwise the cost becomes quickly prohibitive.

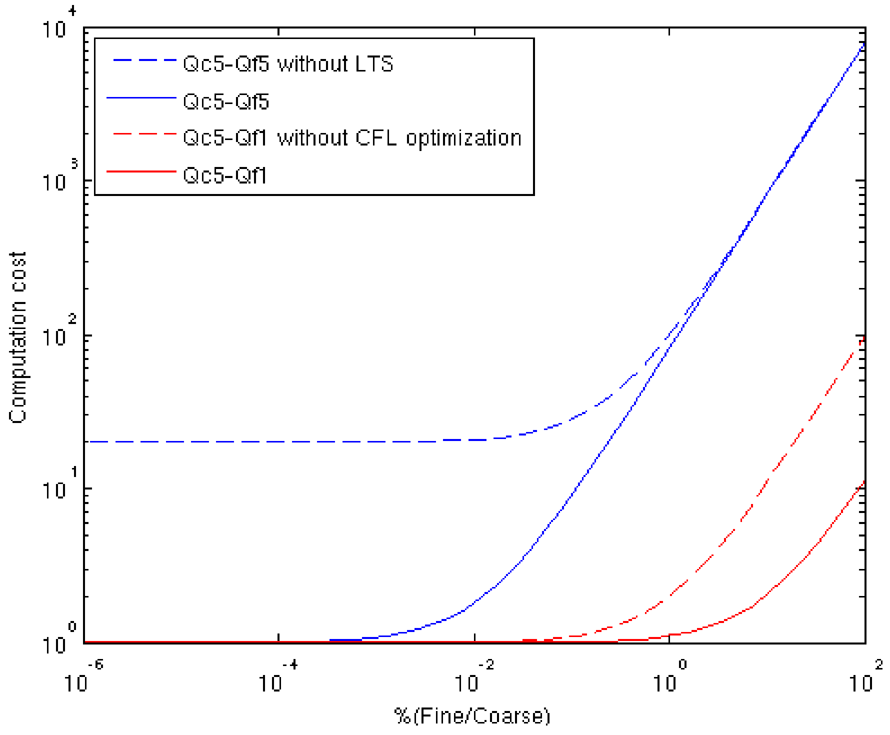


Figure 3.13: Computation cost of a $Q_c = Q_5$ and $Q_f = Q_1$ mesh refined by $p = 20$ according to the percentage of the volume refined.

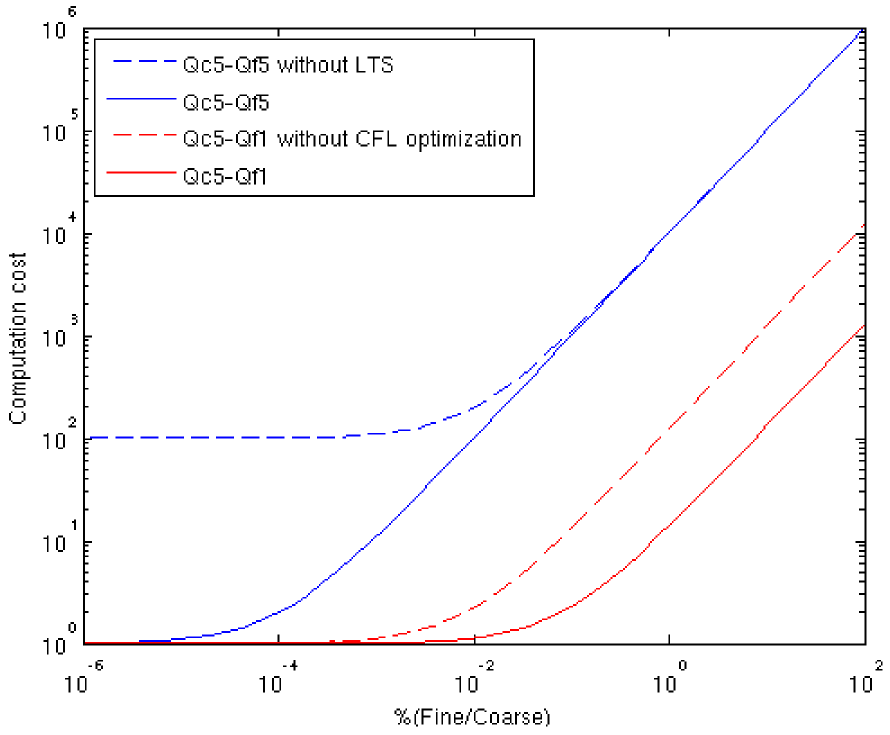


Figure 3.14: Computation cost of a $Q_c = Q_5$ and $Q_f = Q_1$ mesh refined by $p = 100$ according to the percentage of the volume refined.

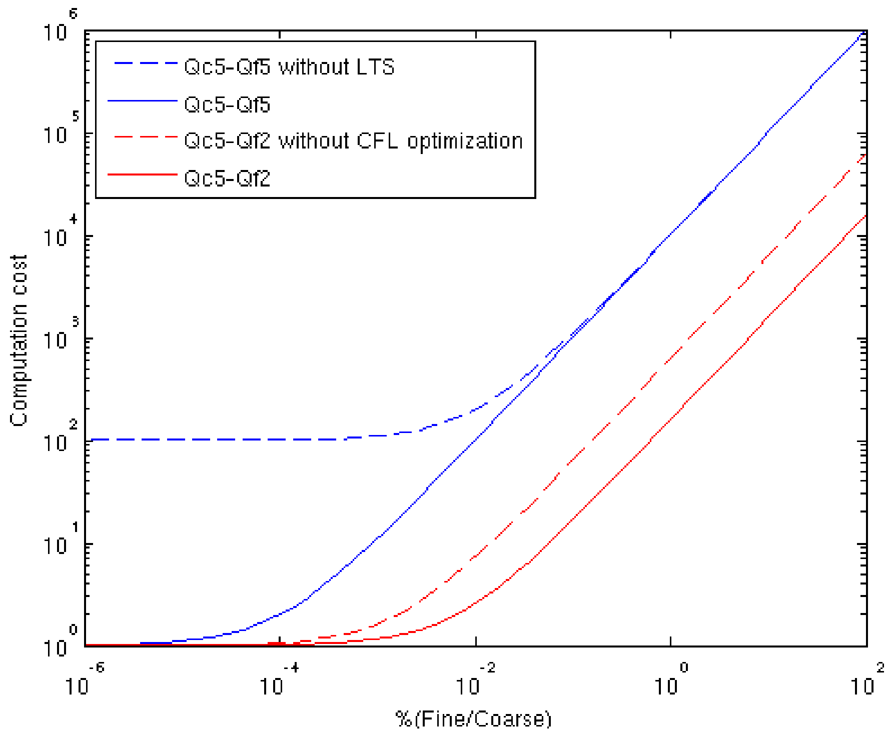


Figure 3.15: Computation cost of a $Q_c = Q_5$ and $Q_f = Q_2$ mesh refined by $p = 100$ according to the percentage of the volume refined.

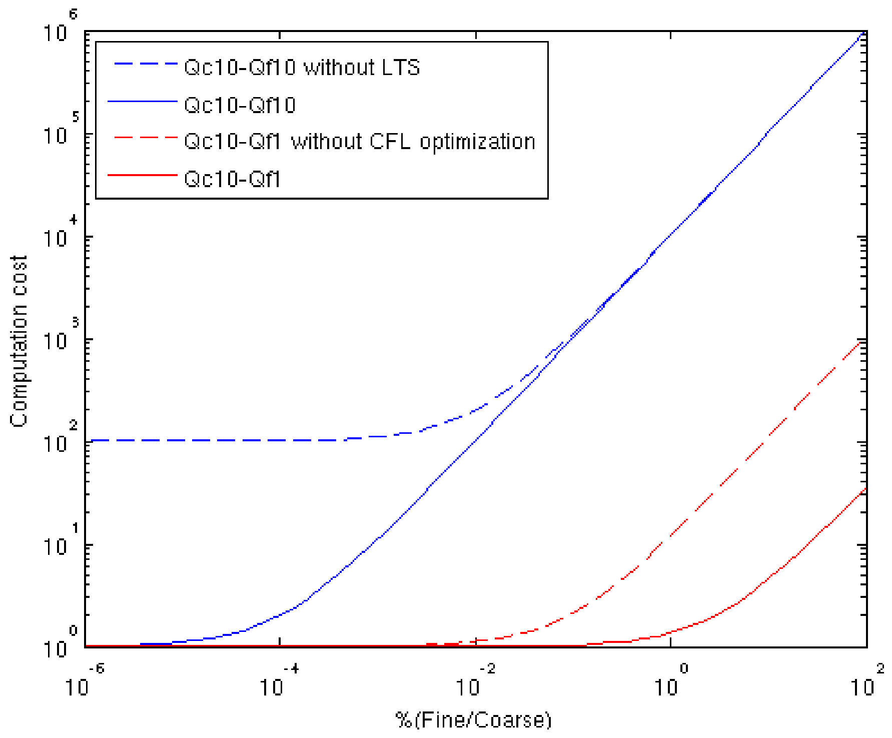


Figure 3.16: Computation cost of a $Q_c = Q_{10}$ and $Q_f = Q_1$ mesh refined by $p = 100$ according to the percentage of the volume refined.

3.4 Numerical experiments

In this section we want to see if the local time-stepping algorithm introduces any local artifact effect. Indeed, we know that this local-time stepping algorithm is second-order accurate in time, but this global information does not give much information about any local spurious effect that might be created by this algorithm. Having a method that converges does not tell us if the local time stepping method introduces spurious effects that could be misinterpreted, or even spoil any interpretation, at the desired accuracy. When using a local-time stepping algorithm, we want to release the constraints imposed by the small cells to the coarse cells. In other words, we want to keep the same mesh, we do not want to have to refine our mesh to reduce the amplitude of any spurious effect. We also do not want to reduce the global time step, as it is precisely the role of a local time stepping method.

In order to study the above mentioned possible effects we compare seismograms of refined simulations with seismograms of a reference solution. We decided to use a numerical solution without refinement as reference solution instead of an exact solution. In this manner we only look at the effect introduced by the local time-stepping algorithm and/or the local space refinement.

For these experiments, we consider an homogeneous medium with $\rho = 2100\ kg.m^{-3}$, $\lambda = 4.2 \times 10^9$ and $\mu = 2.1 \times 10^9$ ($v_s = 1000\, m.s^{-1}$ and $v_p = 2000\, m.s^{-1}$). The physical domain is of size $100m \times 200m$. We use an explosive source located at the point $\mathbf{x}_S$, that is

$$\mathbf{f}(\mathbf{x}, t) = h(t) g(|\mathbf{x} - \mathbf{x}_S|) \frac{\overrightarrow{\mathbf{x} - \mathbf{x}_S}}{|\mathbf{x} - \mathbf{x}_S|}$$

where $h(t)$ is a second order Ricker, with central frequency $f_0 = 40Hz$,

$$h(t) = (2\pi^2 (f_0 t - 1)^2 - 1) e^{-\pi^2 (f_0 t - 1)^2},$$

and $g(|\mathbf{x} - \mathbf{x}_S|)$ is a regularization of a Dirac by a Gaussian centered in $\mathbf{x}_S = (50m, 150m)$ and distributed over a disk of radius $r_0 = 8m$,

$$g(|\mathbf{x} - \mathbf{x}_S|) = \frac{e^{-7 \frac{|\mathbf{x} - \mathbf{x}_S|^2}{r_0^2}}}{r_0^2}.$$

The initial conditions are null. We used Q_3 elements of size $4m$ for these simulations, with Legendre-Gauss function bases (see Figure 1.3.3). We add PML around our physical computational domain.

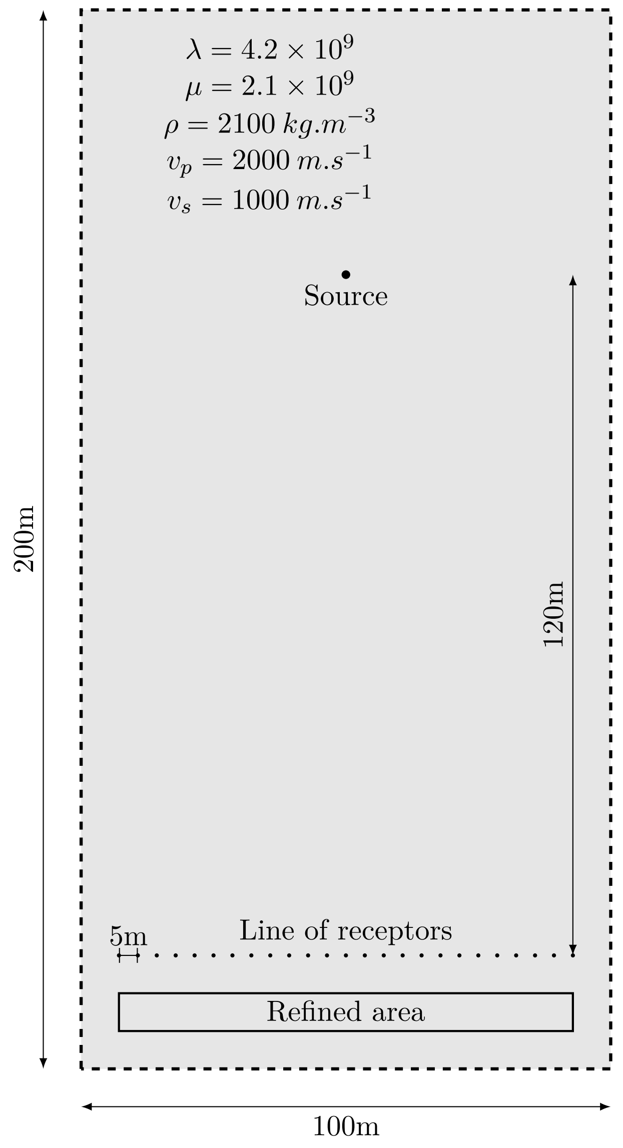


Figure 3.17: Homogeneous medium characteristics.

Our experimental protocol is to first analyze each refinement (spatial and temporal) separately to see what artifact effects they introduce, and then to analyze how they couple together.
Instead of comparing directly our refined seismograms with the reference solution u_{ref} (see Figure 3.18), we calculate the difference with the reference solution u_{ref} to highlight only the spurious effects. It is therefore very important to pay attention to the amplitudes, the absolute amplitude is indicated in each figure by $Amp = ||u_h||_\infty$, and we also give a relative amplitude called $Amp\ error = \frac{||u_{ref}-u_h||_\infty}{||u_{ref}||_\infty}$ which is the ratio of the maximum amplitude of our refined solution u_h divided by the maximum amplitude of our reference solution u_{ref}. We also referred the *offset* to give an idea of the localization of each receptor.
For each experiment the refinement, whether spatial, temporal or both, is applied in the box $[16m, 16m] \times [80m, 20m]$ in the bottom of the domain, as shown on Figure 3.17.

3.4.1 Analysis of time refinement

In this first experiment, we want to illustrate if the local time stepping algorithm introduces spurious effects on a purely temporally refined mesh, thus we have a coarse grid everywhere. We take a set of temporal refinement $p_t = 5, 10, 20, 100$ to see if this introduces any differences.
We note on Figure 3.19 that the spurious effects are of really small amplitude. Moreover, the amplitude of this effect does not seem to depend on the level of time refinement, which is unexpected, the spurious reflections might be only due to the change of temporal scheme. We also note that these effects are of the same amplitude, if not lower, as the amplitude of dispersion in this case.

3.4.2 Analysis of space refinement

In this experiment, we want to investigate the effect of non-conforming spatial refinement without local time-stepping. We take a set of spatial refinement $p_s = 5, 10, 20$, we decided to take a time step adapted to each simulation, thus if Δt_0 is the time step of the reference solution, for each refined simulation we took $\Delta t_{p_s} = \dfrac{\Delta t_0}{p_s}$.
We note on Figure 3.20 that the spurious effects are of small amplitude, though larger than those produced by local time-stepping. We also note that the amplitude and the shape of these spurious effects do not depend much on the level of refinement.

3.4.3 Analysis of coupled space and time refinement

In this experiment, we want to investigate how coupled space and time refinement behave. In that respect, we consider a set of spatio-temporal refinements $p = 2, 5, 10, 20$.
We note on Figure 3.21 that refining both in space and time reduces the spurious effects. The spurious effects are now of the amplitude of the temporal refinement and no more of the amplitude of the spatial refinement. We observe the same behavior as for purely temporal and spatial refinement, the spurious effects depending weakly on the level of refinement.
We note on Figure 3.22 that the behavior is different if we use different polynomial orders for the spatially coarse and fine elements, however the differences are marginal for Q_2 or Q_3 fine elements, and are still small for Q_1 fine elements. On Figure 3.23, we have Q_5 coarse elements, we note that Q_1 fine elements produce much higher artifact effects. We observe on Figure 3.24 that the differences for Q_2 to Q_4 fine elements are marginal.

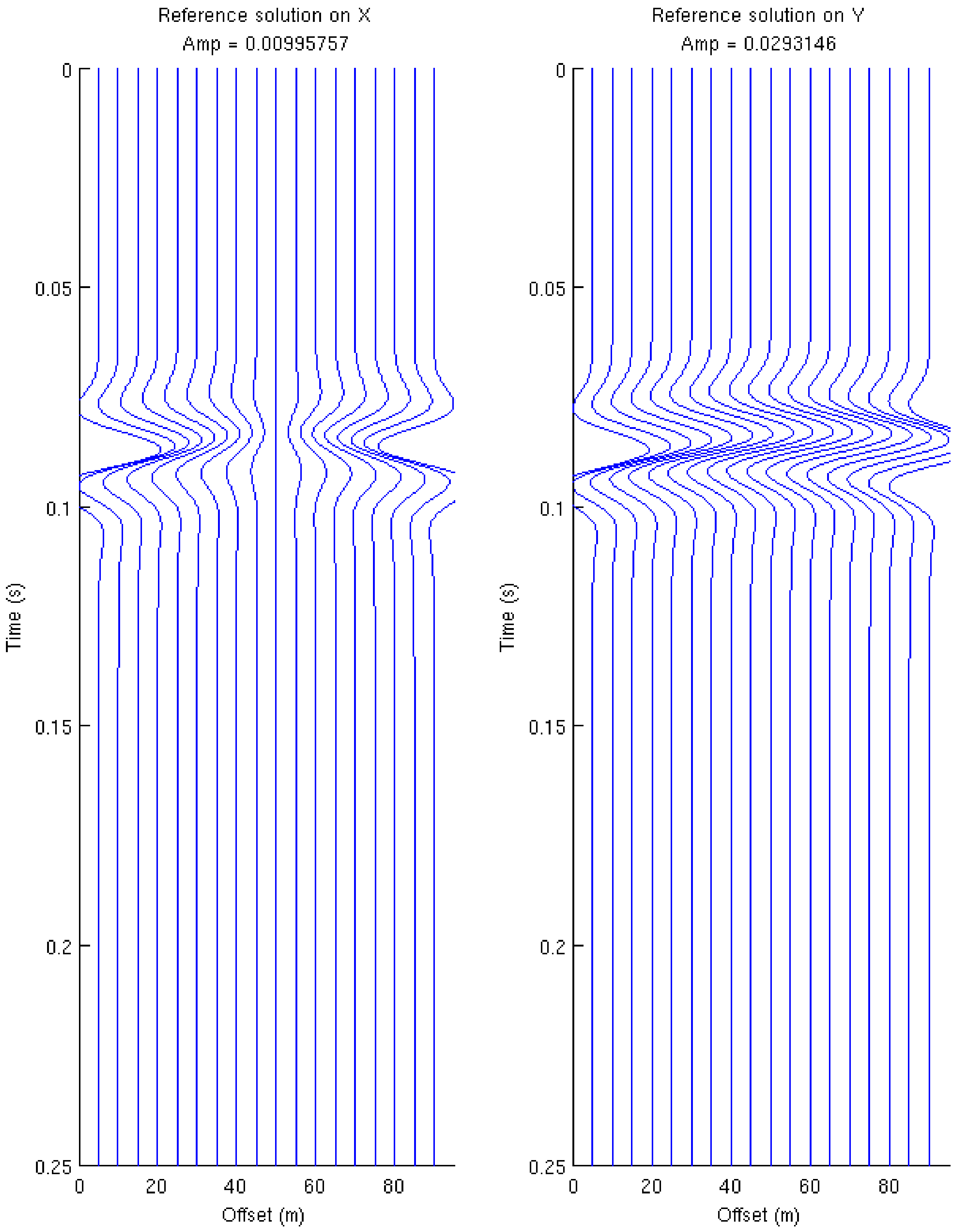


Figure 3.18: Reference solution u_{ref} for tests on Q_3 coarse elements.

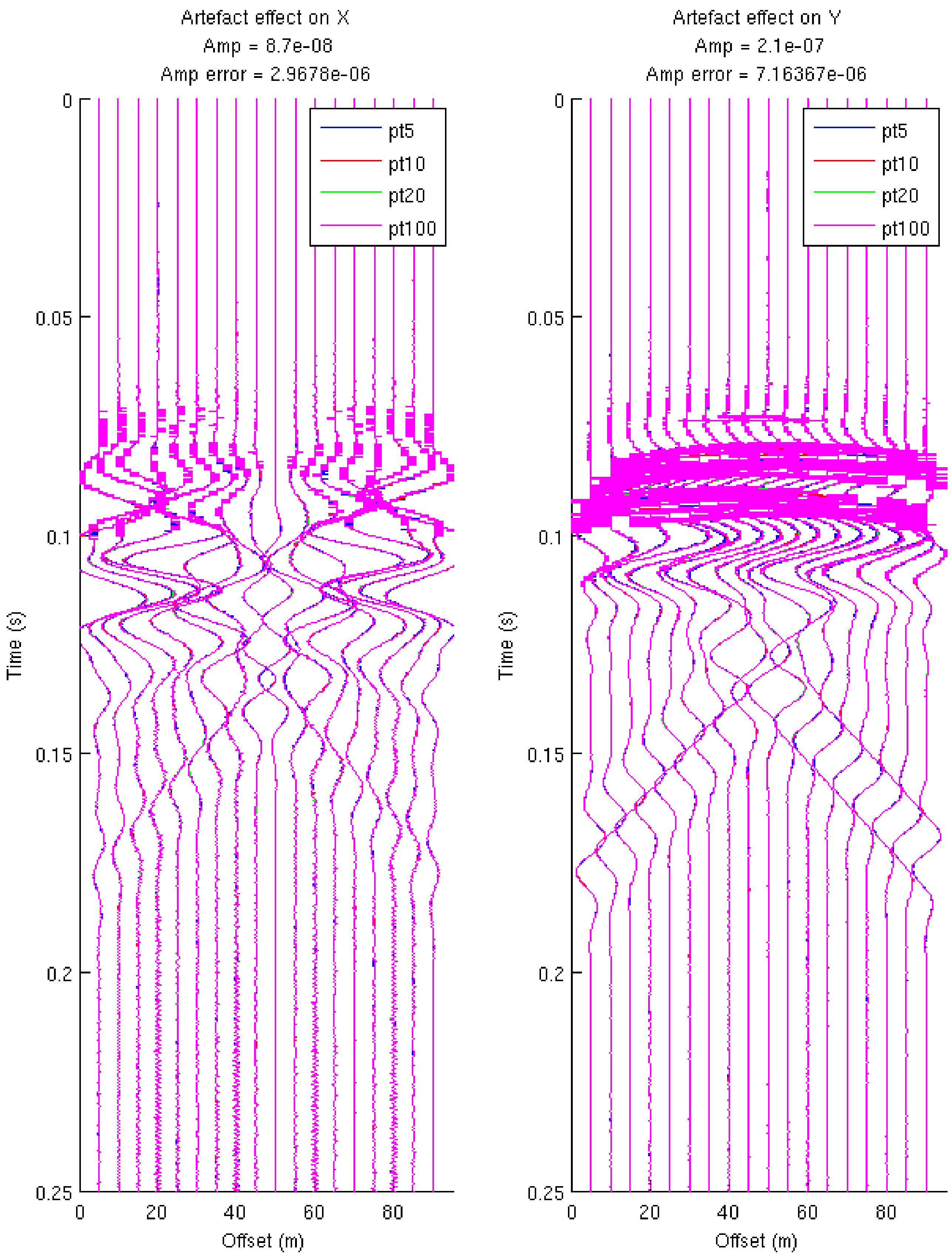


Figure 3.19: Spurious effects $(u_{ref} - u_h)$ for different time refinements $pt = 5, 10, 20, 100$.

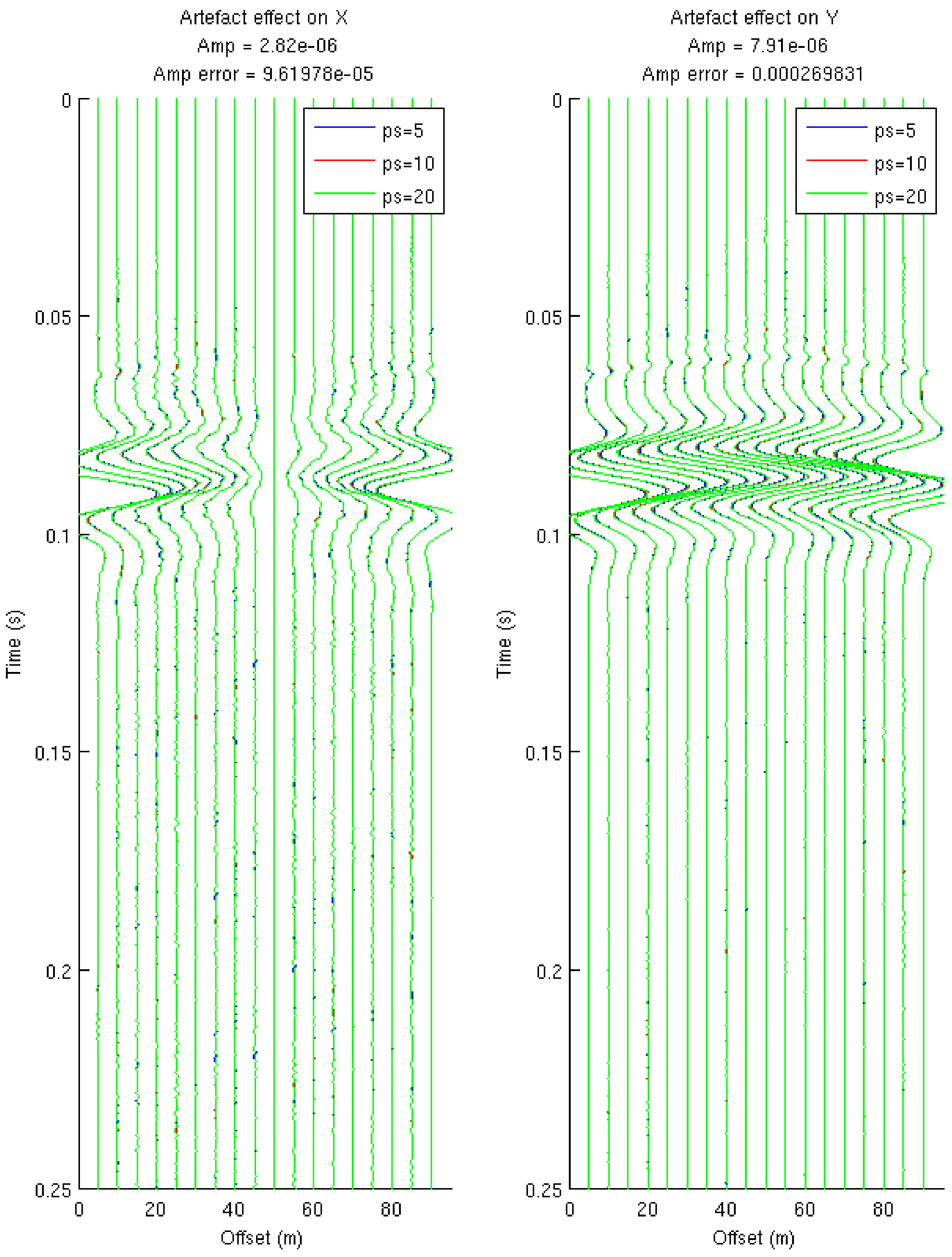
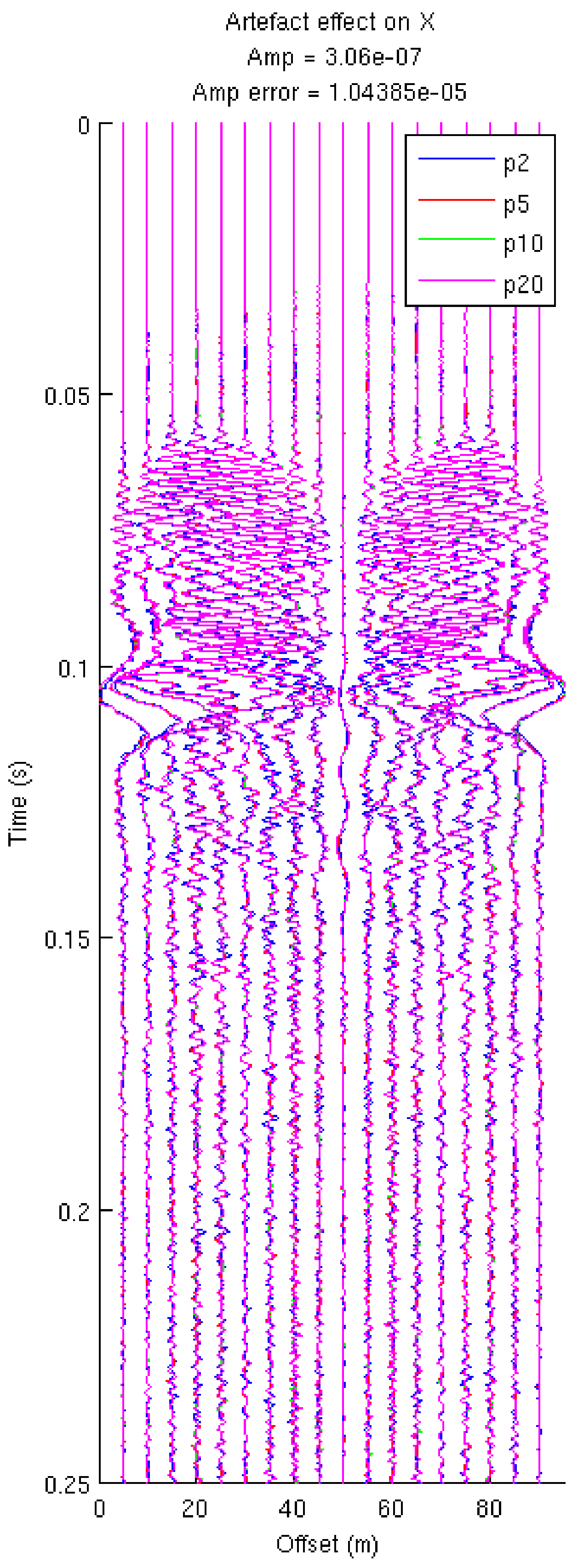


Figure 3.20: Spurious effects $(u_{ref} - u_h)$ for different space refinements $ps = 5, 10, 20$.



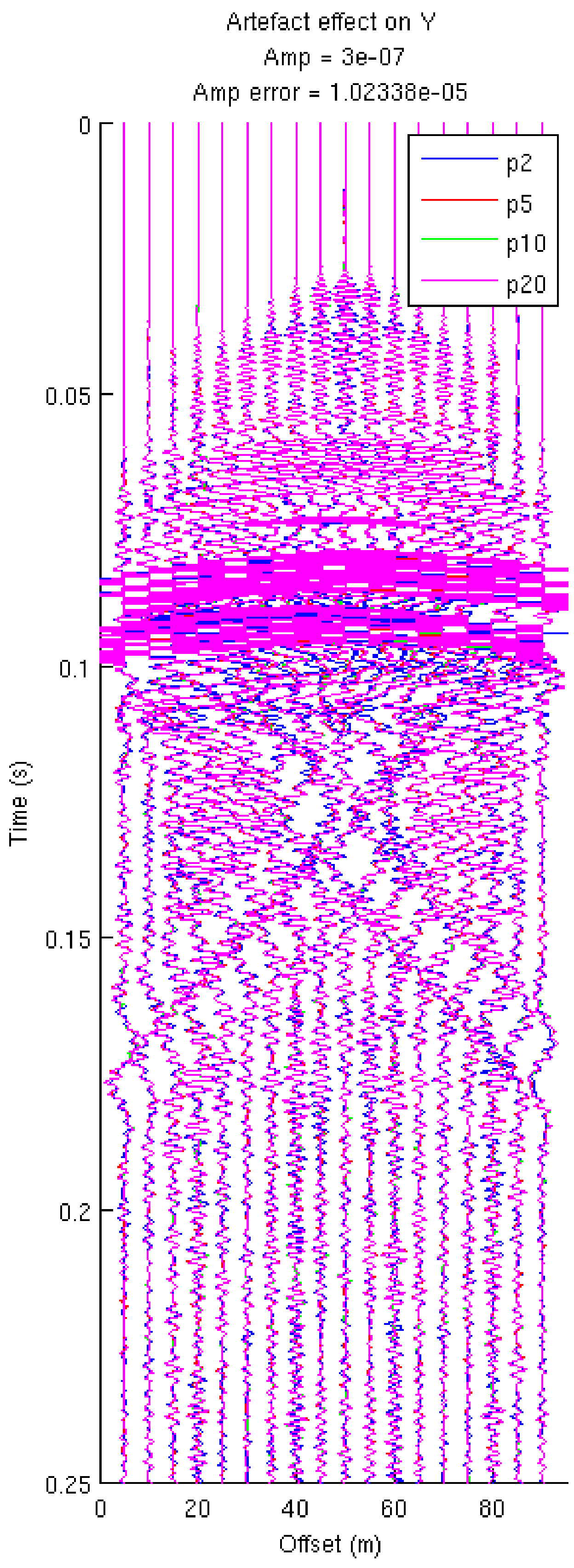


Figure 3.21: Spurious effects $(u_{ref} - u_h)$ for different space and time refinements $p = 5, 10, 20$.

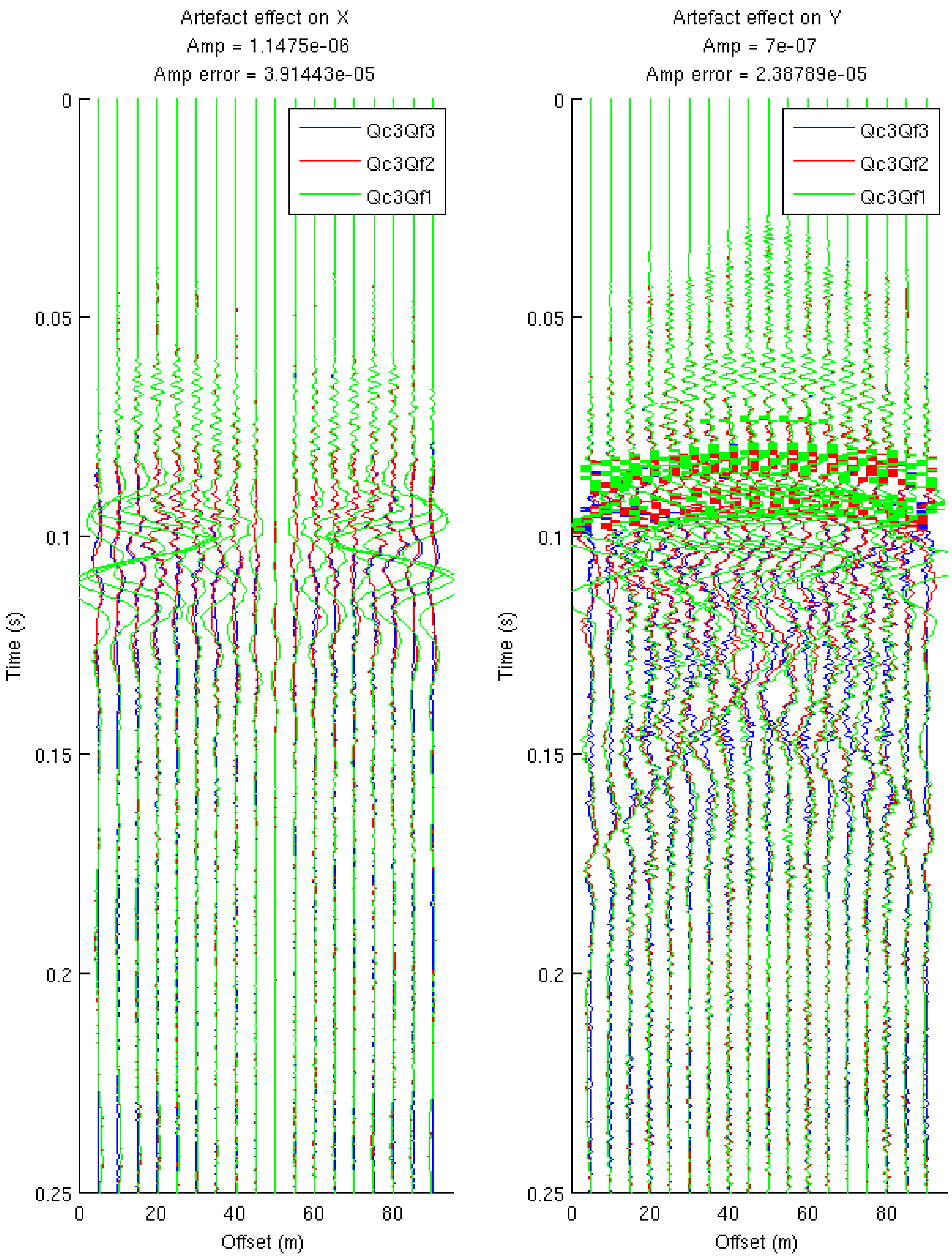


Figure 3.22: Spurious effects $(u_{ref} - u_h)$ for Q_3 coarse elements and Q_3, Q_2, Q_1 fine elements and with temporal refinement $pt = 20$ and spatial refinement $ps = 20$.

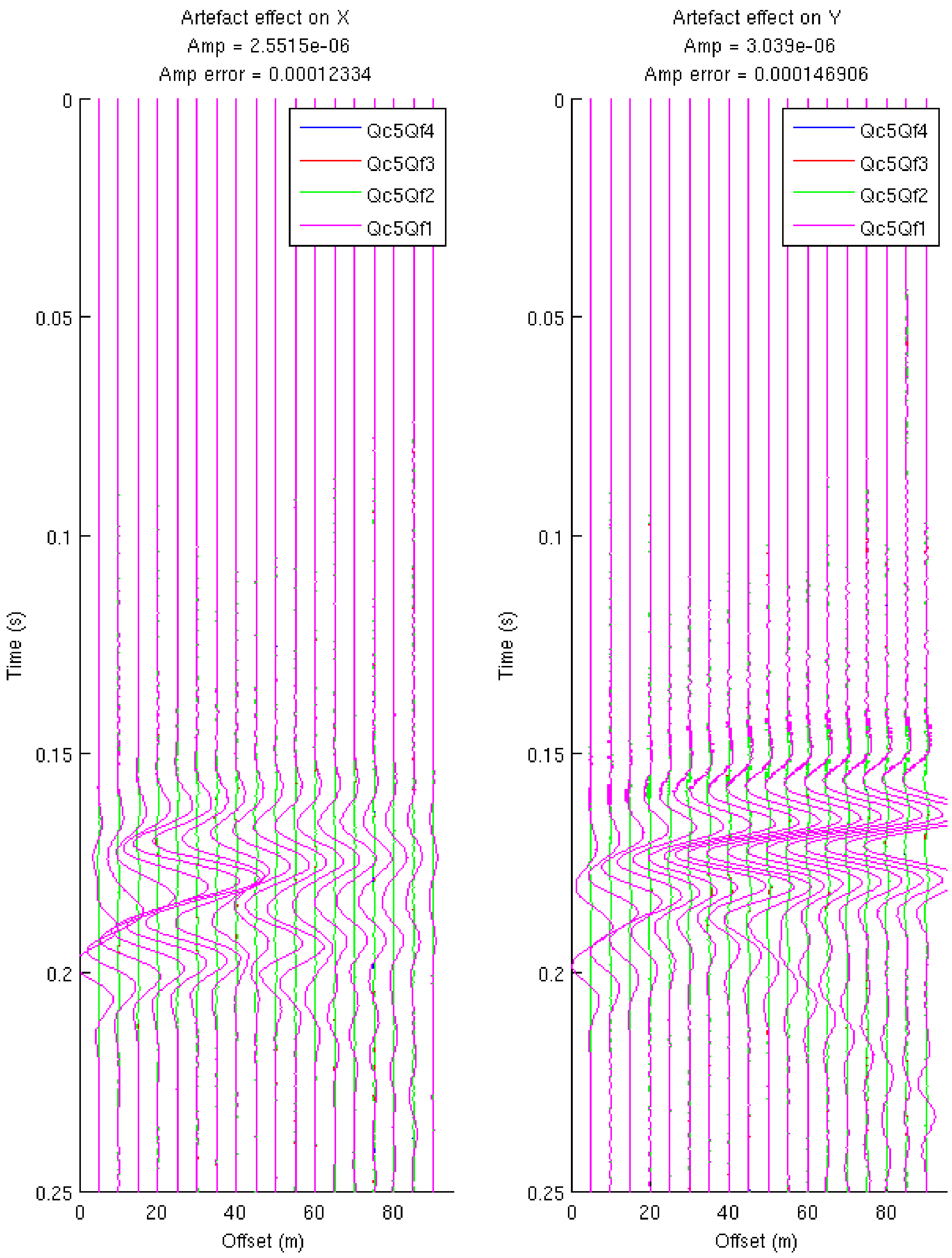
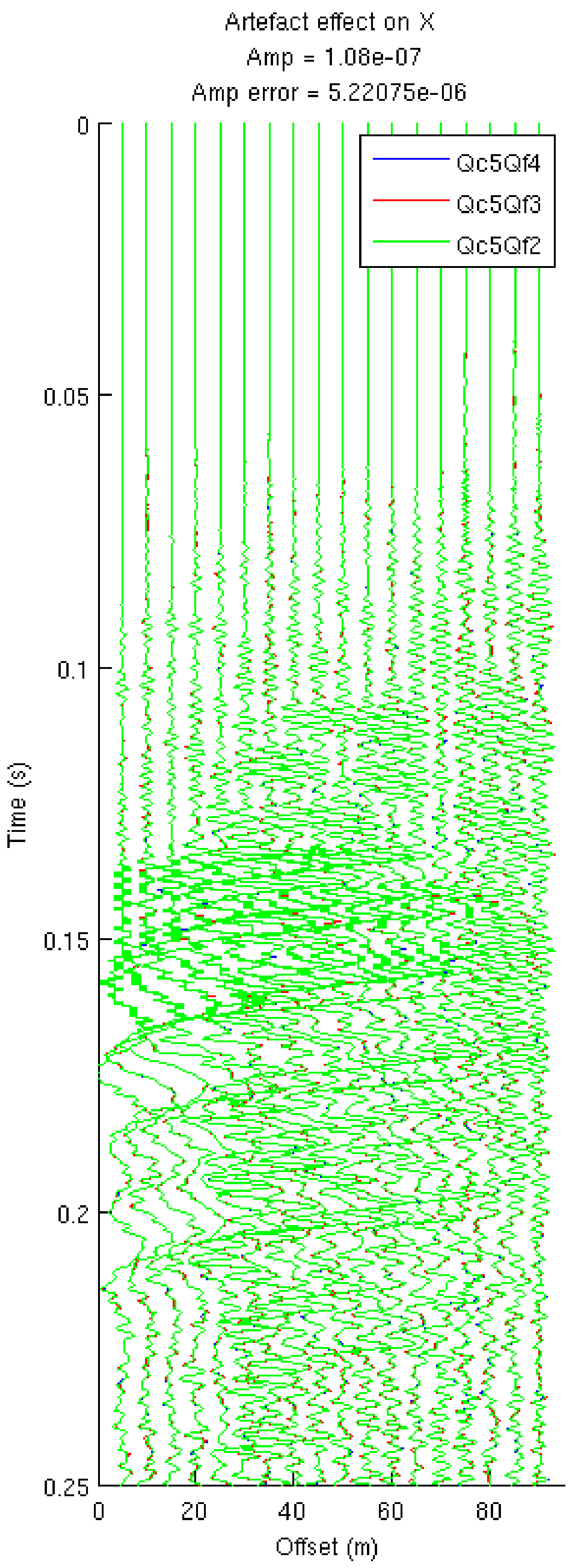


Figure 3.23: Spurious effects $(u_{ref} - u_h)$ for Q_5 coarse elements and Q_4, Q_3, Q_2, Q_1 fine elements and with temporal refinement $pt = 20$ and spatial refinement $ps = 20$.



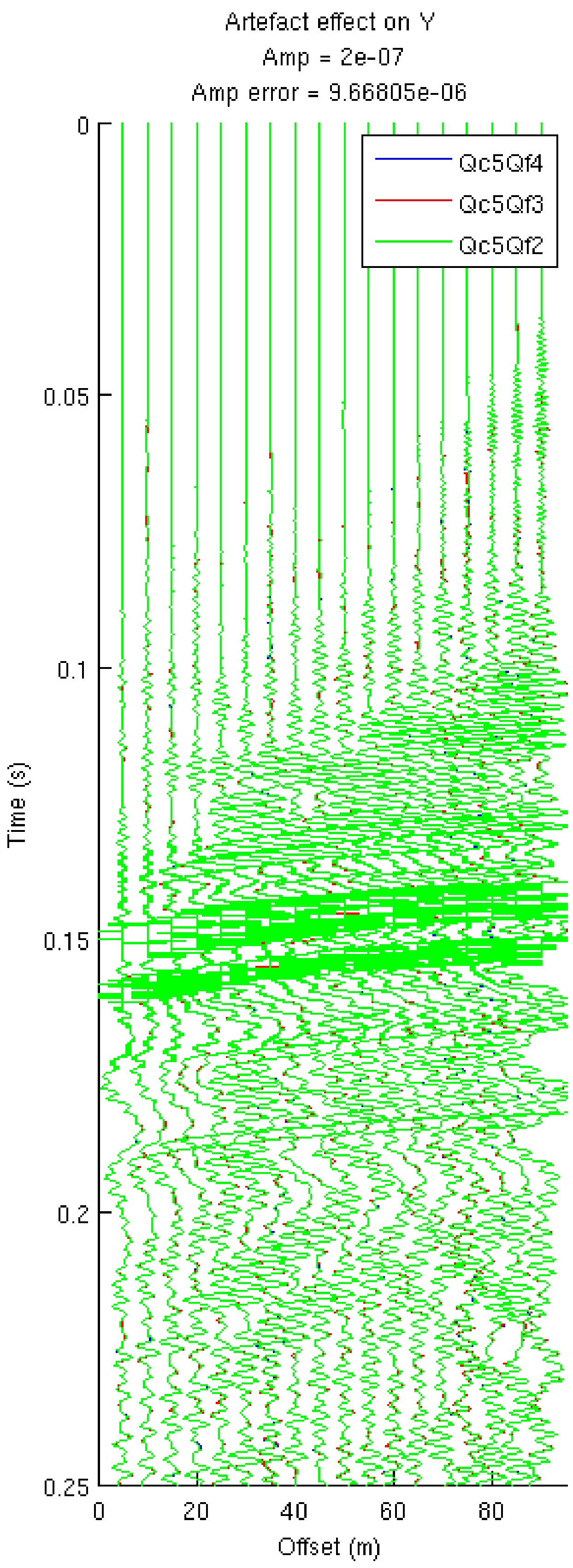


Figure 3.24: Spurious effects $(u_{ref} - u_h)$ for Q_5 coarse elements and Q_4, Q_3, Q_2 fine elements and with temporal refinement $pt = 20$ and spatial refinement $ps = 20$.

3.5 Conclusion

We have presented our strategy to implement local space-time mesh refinements based on J. Diaz and M. Grote's local time stepping scheme. We emphasize the way we introduce the method through the $\tilde{z}$-exact scheme, that helps having a better insight on how the method is built. We found the $\tilde{z}$-exact scheme especially enlightening due to all the properties both schemes share. In particular, the stability and the impact of the *halo* on the stability is shared, even though the $\tilde{z}$-exact scheme does not use local time step. We also showed the really good behavior of the combination of the local time-stepping algorithm and of the non-conforming mesh refinement. Even for severe levels of refinement, artifact effects remain of really low amplitude. We also introduced several strategies to significantly reduce the computational cost. These strategies are really exploiting the strengths of discontinuous Galerkin finite element methods, thus providing a way to monitor the relatively high cost of these methods.

Chapter 4

Numerical results

4.1 Introduction

In this chapter we want to show that our method is able to treat unbounded isotropic highly heterogeneous media, especially highly heterogeneous local areas. This implies to be able to take into account multi-scale phenomena. We want to show that our local space-time mesh refinement catches as well as globally fine meshes the small scale phenomena created by local heterogeneities. In most cases where local mesh refinement is needed, using a globally fine mesh would be impossible, because of the cost it would induce.

Our methodology to show the good behavior is as follows. First, compare with an analytical solution when possible. For more complex experiments we compare the locally refined simulation with a solution globally refined both in space and time. We also compare our results with existing results, or we observe if we have the expected phenomena from theory, *i.e.* ray-theory.

For our numerical experiments we use an explosive source located at the point $\mathbf{x}_S$, that is

$$\mathbf{f}(\mathbf{x},t) = h(t)g(|\mathbf{x}-\mathbf{x}_S|)\frac{\overrightarrow{\mathbf{x}-\mathbf{x}_S}}{|\mathbf{x}-\mathbf{x}_S|}$$

where $h(t)$ is a second order Ricker, with central frequency f_0,

$$h(t) = (2\pi^2(f_0t-1)^2-1)e^{-\pi^2(f_0t-1)^2},$$

and $g(|\mathbf{x}-\mathbf{x}_S|)$ is a regularization of a Dirac by a Gaussian centered in $\mathbf{x}_S$ and distributed over a disk of radius r_0,

$$g(|\mathbf{x}-\mathbf{x}_S|) = \frac{e^{-7\frac{|\mathbf{x}-\mathbf{x}_S|^2}{r_0^2}}}{r_0^2}.$$

The outline of this chapter is the following. We first show some purely elastic experiments to validate both our discontinuous Galerkin approach and our unstructured local mesh refinement approach. In the second section we focus on elasto-acoustic experiments. And we finish with some realistic experiments approaching industrial problems.

4.2 Few words about the rendering method

In this chapter, we sometimes use a snapshot graphical representation to analyze the results obtained by the GD method presented above. Unfortunately, there is currently no tool able to directly exploit numerical solutions from a high-order code. Indeed, the current visualization tools (Paraview, Tecplot, gmsh) are based on a representation using P_1 Lagrange finite elements on simplices with the input data: they use a linear interpolation of the data. If we are not careful, we can lose a lot of information in the graphical representation of our numerical solutions and lose the whole point of using a high precision method. To overcome this problem, ONERA has developed an adaptive method guided by an indicator of subsequent display error allowing the construction of optimized P_1 approximation (*i.e.* limiting the amount of data generated) a solution to a given [**?**] accuracy. This approximation allows us an accurate representation of the numerical solution to within an error fixed by a software standard viewing. In this thesis, we used a software provided by ONERA and based on this method to generate our snapshots. For example, in Figure 4.1, one can see an example of the use of this approach to a solution Q7. The Cartesian grid (pink) that is used for calculating the GD and the triangular mesh of the mesh is obtained by representation of the adaptive method P1 and defining approximation. It is noted that the approach allows a very good rendering of the wealth of information contained in the Q7 digital solution.

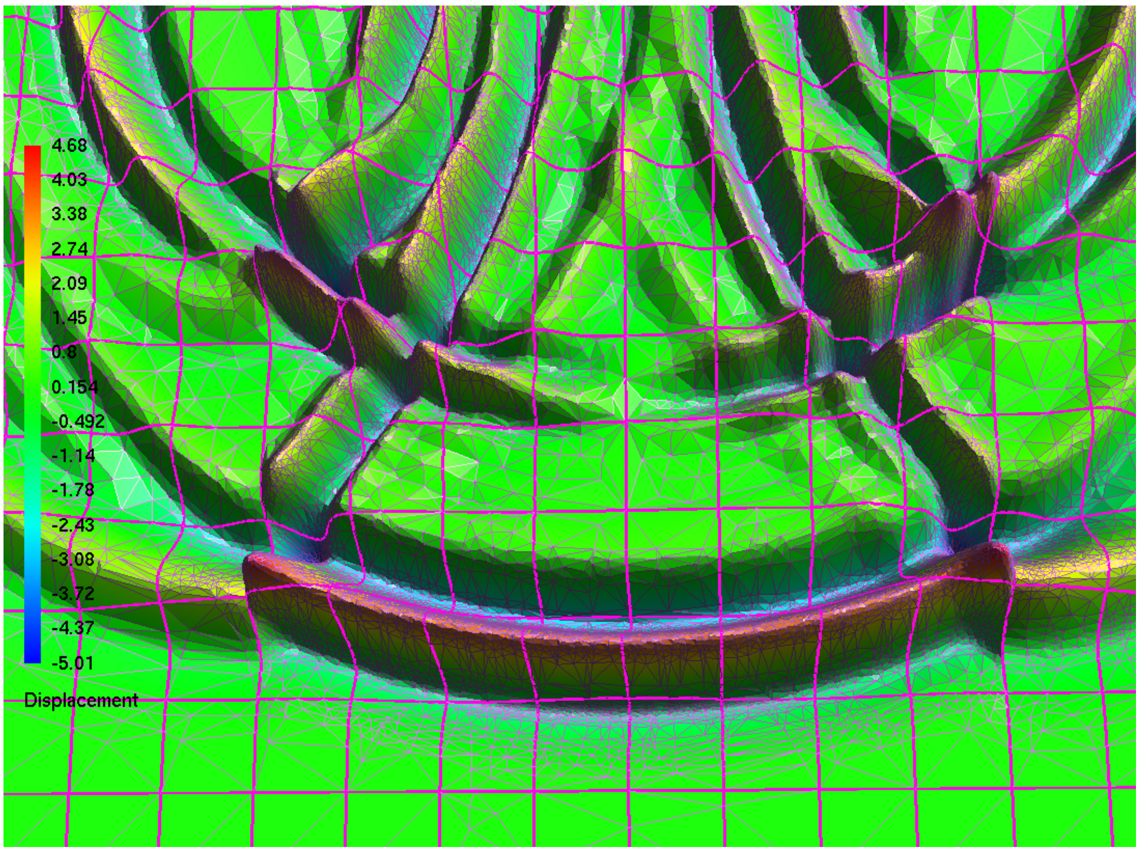


Figure 4.1: Snapshot of a simulation with the computation grid and representation elements highlighted.

4.3 Elastodynamic experiments

4.3.1 Two-layered medium

In this experiment we want to show the ability of the discontinuous Galerkin method without refinement to treat an heterogeneous case compared to the exact solution. In order to do so, we simply compare our solution to the analytical solution on a two-layered medium given by J. Diaz's code Gar6more [**?**].

We propose a simple test case made of a two layered medium. The top layer has the following characteristics: $\lambda = 1.9 \times 10^{10}$, $\mu = 5.5 \times 10^{9}$, $\rho = 3200\ kg.m^{-3}$, $v_p = 3061\ m.s^{-1}$ and $v_s = 1311\ m.s^{-1}$ and the bottom layer has the following characteristics: $\lambda = 7.7612 \times 10^{10}$, $\mu = 5.994 \times 10^{9}$, $\rho = 1850 kg.m^{-3}$, $v_p = 4000 m.s^{-1}$ and $v_s = 1800 m.s^{-1}$. We positioned a pressure regularized Ricker source of central frequency $20Hz$ in the center of the medium, $50m$ above the the two layers interface. We positioned a line of 100 receivers $150m$ above the interface from the abscissa $-200m$ up to $200m$.

The two different simulations displayed on Figure 4.2 and Figure 4.3, which are the analytical solution obtained from Gar6more and our solution made of Q_7 elements, these elements are of size $25m$. In order to compare these two simulations, we compare the line of seismograms for the displacement X and Y. We note that we obtained similar solutions with both methods, *i.e.* arrival times and amplitudes are the same.

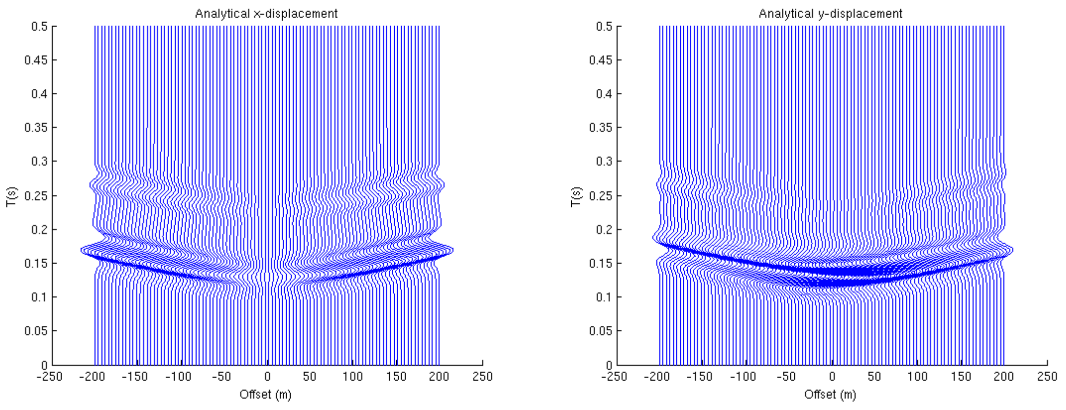


Figure 4.2: Analytical solution.

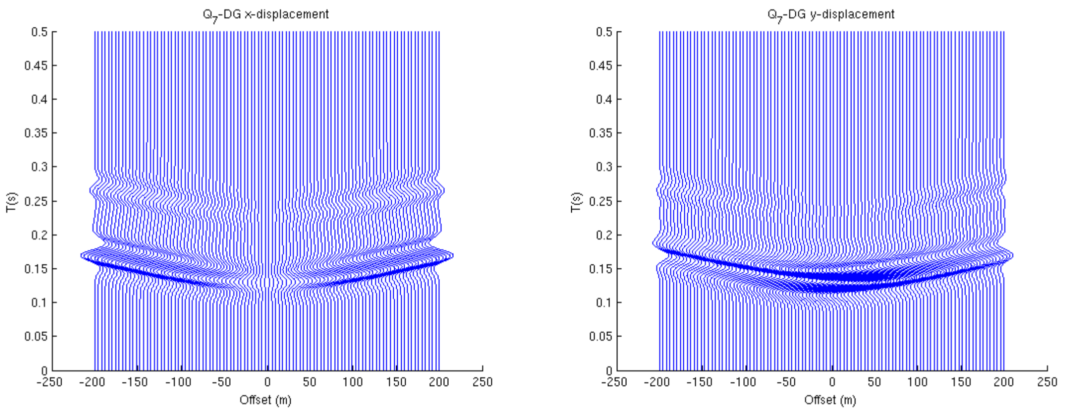


Figure 4.3: Discontinuous Galerkin solution.

4.3.2 Academic test case for local space-time refinement

In this experiment we want to show the good behavior of our method with local space-time mesh refinement in the case of an heterogeneous medium. In order to do so we will compare the locally refined simulation to a fully fine simulation. An heterogeneous differs from the homogeneous test case since the solution might lose regularity in the refined area. Thus, we wonder if the method is able to restore the singular behavior of the solution. We also wonder if the refined area is large enough to catch all this singular phenomenon. However, we will not attempt to characterize how a mesh should be to describe well a singularity, but simply note that all seems to behave well with a refined area of a limited size.

We propose a simple test case made of an homogeneous medium with Dirichlet boundary conditions with a thin layer included in the medium and perturbing the propagation. The homogeneous part of size $200m \times 200m$ has the following characteristics: $\lambda = 7.774 \times 10^9$, $\mu = 3.887 \times 10^9$, $\rho = 2300\,kg.m^{-3}$, $v_p = 2600\,m.s^{-1}$ and $v_s = 1300\,m.s^{-1}$. The thin layer of size $112m \times 0.4m$ has the following characteristics: $\lambda = 5.76 \times 10^8$, $\mu = 4.032 \times 10^9$, $\rho = 1600kg.m^{-3}$, $v_p = 1800m.s^{-1}$ and $v_s = 600m.s^{-1}$. The thin layer is horizontal and positioned at $45m$. We positioned a pressure regularized Ricker source of central frequency $20Hz$ in the center of the medium.

The two different simulations, which we called the locally fine and and fully fine simulations, are both made of Q_3 elements, these elements are of size $0.4m$ for the fully fine simulation and for the refined area of the locally fine simulation, the rest of the locally fine simulation is made of elements of size $4m$. In order to compare these two simulations, we compare seismograms in three different points positioned position above the thin layer $A = (50m, 55m)$, $B = (100m, 55m)$ and $C = (150m, 55m)$. For the locally fine simulation, the refined area is of size $120m \times 4m$.

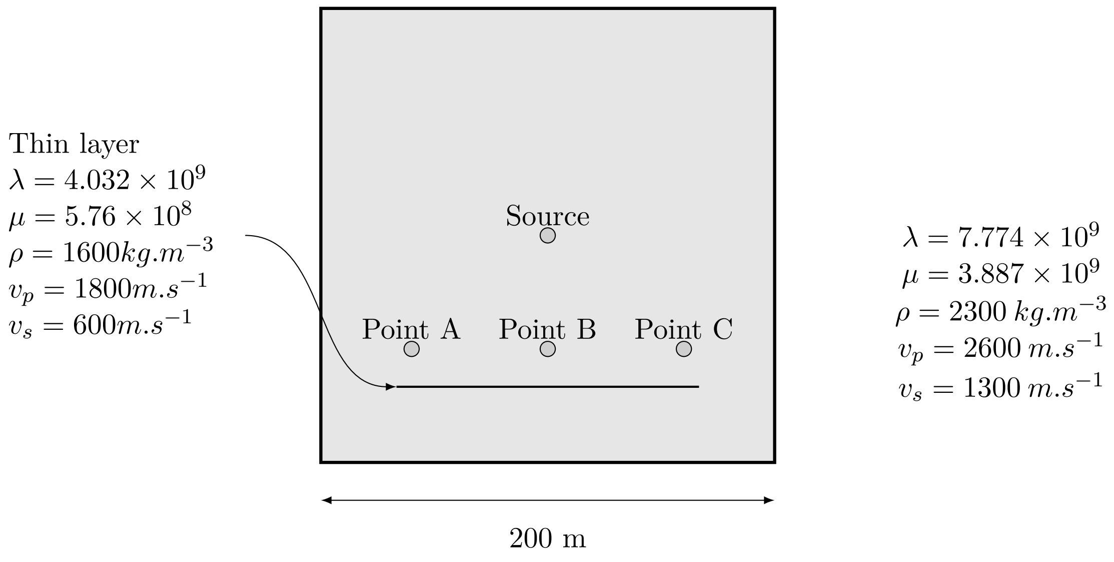


Figure 4.4: Academic test case medium characteristics.

We display on Figure 4.5, 4.6 and 4.7 the seismograms of X and Y displacements for the locally fine and fully fine simulations. We can note that we do not observe any differences between the two simulations. We conclude that the refined area introduces no spurious effect and catches all the effects created by the small heterogeneity as if the whole mesh was fine.

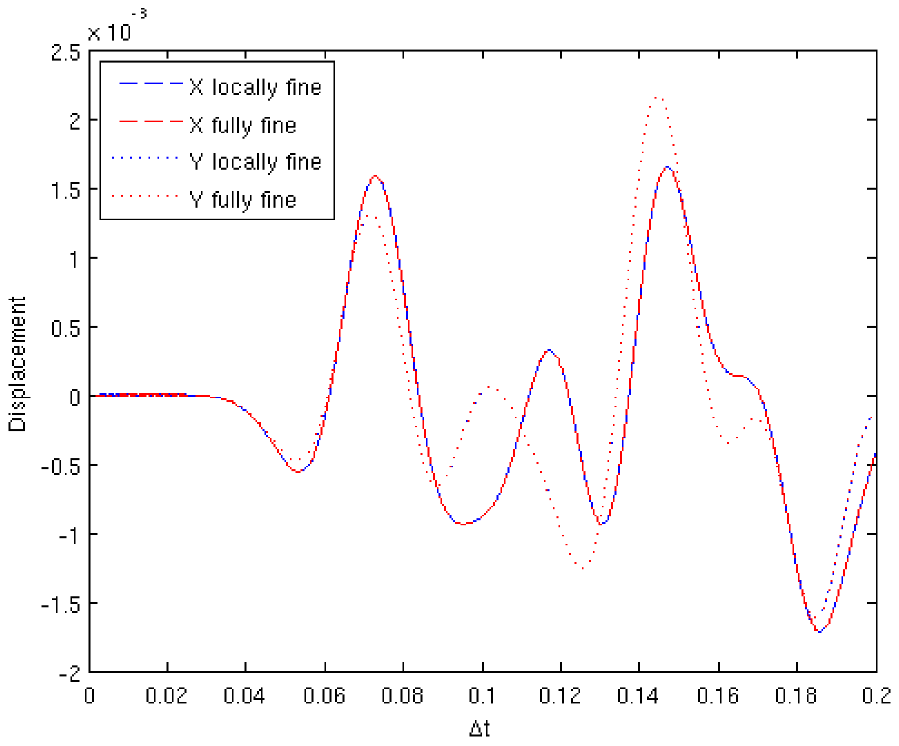


Figure 4.5: Comparison of locally and fully refined mesh sismos at point A.

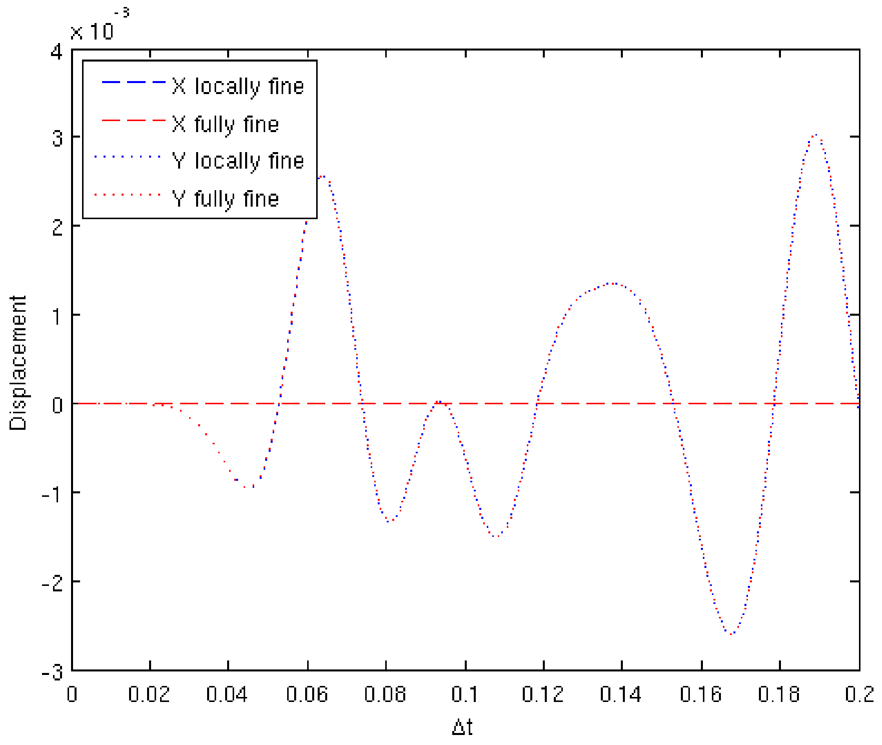


Figure 4.6: Comparison of locally and fully refined mesh sismos at point B.

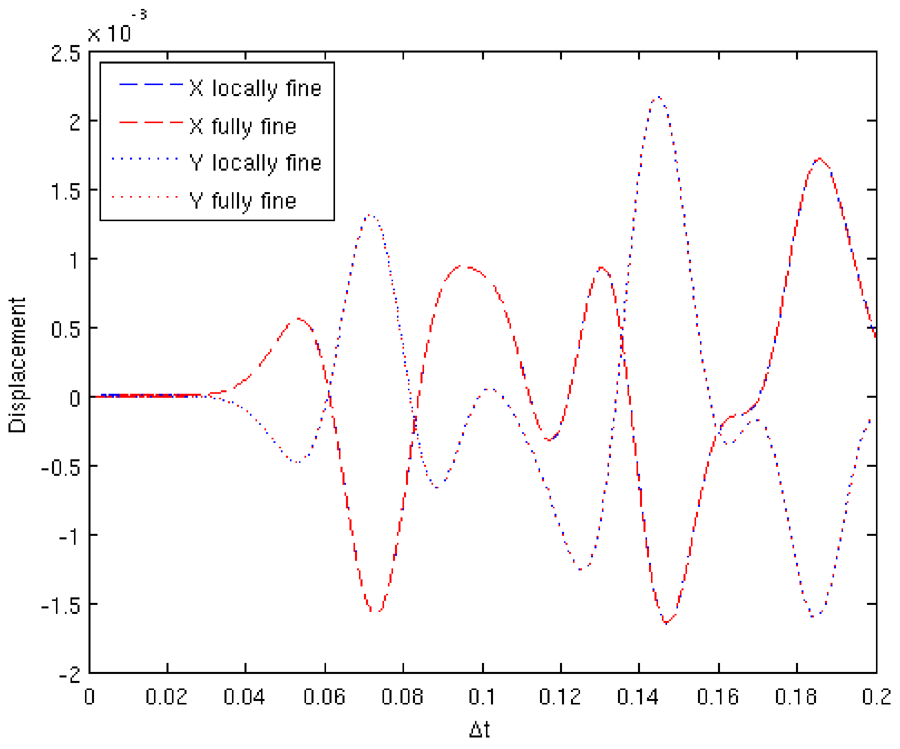


Figure 4.7: Comparison of locally and fully refined mesh sismos at point C.

4.4 Elasto-acoustic experiments

In geophysics, problems that require local mesh refinement are rarely purely elastic problems, some acoustic phenomena are of great importance. The small heterogeneities that lead to the biggest impact on the wave propagation are often made of fluid, *i.e.* cracks. For this reason we though interesting to have a method that can both handle elastic and acoustic media. As we shall see it is not completely trivial to handle both elasticity and acoustic in the same formulation since the continuities are not the same. Usually, the approach is to have two different methods for elasticity and acoustic and to couple by enforcing the transmission condition. This possibility is certainly the best when the acoustic area is something large, e.g. the ocean or a lake. However, in our case it would be really cumbersome to have to couple all erratic small acoustic parts in the refined area, for this reason we preferred to develop a method that naturally handle both elasticity and acoustic without having to couple anything. This comes at the cost that both elastic and acoustic media are using displacement unknowns, whereas the standard way would use pressure unknowns in the acoustic parts and displacement unknowns in the elastic parts. Moreover, having a unified formulation for elasto-acoustic media allows us to keep the same methodology for the implementation.

In this section we first introduce our DG formulation to handle such configurations, then we validate our method and finally we give illustrative examples to show the impact of such heterogeneities.

4.4.1 Split formulation for elasto-acoustic simulations

The main difficulty when one attempts to have an unified method for elasticity and acoustic is that the two equations do not impose the same continuity on the displacement and its derivatives, *i.e.* setting $\mu = 0$ in the discontinuous Galerkin elastic formulation would not be correct because flux terms would impose too much continuities.

Since all continuities in DG methods are implicit, we shall first recall which continuities are induced by our model problem. The standard DG approximation for elastodynamic impose continuities which have no place in the case of an acoustic-acoustic or elastic-acoustic interface. We will consider the three cases, elastic-elastic interface, elastic-acoustic and acoustic-acoustic interface in order to understand which continuities should be imposed according to the case.

Since $u \in L^2(\Omega)$ and $div(\sigma(u)) \in L^2(\Omega)$, which implies that $[\![u{\cdot}\mathbf{n}]\!] = 0$ and $[\![\sigma(u)\mathbf{n}]\!] = 0$, we get the following transmission conditions

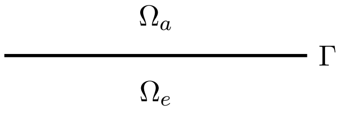


Figure 4.8: Elasto-acoustic interface.

We denote by the subscript N and P the normal and tangential component of a vector relatively to a face.

Elastic - Elastic On Γ we have the following transmission condition

$$\begin{cases} u_K = u_{K'}, \\ \sigma(u_K) = \sigma(u_{K'}). \end{cases}$$

We can rewrite this condition as follow

$$\begin{cases} u_K = u_{K'}, \\ (\sigma(u_K)n_K)_N = (\sigma(u_{K'})n_{K'})_N, \\ (\sigma(u_K)n_K)_T = (\sigma(u_{K'})n_{K'})_T. \end{cases}$$

Elastic - Acoustic On Γ we have the following transmission condition

$$\begin{cases} u_E \cdot n_E = -u_A \cdot n_A, \\ \sigma(u_E)n_E = \lambda_A div(u_A)n_A. \end{cases}$$

We can rewrite this condition as follow

$$\begin{cases} (u_E)_N = (u_A)_N, \\ (\sigma(u_E)n_E)_N = \lambda_A div(u_A), \\ (\sigma(u_E)n_E)_T = 0. \end{cases}$$

Acoustic - Acoustic On Γ we have the following transmission condition

$$\begin{cases} (u_K)_N = (u_{K'})_N, \\ (\lambda_K div(u_K)n_K)_N = (\lambda_{K'} div(u_{K'})n_{K'})_N. \end{cases}$$

In the case of an acoustic-acoustic or elasto-acoustic interface there is no continuity constraint on the tangential component of the displacement, thus there must not be any tangential jumps on those interfaces since they would impose the tangential continuity of the displacement. In the case of an acoustic-acoustic interface the tangential component of the term $\int_F \{\!\{\sigma(u)n\}\!\} \cdot v d\gamma$ is zero by construction. However in the case of an elasto-acoustic interface this term has to be set to zero to strongly impose the tangential continuity of $\sigma(u) \cdot \mathbf{n}$. We cannot use the standard flux term and let the scheme find by itself the good continuity since we would need to penalize this flux term. We do not want to penalize the tangential component since it would also impose the continuity of the tangential component of the displacement which has no reason to be. Indeed, since this term comes from the integration by part, forcing it to zero imposes the method to find null tangential components for the derivatives of the displacement without having to had any penalty. All this results in the following modified local DG bilinear form

$$\begin{aligned} a_h^K(\mathbf{u},\mathbf{v}) = &\int_K \sigma_h^K(\mathbf{u}) : \nabla \mathbf{v}\, dx - \sum_{F \in \mathcal{F}_h^K} \int_F \{\!\{\sigma_h(\mathbf{u})n\}\!\}_N \cdot \mathbf{v}_N\, d\gamma - \sum_{F \in \mathcal{F}_h^K} \int_F \Theta_F \{\!\{\sigma_h(\mathbf{u})n\}\!\}_T \cdot \mathbf{v}_T\, d\gamma \\ &- \sum_{F \in \mathcal{F}_h^K} \int_F [\![\mathbf{u}]\!]_N \frac{1}{2} (\sigma_h^K(\mathbf{v})n)_N\, d\gamma - \sum_{F \in \mathcal{F}_h^K} \int_F \Theta_F [\![\mathbf{u}]\!]_T \frac{1}{2} (\sigma_h^K(\mathbf{v})n)_T\, d\gamma \\ &+ \sum_{F \in \mathcal{F}_h^K} \int_F \alpha_F [\![\mathbf{u}]\!]_N \cdot \mathbf{v}_N\, d\gamma + \sum_{F \in \mathcal{F}_h^K} \int_F \Theta_F \alpha_F [\![\mathbf{u}]\!]_T \cdot \mathbf{v}_T\, d\gamma, \end{aligned}$$

where

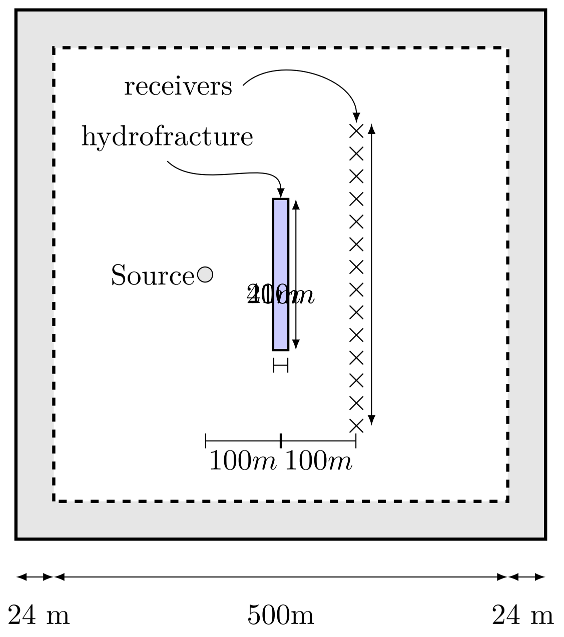
$$\Theta_F = \begin{cases} 1 & \text{if Elastic-Elastic face,} \\ 0 & \text{otherwise,} \end{cases}$$

and where the subscript N and T denote the normal and tangential component respectively.

4.4.2 Validation: scattering by a hydrofracture

The model geometry used to generate the seismograms is shown in Figure 4.9. The source, the receivers and hydrofracture are situated in an elastic medium ($v_p = 3500 m.s^{-1}$, $v_s = 2023 m.s^{-1}$ and $\rho = 2300 kg.m^{-3}$). We used a pressure regularized Ricker source of central frequency $100 Hz$ situated at the origin. The seismograms are realized with 80 receivers disposed between $-200m$ and $200m$. The center line of the fracture lies between $(100m, -100m)$ and $(100m, 100m$, and is $1m$ wide. The hydrofracture is modelled as a single crack represented by a relatively thin rectangle filled with water ($v_p = 1500 m.s^{-1}$ and $\rho = 1020 kg.m^{-3}$).

We used Q_5 elements, with a space step of $4m$. The refined area is spatially refined by a factor $p_s = 4$ with Q_5 elements, and temporally refined by a factor $p_t = 4$. We added $24m$ of PML around our domain.



$$\lambda = 9.3494 \times 10^9$$
$$\mu = 9.4128 \times 10^9$$
$$\rho = 2300\, kg.m^{-3}$$
$$v_p = 3500\, m.s^{-1}$$
$$v_s = 2023\, m.s^{-1}$$

Figure 4.9: Medium with large hydrofracture characteristics.

We display on Figure 4.12 the seismograms we obtained, they can be compared with the ray-theoretical traveltimes on Figure 4.10 or they can also be compared to the seismograms displayed on Figure 4.11 obtained by indirect boundary element method. Since the source is a pressure source, we expect only one P-wave incoming on the fracture. This P-wave is then transmitted inside the fracture into another P-wave, since there is no S-waves in acoustic media. Finally this P-wave is transmitted into a P-wave and an S-wave, this corresponds to the PPP- and PPS-waves front we have on Figure 4.10. There should also be some multiples due to the multiple reflections inside the hydrofracture, however they must be of small amplitude since the angle of incidence is almost normal on the whole fracture. The tips of the fractures also generates some waves, called diffracted waves. Both P- and S-waves are diffracted from the incoming P-wave, this corresponds to PP_d- and PS_d-waves on Figure 4.10. We shall have the PPP-wave arriving first at the receivers, then the two PP_d-waves, then PPS-wave and finally the two PS_d-waves. Both Figure 4.10 and Figure 4.11 were extracted from [**?**]. We note that the seismograms (Figure 4.12) obtained with our method are similar to those obtained with the indirect boundary element method (Figure 4.11) and that we obtain all the reflected and diffracted waves predicted by ray-theory (Figure 4.10). These results validate our local elasto-acoustic approach since boundary element methods and ray-theory give robust reference solutions.

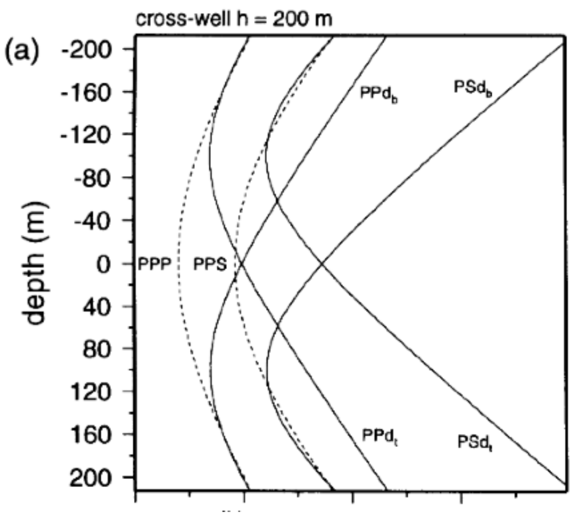


Figure 4.10: Ray-theoretical traveltimes extracted from [?].

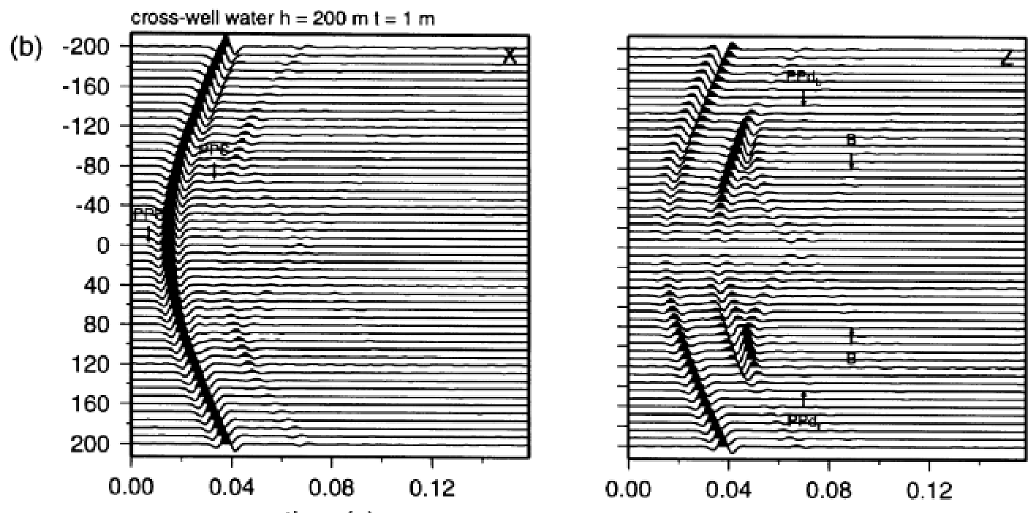


Figure 4.11: Reference seismograms extracted from [?].

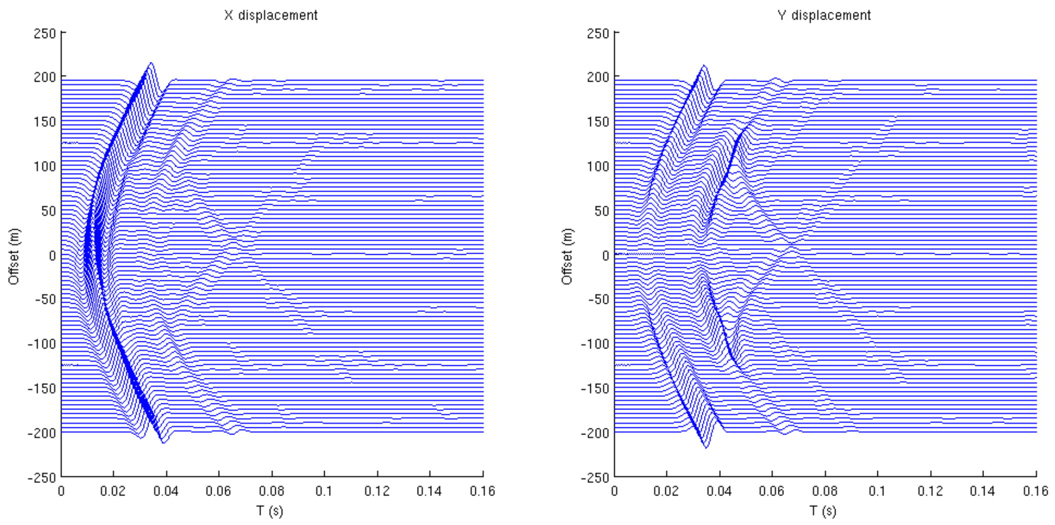


Figure 4.12: Seismograms.

4.5 Illustrative experiments

In this section we want to show illustrative experiments close to industrial problems. We mainly focus on small heterogeneities that resemble to hydrofractures, but real hydrofractures would be much thinner in reality. The impact on wave propagations of those hydrofractures is also highly dependent on several parameters, *i.e.* length, orientation, density, distribution. The impact of those parameters is studied in [**?**] for instance. Here, we simply show the ability of our method to simulate such heterogeneities, without giving to much qualitative analysis on the phenomenon.

4.5.1 Thin fluid-filled crack

In this first illustrative elasto-acoustic experiment we want to show how a tiny crack filled with water can have a great impact on the simulation. However, even if this crack is relatively fine for our experiment compared to the size an element, this crack is still really thick compared to real cracks.

This test case is an homogeneous medium of size $400m \times 400m$ with the following characteristics: $\lambda = 1.7612 \times 10^{10}$, $\mu = 5.994 \times 10^9$, $\rho = 1850\,kg.m^{-3}$, $v_p = 4000\,m.s^{-1}$ and $v_s = 1800\,m.s^{-1}$. The dimension of the crack is $O.4m \times 20m$ and the characteristics of the water are: $\lambda = 2.25 \times 10^9$, $\mu = 0$, $\rho = 1000\,kg.m^{-3}$, $v_p = 1500\,m.s^{-1}$ and $v_s = 0\,m.s^{-1}$. We used a pressure regularized Ricker of central frequency $40Hz$ positioned in center of the medium.

We used Q_3 elements, with a space step of $4m$. The refined area is spatially refined by a factor $p_s = 10$ with Q_1 elements, and temporally refined by a factor of $p_t = 5$. We added $20m$ of PML around our domain.

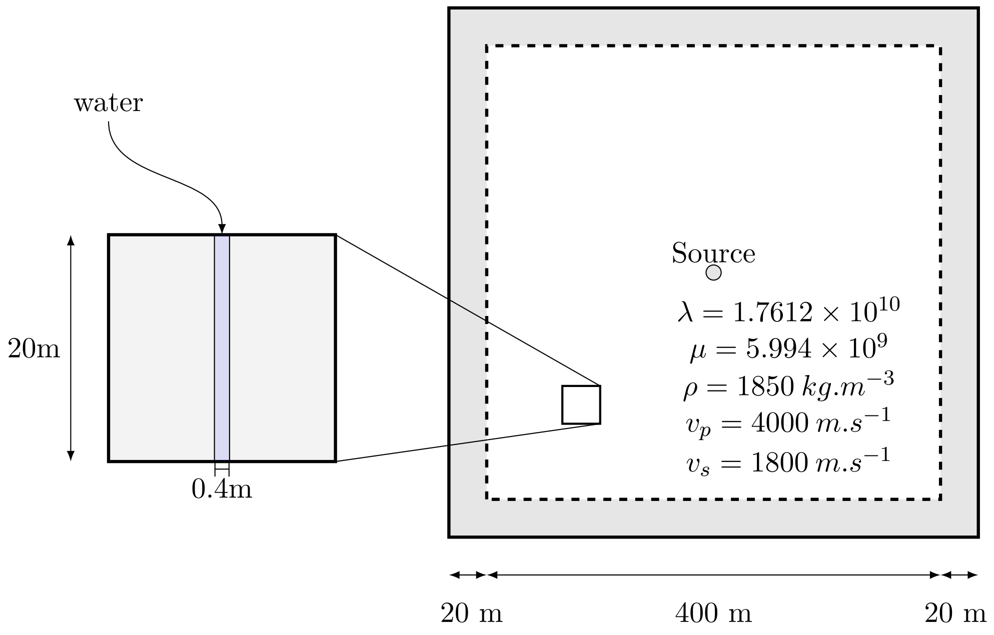

water
20m
0.4m
Source
$\lambda = 1.7612 \times 10^{10}$
$\mu = 5.994 \times 10^9$
$\rho = 1850\, kg.m^{-3}$
$v_p = 4000\, m.s^{-1}$
$v_s = 1800\, m.s^{-1}$
20 m
400 m
20 m

Figure 4.13: Fluid-filled crack medium characteristics.

We display on Figure 4.14 snapshots at different times of X and Y displacements. We note that the impact of only one small crack is undetectable on the primary front wave, however the crack stores some energy and emit its own wave soon after at roughly 10% of the primary front wave (beware of the change of color scale between 0.01s and 0.02s). This resonance effect is most likely similar to the one of a guitar string for instance.

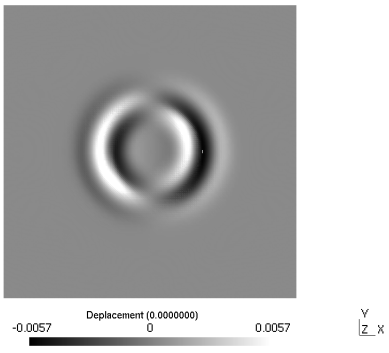
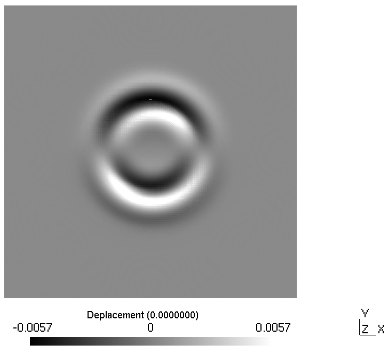
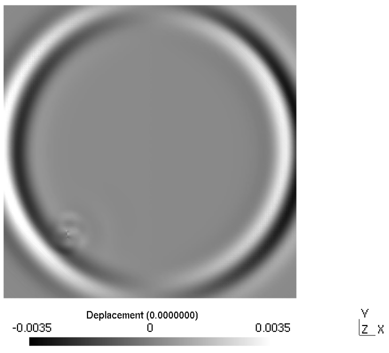
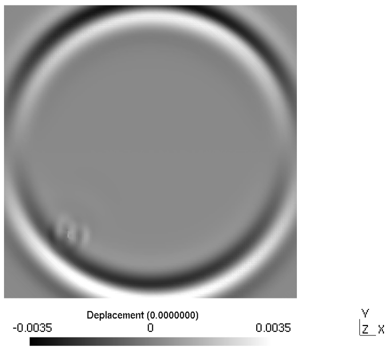
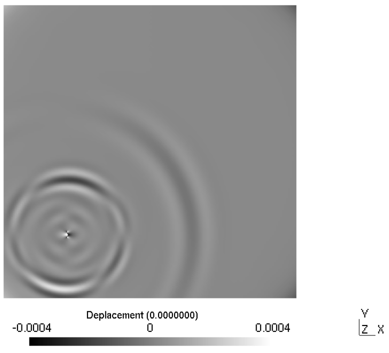
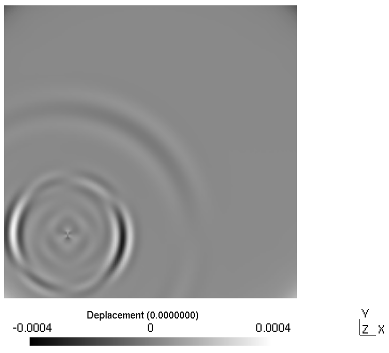
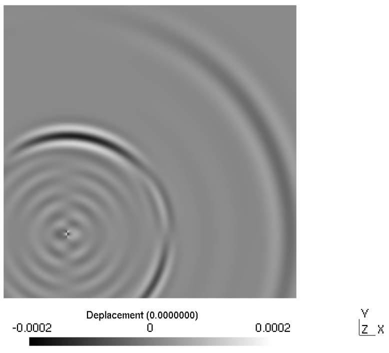
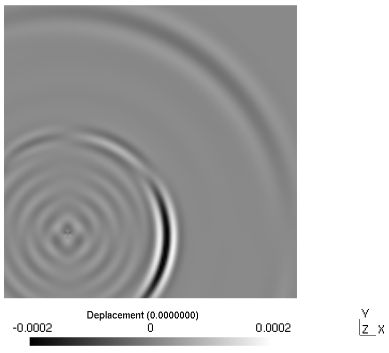
Time	X displacement	Y displacement
$0.005s$		
$0.01s$		
$0.015s$		
$0.02s$		

Figure 4.14: Snapshots at different times of X and Y displacements for a medium with a fluid-filled crack.

4.5.2 Diffracting points

In this second illustrative experiment we want to show how tiny diffracting points filled with water can have a great impact on the simulation. However, the density of diffracting points (around 20%) is superior to what would be relevant for realistic simulations, this was intended in order to have visual snapshots.

We take the same homogeneous medium as in the previous test case with the characteristics: $\lambda = 1.7612 \times 10^{10}$, $\mu = 5.994 \times 10^9$, $\rho = 1850\, kg.m^{-3}$, $v_p = 4000\, m.s^{-1}$ and $v_s = 1800\, m.s^{-1}$. We randomly inserted the diffracted points in a spatially refined area of dimension $20m \times 20m$, these points are squares of size $0.16m$ with water inside. We used a pressure regularized Ricker of central frequency $40Hz$ positioned in $(150m, 150m)$.

We used Q_3 elements, with a space step of $4m$. The refined area is spatially refined by a factor $p_s = 25$ with Q_1 elements, and temporally refined by a factor $p_t = 10$. We added $20m$ of PML around our domain.

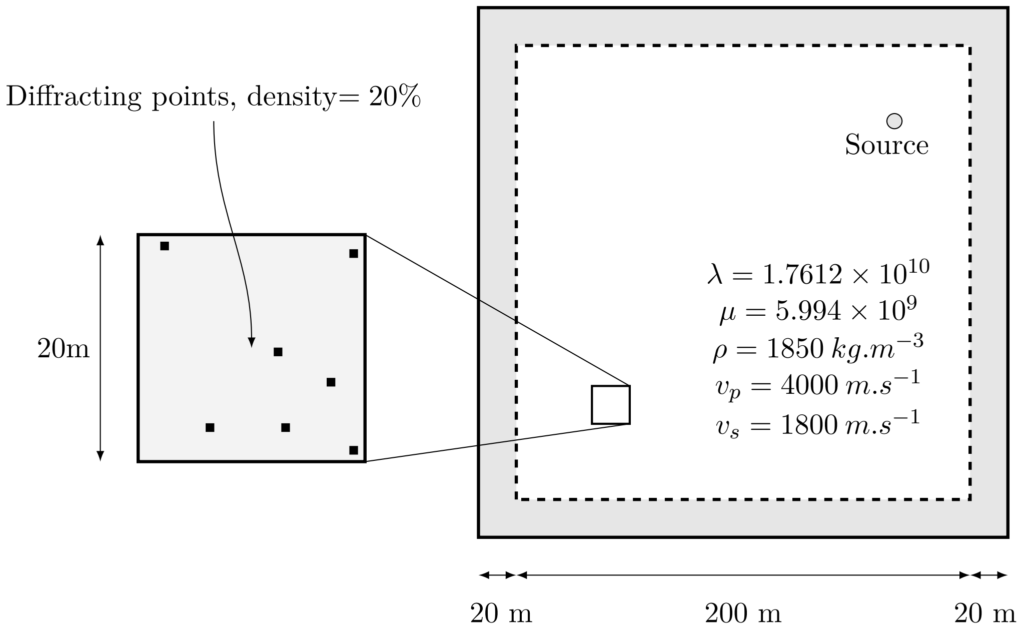

Diffracting points, density= 20%
20m
Source
$\lambda = 1.7612 \times 10^{10}$
$\mu = 5.994 \times 10^9$
$\rho = 1850\, kg.m^{-3}$
$v_p = 4000\, m.s^{-1}$
$v_s = 1800\, m.s^{-1}$
20 m
200 m
20 m

Figure 4.15: Diffracting points medium characteristics.

We display on Figure 4.16, 4.17 and 4.18 snapshots of X and Y displacements. We note a similar reflection as in the previous experiment, the amplitude is similar (around 10% of the primary front), however we do not have the resonance effect we had in the previous experiment.

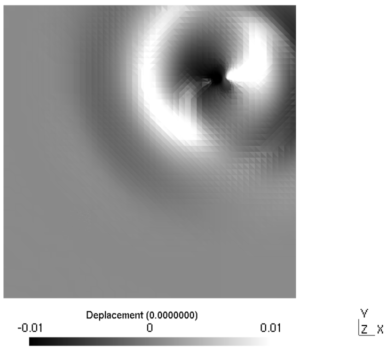
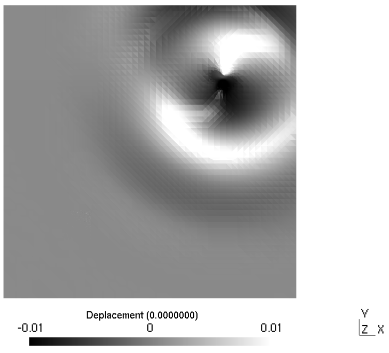
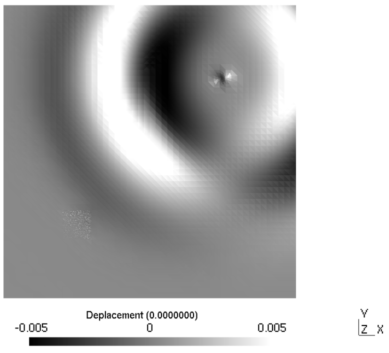
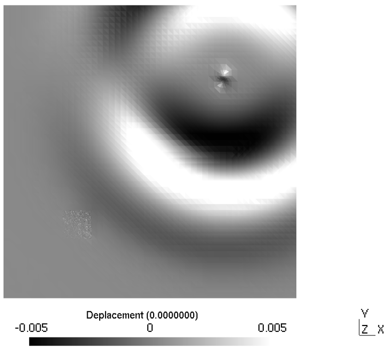
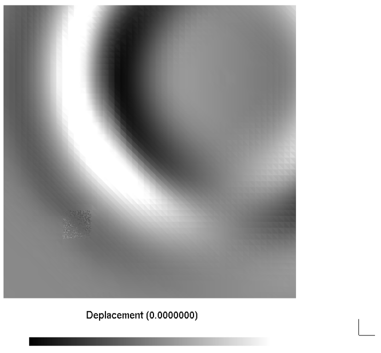
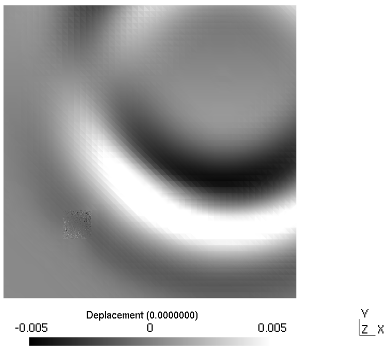
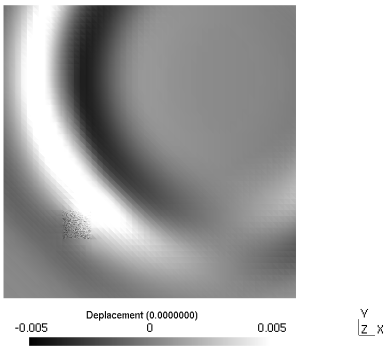
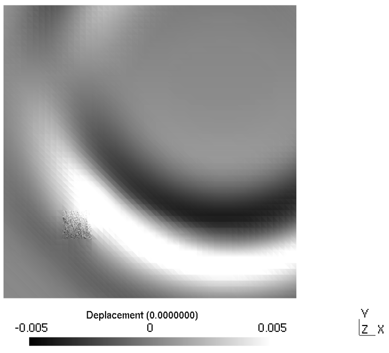
Time	X displacement	Y displacement
$0.025s$		
$0.03125s$		
$0.0375s$		
$0.04375s$		

Figure 4.16: Snapshots at different times of X and Y displacement for a medium with diffracting points.

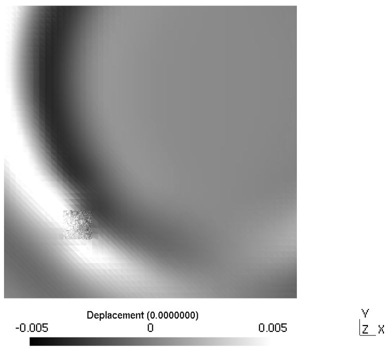
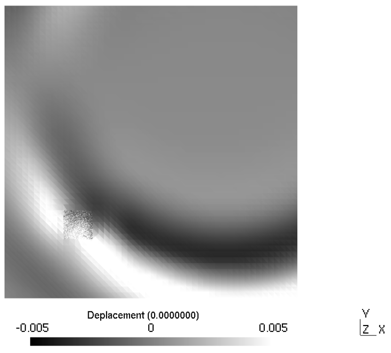
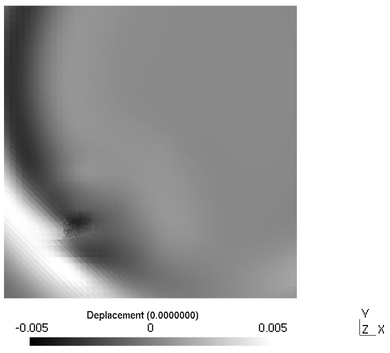
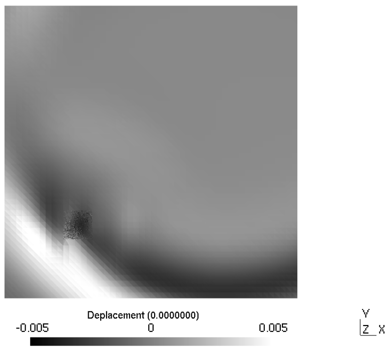
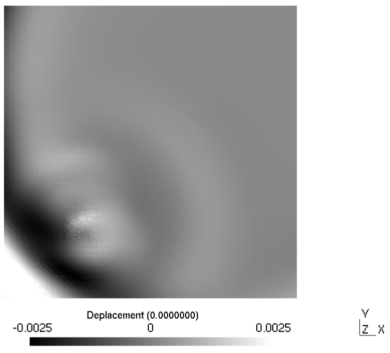
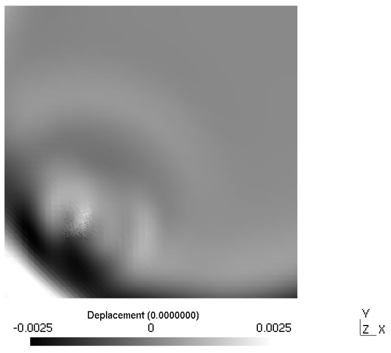
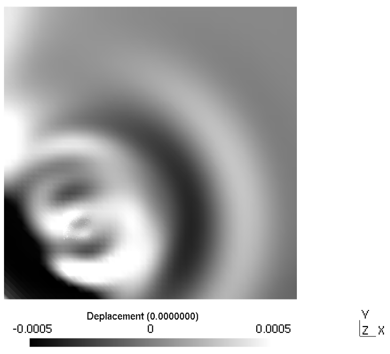
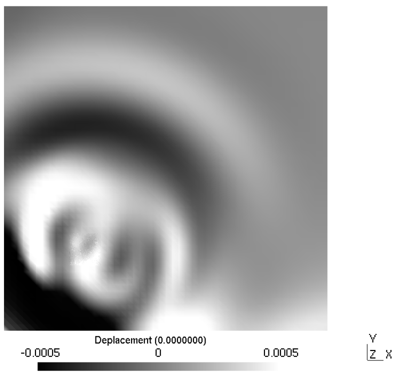
Time	X displacement	Y displacement
0.05s		
0.05625s		
0.0625s		
0.06875s		

Figure 4.17: Snapshots at different times of X and Y displacement for a medium with diffracting points.

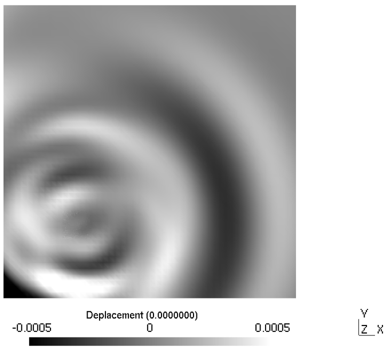
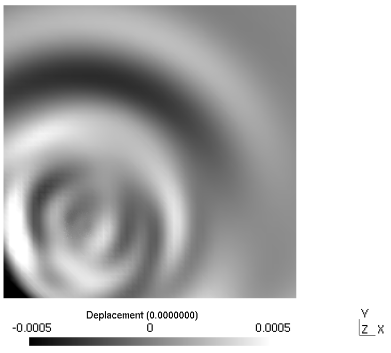
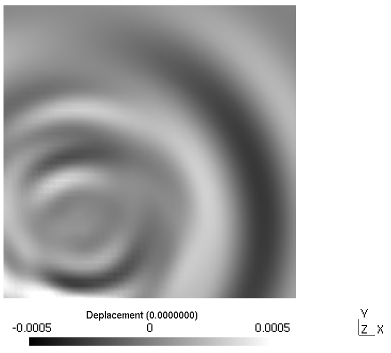
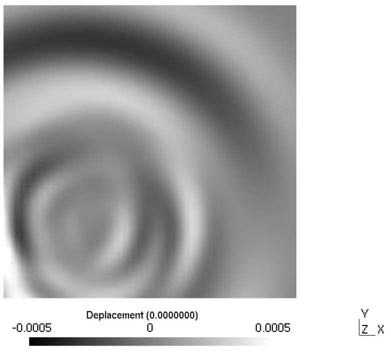
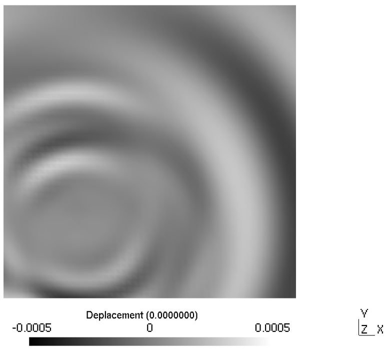
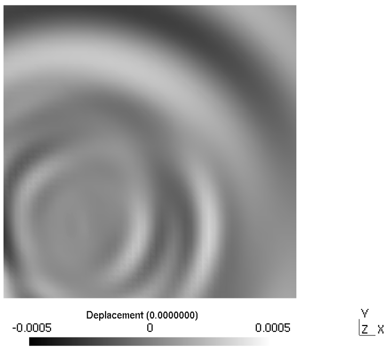
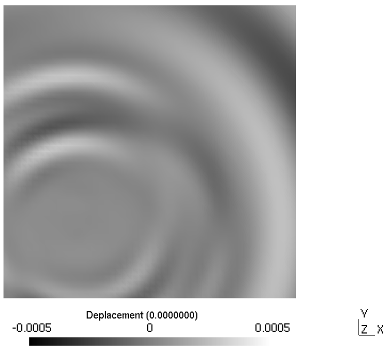
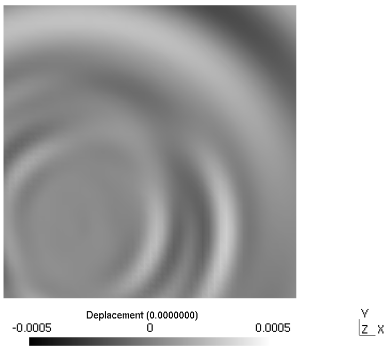
Time	X displacement	Y displacement
$0.075s$		
$0.08125s$		
$0.0875s$		
$0.09375s$		

Figure 4.18: Snapshots at different times of X and Y displacement for a medium with diffracting points.

4.5.3 Corridor of hydrofractures

In this third illustrative experiment we want to show how a corridor of hydrofractures can have a great impact on the simulation. In realistic cases, the orientation of the corridor, the distance between the hydrofractures has a great impact on the resulting phenomenon. Our purpose here is to show the kind of details our method can handle.

We take the same homogeneous medium as in the previous test case with the characteristics: $\lambda = 1.7612 \times 10^{10}$, $\mu = 5.994 \times 10^9$, $\rho = 1850\, kg.m^{-3}$, $v_p = 4000\, m.s^{-1}$ and $v_s = 1800\, m.s^{-1}$. The hydrofractures are $0.16m$ wide and $20m$ long and spaced of $0.48m$. The source is a pressure regularized Ricker of central frequency $40Hz$ positioned in $(150m, 150m)$.

We used Q_3 elements, with a space step of $4m$. The refined area is spatially refined by a factor $p_s = 25$ with Q_1 elements, and temporally refined by a factor $p_t = 10$. We added $20m$ of PML around our domain.

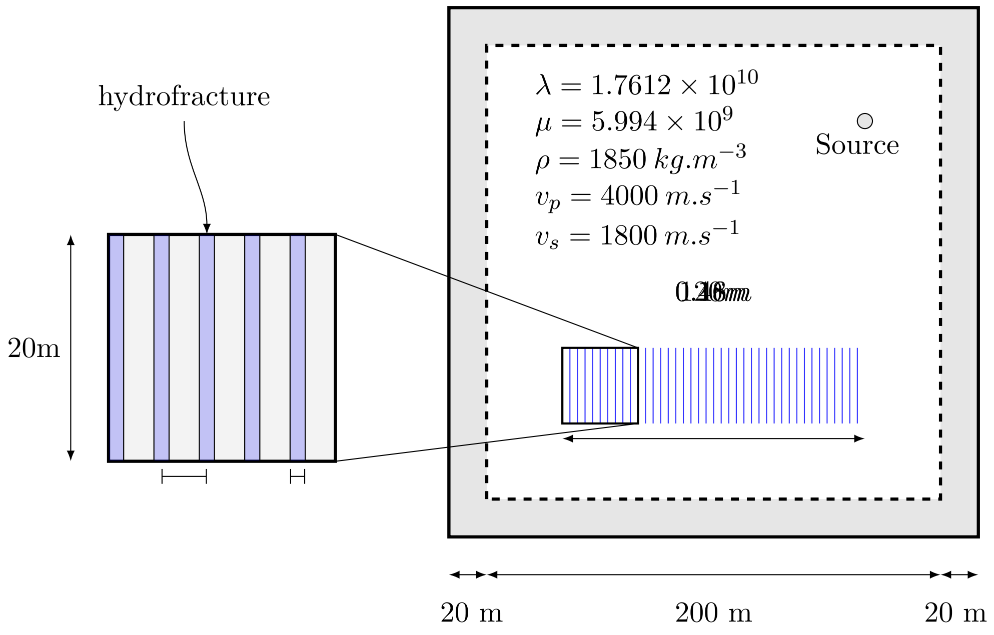


Figure 4.19: Corridor of hydrofractures medium characteristics.

We display on Figure 4.20 and 4.21 snapshots of this simulation for X and Y displacements.

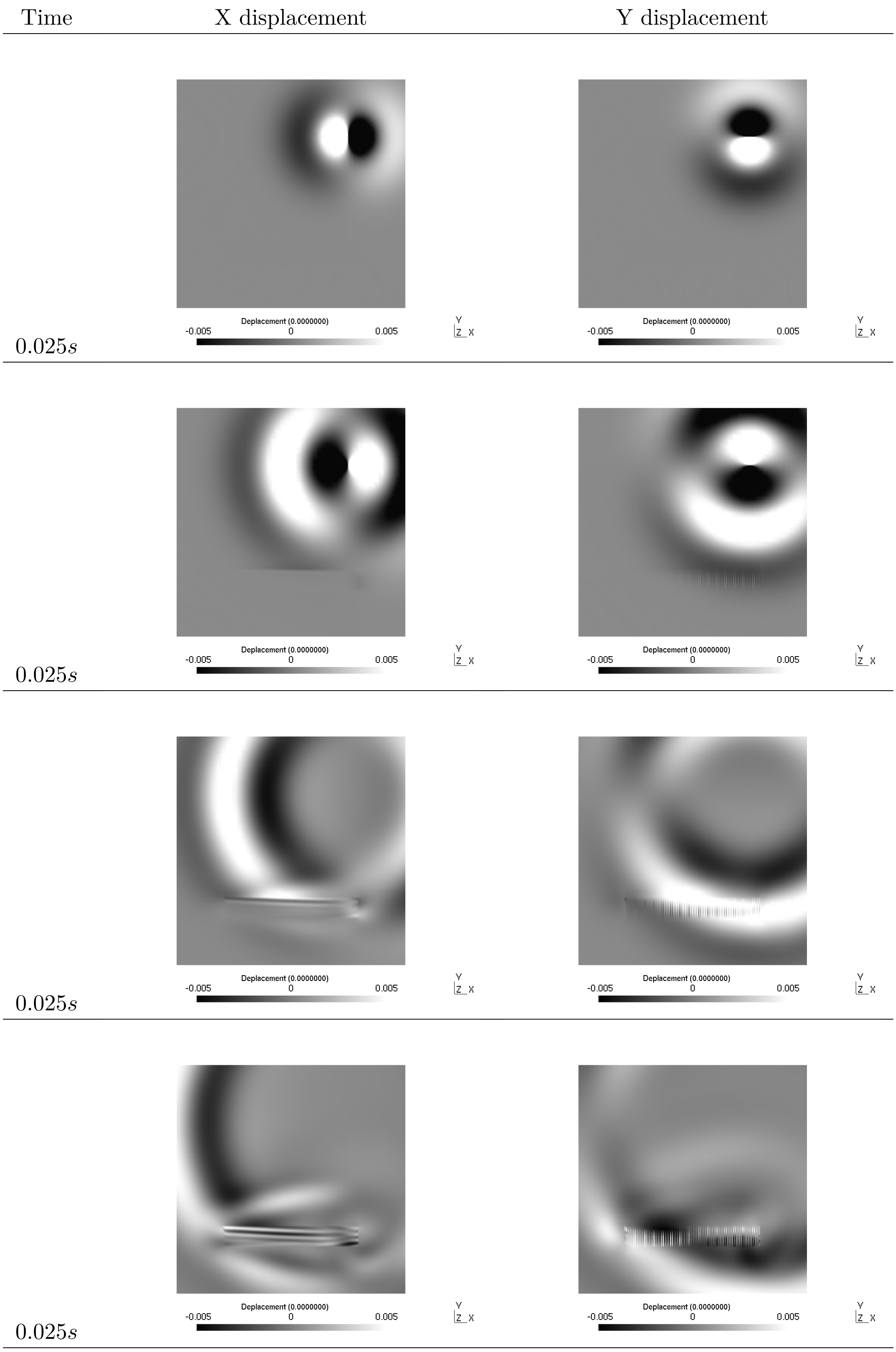


Figure 4.20: Snapshots at different times of X and Y displacement for a medium with a network of fluid-filled cracks.

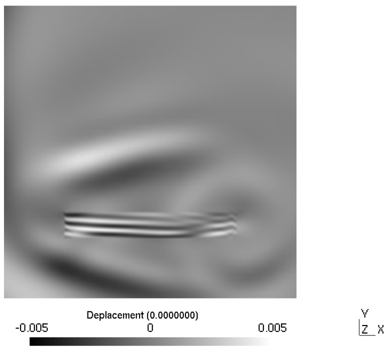
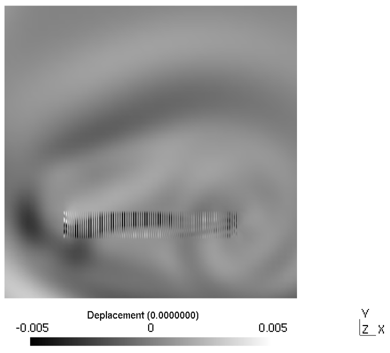
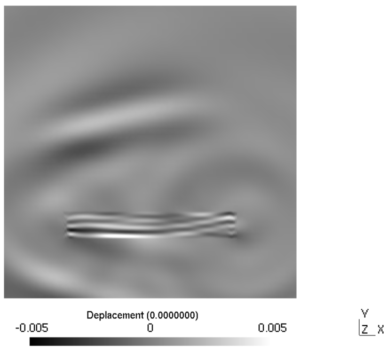
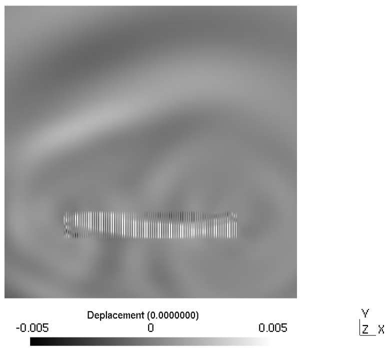
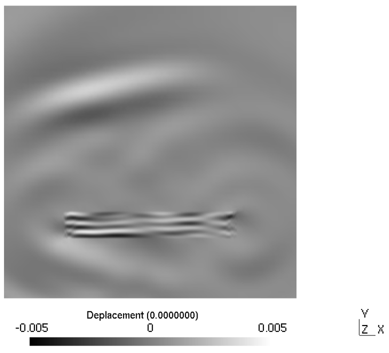
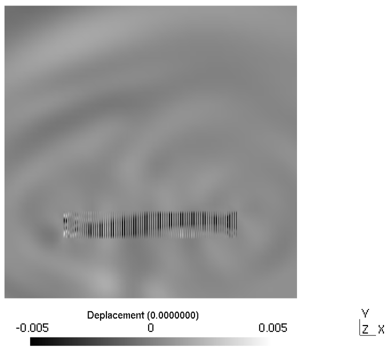
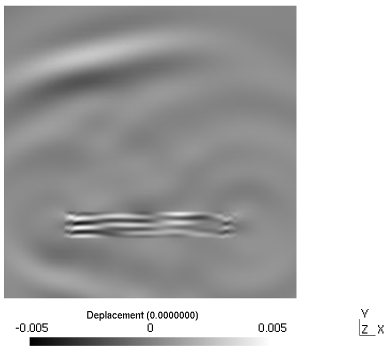
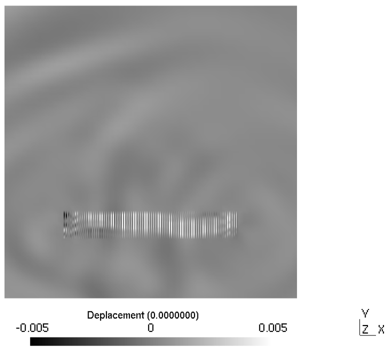
Time	X displacement	Y displacement
0.025s		
0.025s		
0.025s		
0.025s		

Figure 4.21: Snapshots at different times of X and Y displacement for a medium with a network of fluid-filled cracks.

4.6 Conclusion

In this chapter, we validated our approach to treat local elasto-acoustic heterogeneities of different size. We also gave illustrative examples to give an idea of the phenomena that can be observed with our method.

Chapter 5

Implementation and parallelization

In this chapter we describe our implementation of the methods introduced in the previous chapters, and emphasize on the specificities related to the non-conforming block Cartesian meshes. We also introduce our approach for parallelizing such methods.

To implement these algorithms we made an extensive use of the template oriented linear algebra library Eigen [**?**]. The fact that this library uses extensively template metaprogramming [**?**] makes the code especially readable and maintainable. The expression templates used grants effortless compilation time optimized code competing with the best linear algebra libraries. Our code is also highly based on template metaprogramming allowing relatively extensible programming, we especially used *policies* and *traits* metaprogramming concepts [**?**].

One of our earliest concern when implementing our methods was to exploit computationally the prerequisite of our context, *i.e.* the Cartesian grid, and also to exploit the most of the different features offered by DG methods. This led us to keep structured meshes and to write algorithms that attempt to exploit matrix-matrix operations (also called BLAS-3 operations) to get high computing rates.

Our sequential approach also drove our parallel design. We needed to keep structured partition to be consistent with the sequential approach. This led us to consider rectangular subdomains as granularity for parallelism. We decided to have a strategy where we create much more subdomains than we have computational resources, this allows flexibility for the load balancing. Indeed, load balancing can be cumbersome due to coarse and fine subdomains having inherently highly different computational costs. Thus, having many subdomains allows smaller granularity for load balancing.

The outline of this chapter is the following. In the Section 5.1, we introduce the way to compute the local DG matrices in our Cartesian grid case both for conforming and non-conforming meshes. Then, we introduce the data structures and the sequential algorithms we used. In the Section 5.2, we introduce in a first time our parallelization models and strategies. Finally, we give some performances and scalability results for distributed, shared and hybrid memory parallel architectures.

5.1 Implementing the discontinuous Galerkin methods

5.1.1 Local matrices

When implementing the DG methods, one has to compute integrals over volumes and faces. It would be too costly to compute the integrals over each physical element in the mesh. A more economical and effective approach is to use a change of variables to obtain an integral on a fixed element, called the reference element. As is done in the classical finite element methods, each mesh element K (also called physical element) is mapped to a reference element $\hat{K}$, and all computations are performed on the reference element. The aim of the reference element is to achieve all computation regardless of the shape of the physical element. However, with DG methods we potentially have several reference elements and "reference faces" due to the *hp*-adaptivity.

Let $\hat{\mathbf{x}} \in \hat{K}$ and $\mathbf{x} \in K$, we have a mapping of the form

$$F_K(\hat{\mathbf{x}}) = \mathbf{x}.$$

In our Cartesian case, we can rewrite this relation as

$$\mathbf{x} = B_K\hat{\mathbf{x}} + \mathbf{b}_K = \begin{pmatrix} h_K & & 0 \\ & \ddots & \\ 0 & & h_K \end{pmatrix} \hat{\mathbf{x}} + \mathbf{b}_K,$$

where $\mathbf{b}_K$ is a vector mapping the origin of the element.

Let $\{\hat{\varphi}_i\}i_{1\leq i\leq N_{\hat{K}}}$ be a basis of $V_h(\hat{K})$. Then, we have the following relations between the basis of K and $\hat{K}$

$$\varphi_i^K = \hat{\varphi}_i \circ F_K^{-1}.$$

Hence, we have

$$\forall \mathbf{u} \in V_h, \quad \int_K \mathbf{u} = \int_{\hat{K}} \det(B_K)\mathbf{u} \circ F_K = |K| \int_{\hat{K}} \mathbf{u} \circ F_K,$$

where $|K| = h_K^d$ and $|F| = h_K^{d-1}$.

We also have the following relation for partial derivatives

$$\frac{\partial \varphi_i^K}{\partial x_j} = \frac{1}{h}\frac{\partial \hat{\varphi}_i}{\partial \hat{x}_j}.$$

We shall now use these relations to compute the local matrices. In order to do so, we decompose the local spatial operator a_h^K define in Equation (1.31) in two operators $a_h^{K,\lambda}$ and $a_h^{K,\mu}$ such that

$$a_h^K = \lambda_K a_h^{K,\lambda} + \mu_K a_h^{K,\mu}.$$

Hence, we get the relation to calculate K^K

$$\begin{aligned} K_{ij}^K =& a_h^K(\varphi_j, \varphi_i) \\ =& h^{d-2}\left(\lambda_K a_h^{\hat{K},\lambda}(\hat{\varphi}_j, \hat{\varphi}_i) + \mu_K a_h^{\hat{K},\mu}(\hat{\varphi}_j, \hat{\varphi}_i)\right). \end{aligned}$$

We define

$$\hat{K}_{\lambda,ij} = a_h^{\hat{K},\lambda}(\hat{\varphi}_j, \hat{\varphi}_i), \quad \hat{K}_{\mu,ij} = a_h^{\hat{K},\mu}(\hat{\varphi}_j, \hat{\varphi}_i).$$

Thus, we obtain

$$K^K = h^{d-2}(\lambda_K \hat{K}_\lambda + \mu_K \hat{K}_\mu).$$

Similarly, we get the following relations for the flux matrices

$$F^{V_f(K)} = h^{d-2}(\lambda_{V_f(K)} \hat{F}^f_\lambda + \mu_{V_f(K)} \hat{F}^f_\mu),$$

where

$$\hat{F}^f_{\lambda,ij} = a_h^{K,\lambda}(\hat{\varphi}^f_j, \hat{\varphi}_i), \quad \hat{F}^f_{\mu,ij} = a_h^{K,\mu}(\hat{\varphi}^f_j, \hat{\varphi}_i),$$

and

$$M^K = \rho_K h^d \hat{M},$$

where

$$\hat{M}_{ij} = \langle \hat{\varphi}_j, \hat{\varphi}_i \rangle.$$

5.1.1.1 Non-conforming local matrices

Computing integrals over non-conforming faces is more tricky. Fortunately, only flux matrices $F^{V_f(K)}$ change, except for elasto-acoustic interfaces (Section 4.4). In the case of elasto-acoustic interfaces, volume and face integrals have to be separated and cannot be assembled in the same reference matrices $\hat{K}_\lambda$ and $\hat{K}_\mu$.

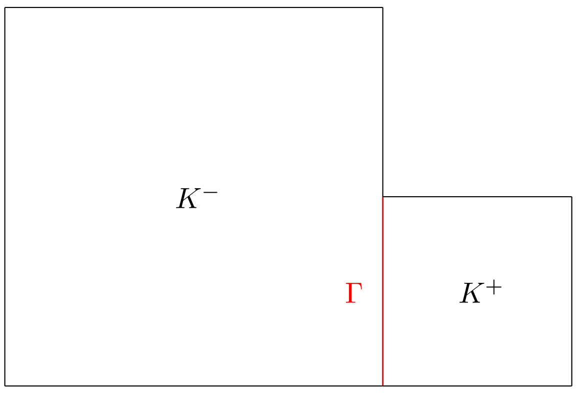


Figure 5.1: Two non-conforming elements.

The three different kinds of term that appear in flux integrals are

$$\begin{aligned}\int_\Gamma \varphi_j^+ \cdot \sigma(\varphi_i^-) \cdot n_\Gamma^- \, d\sigma &= \int_{\hat{\Gamma}} |\Gamma| \varphi_j^+ \circ F_+ \cdot \sigma(\varphi_i^-) \circ F_+ \cdot n_{\hat{\Gamma}}^- \, d\sigma \\ &= \int_{\hat{\Gamma}} |\Gamma| \hat{\varphi}_j^+ \cdot \sigma(\varphi_i^-) \circ F_- \circ (F_-^{-1} \circ F_+) \cdot n_{\hat{\Gamma}}^- \, d\sigma \\ &= \int_{\hat{\Gamma}} |\Gamma| \hat{\varphi}_j^+ \cdot \frac{1}{h_{K^-}} \hat{\sigma}(\hat{\varphi}_i^-) \circ (F_-^{-1} \circ F_+) \cdot n_{\hat{\Gamma}}^- \, d\sigma \\ &= \frac{|\Gamma|}{h_{K^-}} \int_{\hat{\Gamma}} \hat{\varphi}_j^+ \cdot \hat{\sigma}(\hat{\varphi}_i^-) \circ (F_-^{-1} \circ F_+) \cdot n_{\hat{\Gamma}}^- \, d\sigma,\end{aligned}$$

and

$$\begin{aligned}\int_\Gamma \varphi_j^- \cdot \sigma(\varphi_i^+)\cdot n_\Gamma^+ \, d\sigma =& \int_{\hat\Gamma} |\Gamma| \varphi_j^- \circ F_+ \cdot \sigma(\varphi_i^+)\circ F_+ \cdot n_{\hat\Gamma}^+ \, d\sigma \\ =& \int_{\hat\Gamma} |\Gamma| \hat\varphi_j^- \circ F_- \circ (F_-^{-1}\circ F_+) \cdot \sigma(\varphi_i^+)\cdot n_{\hat\Gamma}^+ \, d\sigma \\ =& \int_{\hat\Gamma} |\Gamma| \hat\varphi_j^- \circ (F_-^{-1}\circ F_+) \cdot \frac{1}{h_{K^+}} \hat\sigma(\hat\varphi_i^+)\cdot n_{\hat\Gamma}^+ \, d\sigma \\ =& \frac{|\Gamma|}{h_{K^+}} \int_{\hat\Gamma} \hat\varphi_j^- \circ (F_-^{-1}\circ F_+) \cdot \hat\sigma(\hat\varphi_i^+)\cdot n_{\hat\Gamma}^+ \, d\sigma,\end{aligned}$$

and

$$\begin{aligned}\int_\Gamma \varphi_j^- \cdot \varphi_i^+ \, d\sigma =& \int_{\hat\Gamma} |\Gamma| \varphi_j^- \circ F_+ \cdot \varphi_i^+ \circ F_+ \, d\sigma \\ =& \int_{\hat\Gamma} |\Gamma| \hat\varphi_j^- \circ F_- \circ (F_-^{-1}\circ F_+) \cdot \varphi_i^+ \, d\sigma \\ =& \int_{\hat\Gamma} |\Gamma| \hat\varphi_j^- \circ (F_-^{-1}\circ F_+) \cdot \hat\varphi_i^+ \, d\sigma \\ =& |\Gamma| \int_{\hat\Gamma} \hat\varphi_j^- \circ (F_-^{-1}\circ F_+) \cdot \hat\varphi_i^+ \, d\sigma.\end{aligned}$$

In the two dimensional case, with $h_{K^+} = \dfrac{h_{K^-}}{p_s}$ and $|\Gamma| = h_{K^+}$, for $k \in [0, p_s - 1]$ we have the following relations (Figure 5.1 corresponds to $p_s = 2$ and $k = 0$) to calculate the integral over a face as shown on Figure 5.1

$$\int\limits_\Gamma \varphi_j^+ \cdot \sigma(\varphi_i^-)\cdot n_\Gamma \, d\sigma = \frac{1}{p_s} \int\limits_0^1 \hat\sigma(\hat\varphi_i^-(1, \frac{k+\theta}{p})) \cdot n_\Gamma \cdot \hat\varphi_j^+(0,\theta)\, d\theta,$$

and

$$\int\limits_\Gamma \varphi_j^- \cdot \sigma(\varphi_i^+)\cdot n_\Gamma \, d\sigma = \int\limits_0^1 \hat\sigma(\hat\varphi_i^+(0,\theta)) \cdot n_\Gamma \cdot \hat\varphi_j^-(1, \frac{k+\theta}{p})\, d\theta,$$

and

$$\begin{aligned}\int_\Gamma \varphi_j^- \cdot \varphi_i^+ \, d\sigma =& h_{K^+} \int\limits_0^1 \hat\varphi_i^-(1, \frac{k+\theta}{p}) \cdot \hat\varphi_j^+(0,\theta)\, d\theta \\ =& \frac{h_{K^-}}{p_s} \int\limits_0^1 \hat\varphi_i^-(1, \frac{k+\theta}{p}) \cdot \hat\varphi_j^+(0,\theta)\, d\theta.\end{aligned}$$

Remark 5.1. *For each $k \in [0, p_s - 1]$ the above integrals are unfortunately different, thus the number of local matrices is multiplied by p_s in two dimensions, and by p_s^2 in three dimensions. If the memory becomes an issue, considering nested refinements,* e.g. *two refinements by $p_s = 10$ instead of one by $p_s = 100$, can be a solution.*

In the three dimensional case, with $h_{K^+} = \dfrac{h_{K^-}}{p_s}$ and $|\Gamma| = h_{K^+}^2$, for $k_1, k_2 \in [0, p_s - 1]$ we have the following relations

$$\begin{aligned}\int\limits_{\Gamma} \varphi_j^+ \cdot \sigma(\varphi_i^-) \cdot n_\Gamma \, d\sigma &= \frac{h_{K^+}}{p_s} \int\limits_0^1 \int\limits_0^1 \hat{\sigma}(\hat{\varphi}_i^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s})) \cdot n_\Gamma \cdot \hat{\varphi}_j^+(0,\theta_1,\theta_2) \, d\theta_1 d\theta_2 \\ &= \frac{h_{K^-}}{p_s^2} \int\limits_0^1 \int\limits_0^1 \hat{\sigma}(\hat{\varphi}_i^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s})) \cdot n_\Gamma \cdot \hat{\varphi}_j^+(0,\theta_1,\theta_2) \, d\theta_1 d\theta_2,\end{aligned}$$

and

$$\begin{aligned}\int\limits_{\Gamma} \varphi_j^- \cdot \sigma(\varphi_i^+) \cdot n_\Gamma \, d\sigma &= h_{K^+} \int\limits_0^1 \int\limits_0^1 \hat{\sigma}(\hat{\varphi}_i^+(0,\theta_1,\theta_2)) \cdot n_\Gamma \cdot \hat{\varphi}_j^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s}) \, d\theta_1 d\theta_2 \\ &= \frac{h_{K^-}}{p_s} \int\limits_0^1 \int\limits_0^1 \hat{\sigma}(\hat{\varphi}_i^+(0,\theta_1,\theta_2)) \cdot n_\Gamma \cdot \hat{\varphi}_j^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s}) \, d\theta_1 d\theta_2,\end{aligned}$$

and

$$\begin{aligned}\int_{\Gamma} \varphi_j^- \cdot \varphi_i^+ \, d\sigma =& h_{K^+}^2 \int\limits_0^1 \int\limits_0^1 \hat{\varphi}_i^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s}) \cdot \hat{\varphi}_j^+(0,\theta_1,\theta_2) \, d\theta_1 d\theta_2 \\ =& \frac{h_{K^-}^2}{p_s^2} \int\limits_0^1 \int\limits_0^1 \hat{\varphi}_i^-(1, \frac{k_1+\theta_1}{p_s}, \frac{k_2+\theta_2}{p_s}) \cdot \hat{\varphi}_j^+(0,\theta_1,\theta_2) \, d\theta_1 d\theta_2.\end{aligned}$$

5.1.2 Data structure

Since we are using Cartesian grids we wanted to keep structured meshes. Unfortunately, the refined areas disrupt this structured aspect. The solution we chose to overcome this issue was to have local unstructured meshes between coarse and fine grids leading to hybrid meshes. To preserve the regular data structure, we decided to keep the unnecessary coarse elements in the refined area in order to preserve the structured indexing of the coarse grid. We call these elements ghost elements (see Figure 5.3). Therefore, unnecessary computation is performed on these ghost elements. Since the locally refined areas should be of limited size, the extra cost of the ghost elements is relatively small. Moreover, having a completely unstructured mesh would cost more both in computation and memory.

The data structure for the coarse and the fine grids are thus matrices of size $nb_{dof} \times nb_{elts}$, where nb_{dof} is the number of degrees of freedom per element of the corresponding grid and nb_{elts} the number of elements of the corresponding grid. If there is N_X and N_Y elements in the direction X and Y respectively ($nb_{elts} = N_X \times N_Y$), and if we denote by $ind(i,j)$ the index of an element at the position (i,j) on the Cartesian grid. Then, we have the standard structured relations

$$\begin{aligned} ind(i+1,j) =& ind(i,j)+1, \\ ind(i-1,j) =& ind(i,j)-1, \\ ind(i,j+1) =& ind(i,j)+N_X, \\ ind(i,j-1) =& ind(i,j)-N_X. \end{aligned}$$

The data structure for the halo is a standard unstructured data structure. Each element stores the indices of its neighboring elements. Besides, each element has a tag to specify if it is coarse, fine, halo-coarse or halo-fine element (see Section 3.2.2.4). Thus we have matrices containing all degrees of freedom of size $nb_{dof} \times nb_{elts}$, where nb_{elts} is the number of elements in the halo, and a matrix of size $nb_{dof} \times nb_{hf}$ to store the vectors w^K described in Algorithm 3.2, where nb_{hf} is the number of halo-fine elements.

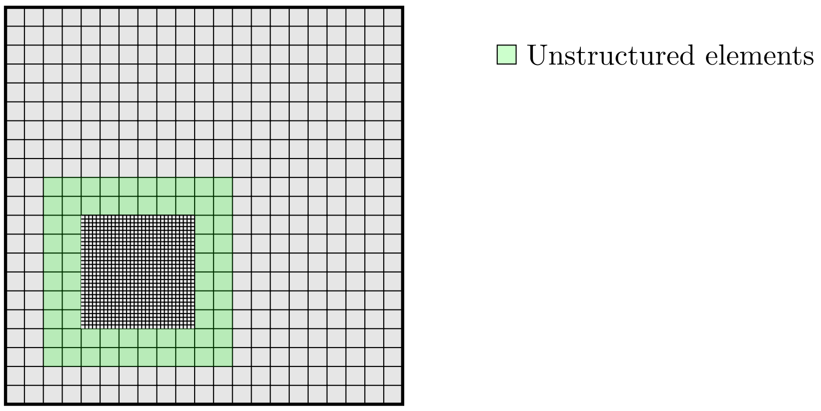


Figure 5.2: Representation of a refined mesh.

This data structure choice brings many implementation difficulties. In particular, the unstructured mesh linking the coarse and the fine grids is what we call the halo in Section 3.2.2.4. Thus all the algorithmic complexity of the local time stepping method happens in this unstructured part. Finally, we obtain a data structure, composed of three substructures, that corresponds to the three algorithms we described in Section 3.2.2.5.

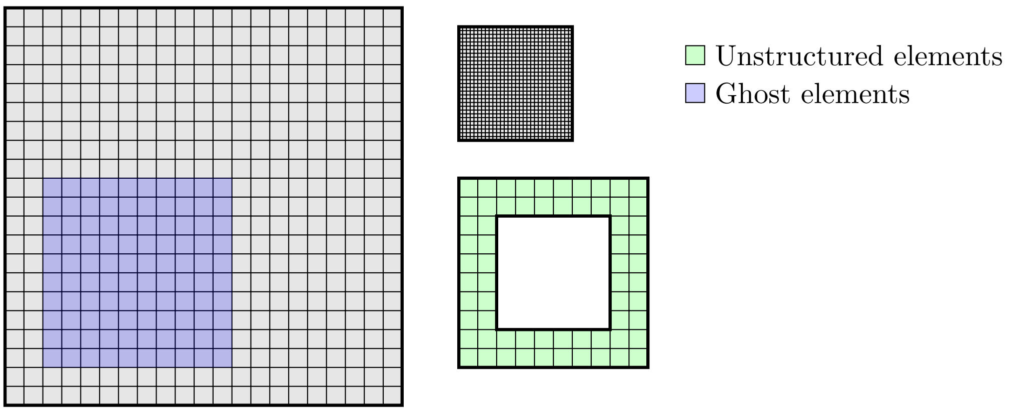


Figure 5.3: Representation of the different structures.

Implementing the algorithm for the coarse grid and the fine grids is straightforward. However, the unstructured halo concentrates all the difficulties:

- For the purpose of the local time stepping algorithm, each element of the halo has to be tagged as halo-coarse or halo-fine as described in Section 3.2.2.4;

- Halo-fine element data structure needs to be duplicated in order to store fluxes coming from halo-coarse elements (vector $\mathbf{w}^K$ in Algorithm 3.2);

- We need two indirection tables for the exchange of fluxes between the halo and the coarse grid, and between the halo and the fine grid;

- The fluxes between the halo and the fine grid are fluxes on non-conforming elements.

5.1.3 Computing the spatial DG approximation

Since we have structured data, we wanted to exploit this property to perform matrix-matrix multiplications (also called BLAS-3 operations) which are really computationally efficient. This led us to rearrange the order in which the calculations are generally performed. Usually, for each element we assemble the local matrices from the reference matrices and we compute the different contributions of this element, leading to the following pseudo-algorithm:

Algorithm 5.1 Standard DG algorithm.

```
for each element $K \in \mathcal{T}_h$ do
  Compute local volume matrix $K^K$
  Compute local face matrices $\forall f \in \mathcal{F}_K,\ F^{V_f(K)}$
  Compute $\tilde{\mathbf{u}}^K = K^K \mathbf{u}_n^K + \sum_{f \in \mathcal{F}_K} F^{V_f(K)} \mathbf{u}_n^{V_f(K)}$
  Update $\mathbf{u}^K$: $\mathbf{u}_{n+1}^K = 2\mathbf{u}_n^K - \mathbf{u}_{n-1}^K + \frac{\Delta t^2}{\rho_K h_K^2} \hat{M}^{-1} \tilde{\mathbf{u}}$
end for
```

Now we introduce another algorithm, using as much as possible matrix-matrix products. We assume that the degrees of freedom are arranged in a matrix such that $U_n := \left(u_n^{K_1} \cdots u_n^{K_N}\right)$.

Algorithm 5.2 Matrix-matrix oriented DG algorithm.

```
$\tilde{U} = 0$
for all local reference matrices $A \in \{\hat{K}_\lambda, \hat{K}_\mu, \hat{F}_\lambda^f, \hat{F}_\mu^f ...\}$ do
  Compute $U_{tmp} = \hat{M}^{-1} A U_n$
  Multiply each column of $U_{tmp}$ by the intended scalar ($e.g.$ $\frac{\lambda_K}{\rho_K h_K^2}, \frac{\mu_K}{\rho_K h_K^2}, ...$)
  if $A$ is a volume matrix ($e.g.$ $\hat{K}_\lambda, \hat{K}_\mu$) then
    $\tilde{U} = \tilde{U} + U_{tmp}$
  else
    $\tilde{U} = \tilde{U} + shift(f, U_{tmp})$
  end if
end for
Update $U$: $U_{n+1} = 2U_n - U_{n-1} + \Delta t^2 \tilde{U}$
```

The function $shift$ in the Algorithm 5.2 shift all the columns of U_{tmp} of $1, -1, N_X, -N_X$ according to the considered face f used to compute the flux.

Concerning the halo, we use an algorithm of the kind of the standard Algorithm 5.1 respecting the halo local time stepping Algorithm 3.2 due to the unstructured data structure. The fine element algorithm (Algorithm 3.3) can easily be adapted in the form of the Algorithm 5.2.

5.2 Parallelization

There are two main approaches to parallelize a code: shared and distributed memory parallelization. Shared memory parallelization works with thread using the same memory space, but concurrency between the threads appears since the shared data cannot be modified in the same time. This concurrency must be minimized so as not to reduce performances. Distributed memory parallelization works with processes exchanging messages, each process working with its own private memory. There are two important aspects to consider, the load balance and the amount of communications. The load balance is the way work is distributed between the processes, the more it is balanced the less processes wait for each other. The amount of communications is also important since the network has a limited bandwidth and possibly large latency. Ideally, processes overlap the communications with the computations.

5.2.1 Parallelization general ideas

The parallelization of the coarse and fine grids is straightforward, we partition the domains into rectangles (usually squares except for the PML). By doing so we can easily obtain subdomains that have the same computational load. The main issue was: how do we partition the halo. The strategy we decided to keep is to partition the halo in the continuity of the partitions of the fine grid, as shown on Figure 5.4 and 5.5. This choice makes the indirections between the coarse grid and the halo even more tedious to implement, since a refined area can overlap the coarse grid arbitrarily.

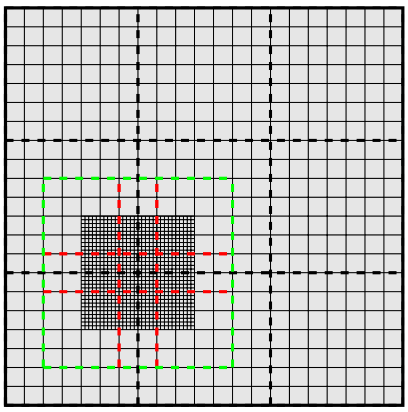


Figure 5.4: Representation of partitioning cutting lines.

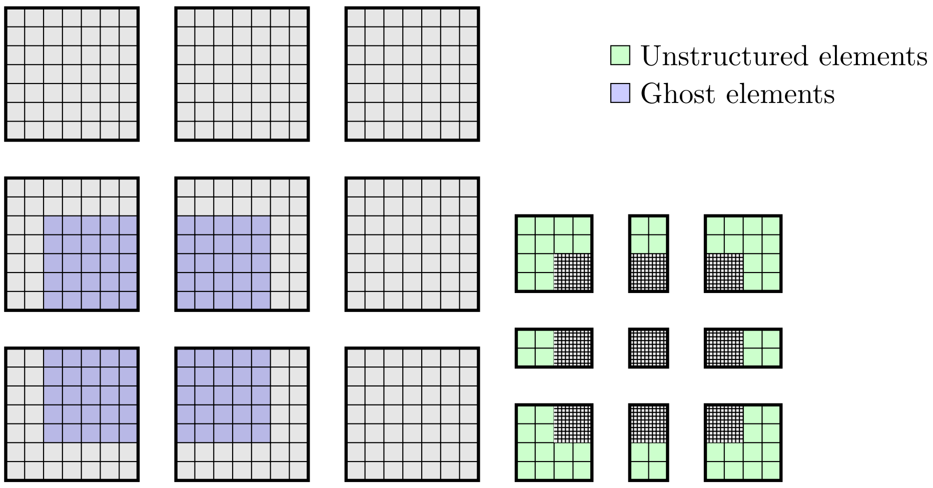


Figure 5.5: Representation of the different subdomains for parallelization.

The algorithm to apply for any of these subdomains, *i.e.* coarse, fine or halo, is exactly the same:

1. Compute all
2. Send boundary fluxes intended for other subdomains,
3. Receive boundary fluxes from surrounding subdomains,

as described in Figure 5.6.

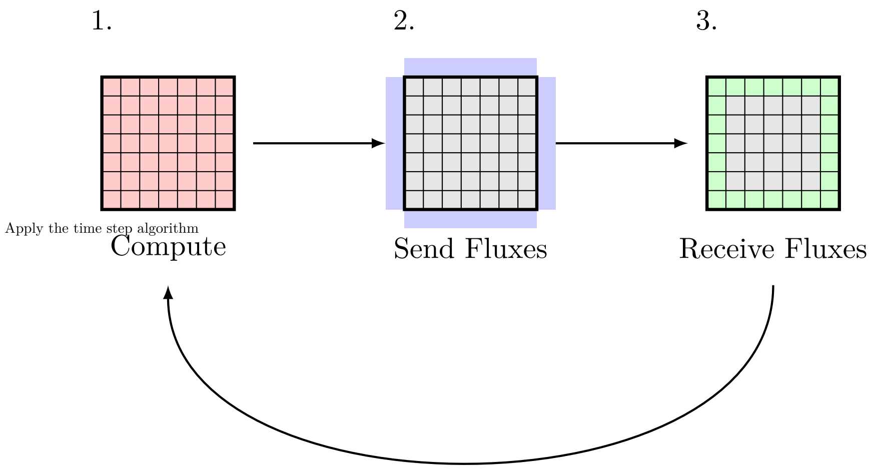


Figure 5.6: Execution diagram of a subdomain.

Computing, sending fluxes and updating subdomains can be performed asynchronously, however receiving fluxes is blocking the progress. Inspired by the *model-view-controller* software architectural pattern, we decided to use a *controller* that handles the fluxes between subdomains. As soon as a subdomain is waiting to receive fluxes, it goes in a set of *unready subdomains* waiting that the controller has received its fluxes. When the controller has received all fluxes for a subdomain, the controller moves this subdomain in a set of *ready subdomains.* These ideas allow a completely asynchronous execution. The Figure 5.7 sum up these ideas in a diagram.

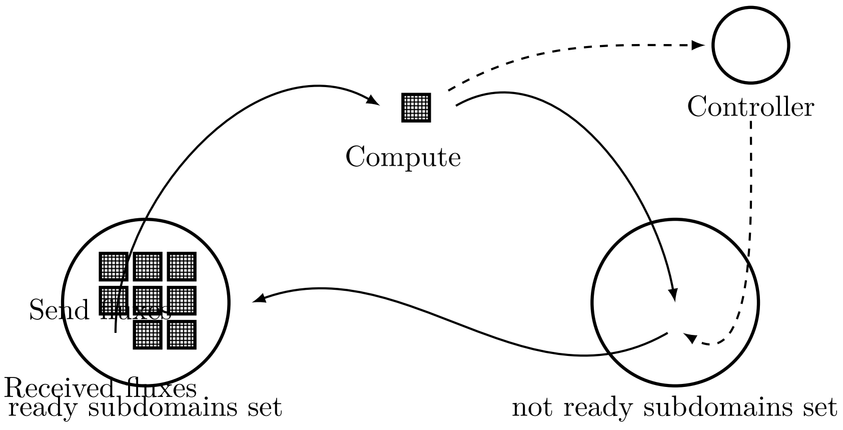


Figure 5.7: Subdomains live cycle.

Instead of having one subdomain per core, we preferred to have many smaller subdomains per process. By doing so, shared and distributed memory parallelism almost work the same. Moreover, the set of subdomains per process can contain subdomains of relatively different computational weights, what counts is the total weight which should be well balanced between processes. Now, we shall explain how we exploit these ideas in a shared and a distributed memory parallel context.

5.2.1.1 Shared memory parallelization

We investigated two different strategies to exploit shared memory parallelism. The first idea is to let each thread pick subdomains in the *ready subdomains set.* All threads send their fluxes to a unique controller as described by the diagram in Figure 5.8. We refer to this strategy as the *subdomain based strategy.* The second idea is to parallelize the *for* loop on the operators in the Algorithm 5.2. We refer to this strategy as the *operator based strategy.*

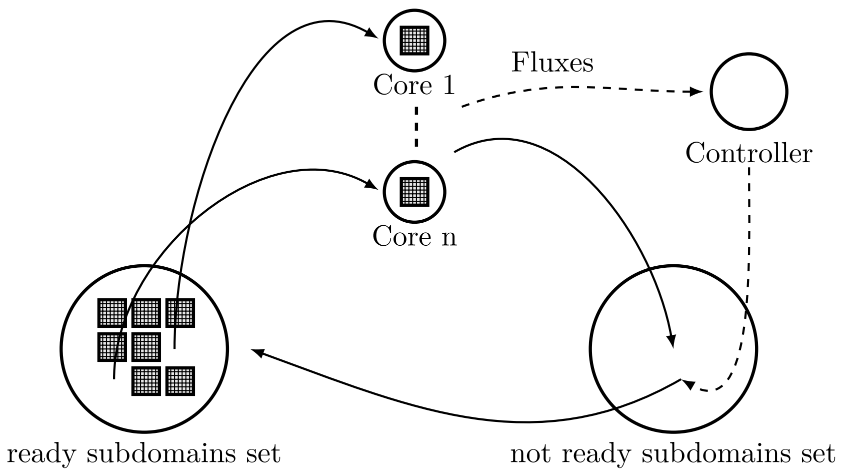


Figure 5.8: Shared memory parallelism diagram.

5.2.1.2 Distributed memory parallelization

The main idea to use distributed memory parallelism is to duplicate the structure described on Figure 5.7 on each each process. Each subdomain knows if its neighbors are distant or local, and if they are distant they send the fluxes to the suited distant controller as represented on Figure 5.9.

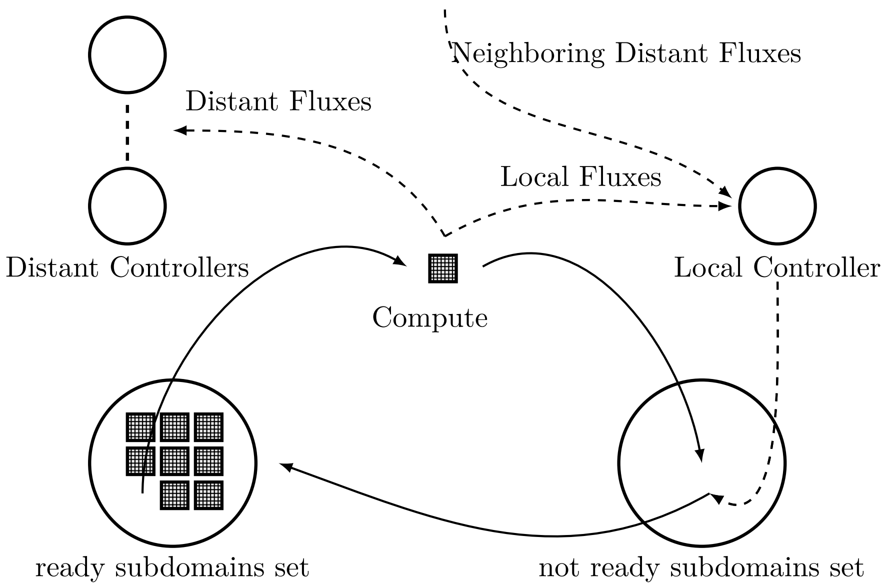


Figure 5.9: Distributed memory parallelism diagram.

5.2.2 Performances and scalability

Graph partitioning strategy: One of the most important aspect when having distributed memory parallelism is the load balancing and the minimization of the communication volume. We use a standard graph partitioning strategy. We associate with each subdomain a computational cost, which corresponds to the weights of the vertices of the graph. We also associate weights with the edges of the graph to represent the volume of communications between two subdomains since the communication volume between two subdomains is not always the same due to the local time stepping method. Once we have defined our graph we need to realize an n-cut, according to the number of MPI process we want. In graph theory, an n-cut is a partition of the vertices of a graph into n disjoint subsets. Any n-cut determines a cut-set, the set of edges that have one endpoint in two different subsets of the partition. In a distributed memory parallel context, this n-cut must be computed in order to have the same (or approximately) node weights in each subset of the partition, and a minimal weight the cut-set, or close to the minimum.

Remark 5.2. *In finite element methods it is more common to realize the partitioning on the elements rather than on subdomains since the shape of the subdomain has an important impact on the quantity of communications. However, in the case of Cartesian grids, square domains have an optimal cut size for graphs (the proof of this result is straightforward).*

Without PML or spatial refinements having vertices weights proportional to the size of the subdomains gives good load balancing. However, having non-structured, non-conforming meshes, different polynomial orders, different number of local matrices make accurate prediction of the computation cost of a subdomain really challenging, especially for PML and halo subdomains. This imposes to have efficient heuristics to evaluate the computational cost of each subdomain according to its specificities in order to ensure an effective partitioning.

The partition defines what we call local and distant communications. We call local communication any communication between two subdomains of the same partition. Similarly, we call distant communication any communication between two subdomains of different partitions. Typically, distant communications happen between MPI processes, and local communications happen between OpenMP threads.

To achieve the graph partitioning, we used the software METIS [**?**]. We give an example of the kind of graph we have to partition based on the subdomains displayed on Figure 5.5.

Warning about local time-space refinement: Choosing the right size for the subdomains is of great importance. The smaller the subdomains the higher is the cost of the communications between subdomains. However, at constant number of partitions, the volume of distant communication stay approximately the same, only the number of distant communications increases. If the bandwidth is saturated, tuning the size of the subdomains can be a solution.

Refined areas can be particularly cumbersome to have a good load balancing. Indeed, when refining an area the computational cost is at least multiplied by $p_t p_s^d$. This can quickly creates subdomains that carry most of the computational cost. For instance, if we assume that the computational cost of the coarse grid is 1. Then the cost of the refined area is $r p_t p_s^d$, where r is the proportion of the space which is refined. Then, we give in Table 5.1 and Table 5.2 the proportion of the total space such that the fine part has the same computational cost than the coarse grid. For instance, in two dimensions for a local refinement per $p_t = p_s = 10$ the volume of the refined area should be of 0.1% of the total volume in order to have approximately the same computation cost in the coarse and refined areas. This volume has to be reduced to 0.01% of the total volume in three dimensions. Fortunately, we can partition the refined area in several subdomains such that managing load balancing is still achievable. Nevertheless, refined subdomains can quickly have a computational cost way higher than other subdomains making load

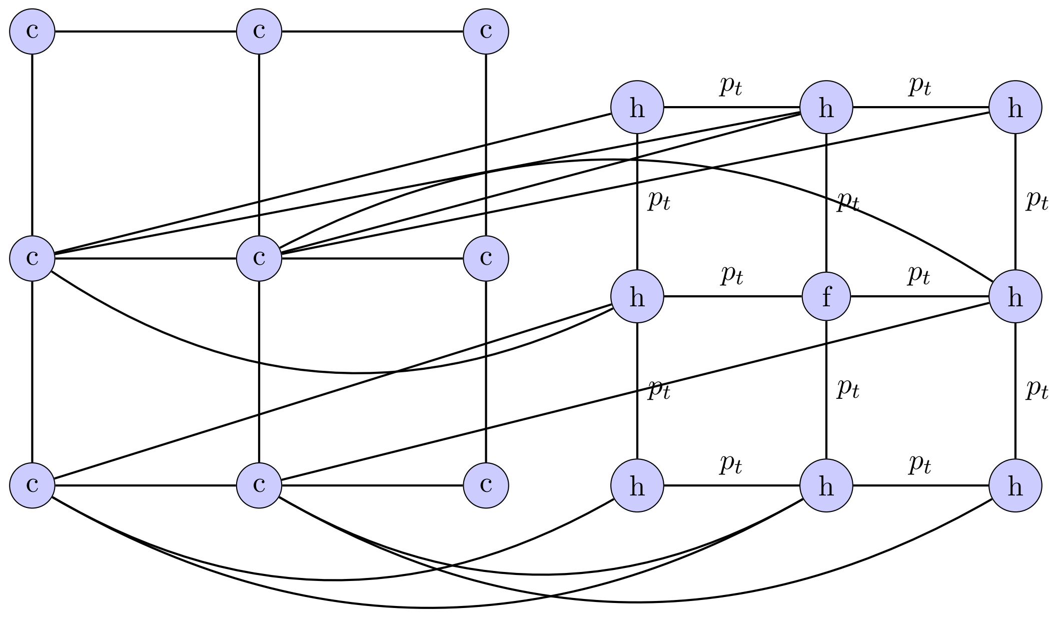


Figure 5.10: Graph representation of the partitioning of Figure 5.5.

balancing difficult or even impossible. Besides, the way we manage the partitioning of the refined area prevents the creation of subdomains smaller than the size of a coarse element. Note that this last limitation is due to the implementation choice to attach the halo to the corresponding fine subdomain (as represented on Figure 5.5).

$p_s = p_t$	r
2	12.5%
10	0.1%
20	0.0125%

Table 5.1: Proportion of the refined area such that the coarse grid and refined area have the same computational cost in two dimensions.

$p_s = p_t$	r
2	6.25%
10	0.01%
20	0.000625%

Table 5.2: Proportion of the refined area such that the coarse grid and refined area have the same computational cost in three dimensions.

Priority between tasks: The larger the number of subdomains, the easier it is to overlap communications with computation since most communications become local. However, we decided to implement a *priority* between the subdomains. Our strategy is to grant a higher priority to subdomains that have distant (MPI) communications to perform. The more distant communications the subdomains has, the higher is its priority in the ready subdomains set.

5.2.2.1 Overview of the computer

We give here a quick overview of the computer on which we ran our performance tests. Each of the 158 computing nodes has the following characteristics:

- two Intel Sandy Bridge processors (EP E5-2670) (8 cores 2.6 Ghz (8 flops per cycle per core is 330 GFlops / s peak performance per node)),
- 32 GB of memory per node (DDR3 memory clocked at 1600 MHz),
- L1 caches (instruction and data) 32 KB, 256 KB L2 cache per core,
- L3 cache 20 M0 shared by the 8 cores of each processor.

Infiniband interconnection network offers a bandwidth of 5 GB / s between nodes. The MPI latency is less than 1 microsecond. The installed operating system on the nodes is CentOS 6.2 or 6.2 RedhatEnterprise.

A maximum of 8 nodes per run could be taken, corresponding to 128 cores.

5.2.2.2 Impact of the size of the subdomains on performances

From an ideal point of view the size of the subdomains should not impact the sequential performances. However, many memory effects come into play. In order to show the performances according to the subdomain sizes and of the polynomial spaces Q_k we measured the performance in percentage of the peak for a domain composed of four subdomains achieving 1000 time steps. The results are displayed in Figure 5.11. We note that high order polynomial basis have better performances regardless of the size of the subdomains. We also note that the size of the subdomains influence less the performances, except for a moderate peak for sizes between 10×10 to 30×30 according to the polynomial basis order.

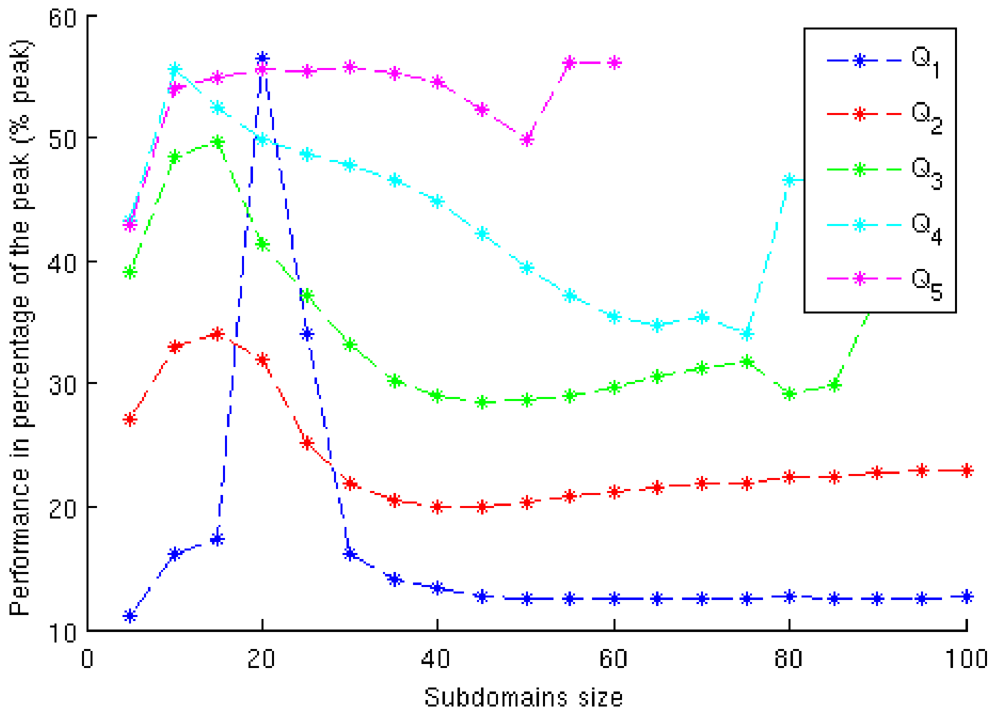


Figure 5.11: Performance in percentage of the peak according to the size of the subdomains.

5.2.2.3 MPI performances

We implemented our MPI communications with asynchronous non-blocking communications. This allows processes to continue computation right after they send their messages, thus hiding the communications as much as possible. Actually, we tried blocking communications and about 30% of the computation time was spent in waiting time to send and receive messages, whereas it is of less than 1% of the computation time with asynchronous non-blocking communications.

There are two common notions of performance scalability in the context of high performance computing:

- the *weak scalability*, which is defined as how the solution time varies with the number of processors for a fixed problem size *per processor*,
- the *strong scalability*, which is defined as how the solution time varies with the number of processors for a fixed *total* problem size.

Weak scalability: To perform the weak scalability tests, we give to each MPI process a set of 4×4 subdomains of size 20 made of Q_3 elements and we performed 1000 time steps. We report in Table 5.3 and display on Figure 5.12 the computing times for 1 to 128 MPI processes. We note that we have an almost perfect *weak scalability* since the computation times are almost constant from 1 to 128 MPI processes.

Number of MPI processes	Time (s)	Speed up
1	174	-
2	175	1.99
4	173	4.02
8	175	7.95
16	175	15.91
32	176	31.63
64	178	62.56
128	179	124.42

Table 5.3: Weak scalability.

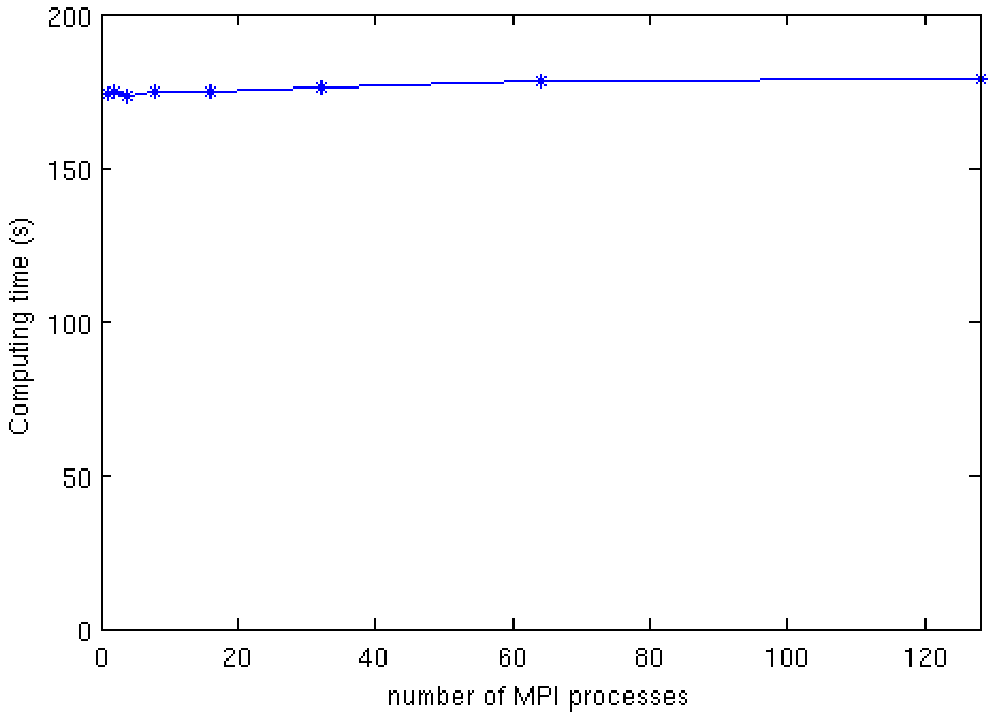


Figure 5.12: Weak scalability.

Strong scalability: To perform the strong scalability tests, we used a domain made of 32×32 subdomains of size 20×20 with Q_3 elements and we realized 1000 time steps. We report on Table 5.4 and on Figure 5.13 the results. We note that the code scales really well. The performances are slightly less good than in the weak scalability experiments, this is most likely due to the lower amount of computation that leads to communications not as well overlapped.

Number of MPI processes	Time (s)	Speed up
1	10999	-
2	5558	1.98
4	2801	3.93
8	1418	7.76
16	708	15.54
32	355	30.98
64	178	61.79
128	99	111.10

Table 5.4: Strong scalability.

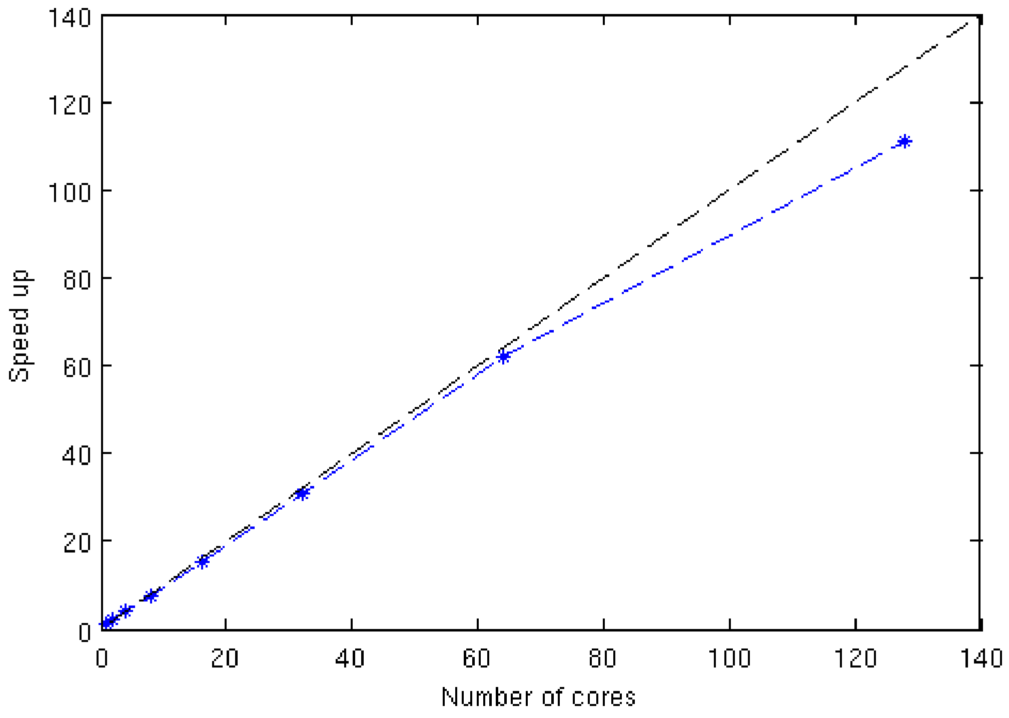


Figure 5.13: Strong scalability.

5.2.2.4 Hybrid OpenMP-MPI performances

We investigate here the performances of hybrid MPI OpenMP (distributed and shared memory) parallelization. The performances of pure MPI parallelization being already really good in the situation we tested, this hybrid parallelization would only be interesting for more demanding simulations. Such simulations could use a higher number of cores, or it could be a situation where we would not be able to divide the subdomains with a good load balancing. Indeed, when the number of MPI processes becomes too large the amount of communication becomes the bottleneck, thus using OpenMP relaxes the communications. Subdomains arising from highly refined areas often lead to difficult load balances, in such situations we can use OpenMP to spend more computing power on these subdomains thus reducing virtually their weights. For instance, an area refined by 100 in

2D will cost approximately 10^6 times the cost of the unrefined coarse area. It is therefore often impossible to have a correct load balance with purely MPI parallelization. We used the same test configuration as in the weak scalability study. We report in Table 5.5 and Table 5.6 the computing times for a node of 16 cores with various distributions for the subdomain and operator based parallel strategies.

MPI processes	OMP threads	Time (s)
1	16	347
2	8	227
4	4	184
8	2	180
16	1	177

Table 5.5: Performances on a node of 16 cores of hybrid MPI/OpenMP for different distributions and the *subdomain based* OpenMP strategies.

MPI processes	OMP threads	Time (s)
1	16	532
2	8	251
4	4	208
8	2	187
16	1	177

Table 5.6: Performances on a node of 16 cores of hybrid MPI/OpenMP for different distributions and the *operator based* OpenMP strategies.

We note that the results given in Table 5.5 and 5.6 favors the subdomain based strategy over the operator based strategy. This can be explained by the more restricted data locality of the operator based strategy over the subdomain based strategy. We emphasize that for small number of processors OpenMP is a lot less efficient than MPI since there is a factor two in the performances for the subdomain based strategy and a factor 3 for the operator based strategy.

5.2.2.5 Realistic case performances

In the previous performance tests we were using only subdomains of same weight, *i.e.* without local refinement or PMLs. Thus, accurate estimation of the computational cost of each subdomain was not an issue. In realistic simulations, due to PML, halo and fine subdomains the weights become inherently heterogeneous. Accurate estimation of the computational cost becomes essential to compute an efficient load distribution.

In order to study the performances in realistic conditions we take a coarse domain of 640×640 Q_3 elements, which we surround with 5 Q_3 elements depth PML and two refined areas. The first refined area is refined by a factor 10 composed of 100×100 Q_3 elements corresponding to 10×10 coarse elements. The second area is refined by a factor 3 composed of 63×63 Q_3 elements corresponding to 21×21 coarse elements. This corresponds to approximately 15 millions of degrees of freedom. We represent this mesh on Figure 5.14.

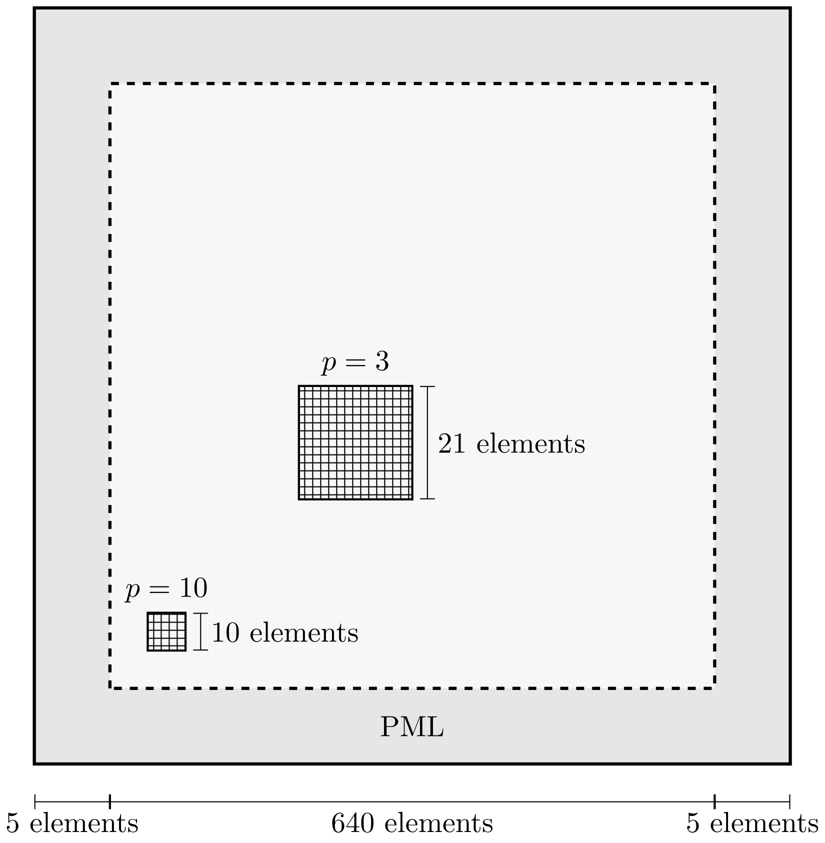


Figure 5.14: Representation of the mesh used for the realistic case performance tests.

According to the study on optimal size in Section 5.2.2.2, we decided to take coarse element subdomains of size 20×20 elements, leading to 32×32 coarse element subdomains. We report the results of this first attempt in Table 5.7. Computational costs are well estimated but looking at processors activity shows that partitioning is still unbalanced due to halo and fine parts having too heavy weights. Finally, to have a better load balancing we reduced the size of halo and fine element subdomains to 10×10 elements. All MPI processes had roughly the same computational work which leads to better performances. We give the computation times of this final attempt in Table 5.8. We performed 1000 time steps for each simulation. We note that the gain is substantial comparing the results in Table 5.7 and Table 5.8. The higher the number of cores the more the load imbalance has a significant impact on the performances.

MPI processes	Time (s)
32	563
64	331
128	216

Table 5.7: Computation times for different numbers of MPI processes. The evaluation of halo element subdomains weights were well estimated but the subdomains were too large leading to unbalanced load balancing.

MPI processes	Time (s)
128	121
64	243
32	482

Table 5.8: Computation times for different numbers of MPI processes. The evaluation of halo element subdomains weights were well estimated and of similar weights as coarse element subdomains leading to a good load balancing.

5.3 Conclusion

In this chapter we introduced our approach to exploit efficiently the Cartesian structured grid, we showed good performances, especially on high polynomial orders. This adds another argument to use high order polynomial when possible.

We showed really good parallel scalability of our MPI implementation due to the use of asynchronous non-blocking communications. In contrast, our OpenMP implementation had a limited scalability due to a limited control of data locality. Thus shared memory parallelism might only be interesting for really high number of cores or to put more computing power on demanding subdomains, *e.g.* highly refined subdomains. However, performing load balancing with PML and refined subdomains was more challenging than expected. Nevertheless, we obtained an efficient load balancing heuristic after extensive numerical experiments to tune the weights of the graph to perform the work distribution.

Conclusion

5.4 General results

In this work, we have proposed an efficient and reliable way to achieve local spatio-temporal mesh refinement for the second order elastodynamic equation. We have first presented the discontinuous Galerkin methods, we motivated our choice by the numerous features these methods offer. In particular, the discontinuous Galerkin methods are some of the rare methods to offer the required h-adaptivity in its standard formulation. Moreover, the p-adaptivity of these methods offers interesting opportunities in a local mesh refinement context. We then presented absorbing layers, called perfectly matched layers (PML), for the second order elastodynamic equation. We used a second order formulation which is less standard than a first order formulation but this facilitated the implementation. We proposed a discontinuous Galerkin formulation for the spatial discretization of the PML and a finite difference time discretization. Both discretizations are not straightforward since there are many possibles choices. Our choices were driven by the desire to keep the CFL stability condition unchanged, which we have shown numerically. Then, we presented the local time stepping method we chose. In the first part of the Chapter 3, we attempted to give a clear insight in the construction of this method. We have proposed different strategies to exploit the p-adaptivity in order to reduce the memory and computational costs of local space-time mesh refinements. We showed that mesh refinement and the local time stepping method introduce spurious effects of really low amplitude. Following this, we proposed a modification of the discontinuous Galerkin method to allow elasto-acoustic media. Finally, we validated our choices of methods on canonical experiments and showed the capabilities on illustrative experiments.

In the last part, we explained our choices for the implementation. In particular, we attempted to exploit discontinuous Galerkin features to have efficient computation. We showed that our implementation exhibits efficient use of matrix-matrix operations (BLAS-3 kernels). We also proposed an asynchronous non-blocking MPI and OpenMP parallelization strategies. The code demonstrated a good scalability of the MPI parallelization up to 128 cores.

5.5 Perspectives

The closest task to realize would be to validate our three dimensions prototype. A relatively simple improvement to our software would be to add multi-level local time stepping method as introduced in [**?**], this would add more flexibility and would also highly reduce the cost of really high local refinements required in some cases, *e.g.* a simple hydrofracture requiring a refinement by 100. For such high refinements, new spatial refinements strategies should be found to limit the huge increase in computational and memory costs. Cartesian meshes are convenient but when more accuracy is required they result in model error (misrepresentation of the medium) being superior to numerical errors (misrepresentation of the solution), destroying all the appealing features of high order methods, in particular discontinuous Galerkin methods. Therefore, locally non Cartesian meshes that follow the medium discontinuities might be necessary to achieve high accuracy.

Appendix A

Sobolev spaces

Definition A.1 - $L^2(\Omega)$ space.
The vector space $L^2(\Omega)$ is the space of square-integrable functions:

$$L^2(\Omega) = \{v \text{ measurable: } \int_\Omega v^2 < \infty\}.$$

The space $L^2(\Omega)$ is a Hilbert space with respect to the following inner product and norm:

$$(u, v)_\Omega = \int_\Omega uv, \quad \|v\|_{L^2(\Omega)} = \left(\int_\Omega v^2\right)^{\frac{1}{2}}.$$

We extend naturally these definitions to vector functions $\mathbf{u} = (u_i)_{1\leq i\leq d}$ and $\mathbf{v} = (v_i)_{1\leq i\leq d}$:

$$(\mathbf{u}, \mathbf{v})_\Omega = \int_\Omega \mathbf{u} \cdot \mathbf{v}, \quad \|\mathbf{v}\|_{L^2(\Omega)} = \left(\sum_{i=1}^{d} \|v_i\|^2_{L^2(\Omega)}\right)^{\frac{1}{2}}.$$

Definition A.2 - $L^\infty(\Omega)$ space.
The space $L^\infty(\Omega)$ is the space of bounded functions:

$$L^\infty(\Omega) = \{v \ : \ \|v\|_{L^\infty(\Omega)} < \infty\},$$

with the norm

$$\|v\|_{L^\infty(\Omega)} = \operatorname*{ess\,sup}_{x\in\Omega}\{|v(x)|\}.$$

Since our equations involve partial derivatives, we need to define a differentiation that is compatible with our functional spaces. This differentiation, called weak differentiation, is define in $L^2(\Omega)$. This notion generalizes the usual differentiation and is a special case of differentiation in the sense of distributions.

Definition A.3 - Weak derivative in $L^2(\Omega)$.
Let v be a function of $L^2(\Omega)$. We say that v is weakly differentiable in $L^2(\Omega)$ if there exists functions $w_i \in L^2(\Omega)$ such that for all function $\phi \in C_0^\infty(\Omega)$, we have

$$\int_\Omega v(x)\frac{\partial \phi}{\partial x_i}(x)\, dx = -\int_\Omega w_i(x)\phi(x)\, dx.$$

Each w_i is called the i-th weak partial derivative of v and noted $\frac{\partial v}{\partial x_i}$.

This definition can easily be generalized by recurrence to n times weakly differentiable functions. We say that a function $v \in L^2(\Omega)$ is n times weakly differentiable if all the weak derivatives of order $n-1$ are weakly differentiable. If we define the multi-index $\alpha = (\alpha_1, .., \alpha_d) \in \mathbb{N}^d$ and $|\alpha| = \sum_{i=1}^d \alpha_i$, we note

$$\partial^\alpha v = \frac{\partial^{|\alpha|} v}{\partial x_1^{\alpha_1} ... \partial x_d^{\alpha_d}}.$$

Remark A.1. *Of course, if a function is strongly differentiable it is weakly differentiable, and the derivatives are equals. The meaning of the notation $\frac{\partial}{\partial x_i}$ is unambiguous since the strong and weak derivatives coincide if they exist.*

Definition A.4 - Sobolev space $H^1(\Omega)$.
The Sobolev space $H^1(\Omega)$ is defined as

$$H^1(\Omega) = \{v \in L^2(\Omega) \ : \ \forall i \in 1, .., d, \ \frac{\partial v}{\partial x_i} \in L^2(\Omega)\}.$$

Remark A.2. *In physics or mechanics, the Sobolev space is often called energy space in the sense that it consists of finite energy functions.*

Definition A.5 - Sobolev space $H^s(\Omega)$.
Similarly, we define Sobolev space $H^s(\Omega)$ for integer s:

$$H^s(\Omega) = \{v \in L^2(\Omega) \ : \ \forall\, 0 \leq |\alpha| \leq s, \partial^\alpha v \in L^2(\Omega)\}.$$

In particular, we have

$$H^2(\Omega) = \{v \in H^1(\Omega) \ : \ \frac{\partial^2 v}{\partial x_1^2}, \frac{\partial^2 v}{\partial x_1 \partial x_2}, \frac{\partial^2 v}{\partial x_2^2} \in L^2(\Omega)\}.$$

The Sobolev norm associated with $H^s(\Omega)$ is

$$\|v\|_{H^s(\Omega)} = \left(\sum_{0 \leq |\alpha| \leq s} \|\partial^\alpha v\|^2_{L^2(\Omega)} \right)^{\frac{1}{2}}.$$

Definition A.6 - Sobolev space $H^{s+\frac{1}{2}}(\Omega)$.
Given $v \in H^s(\Omega)$, we define the following splitting:

$$v = v_1 + v_2$$

where $v_1 \in H^s(\Omega)$ and $v_2 \in H^{s+1}(\Omega)$.Then for a given number t, we define the kernel

$$K(v,t) = \left(\inf_{v_1 + v_2 = v} (\|v_1\|^2_{H^s(\Omega)} + t^2 \, \|v_2\|_{H^{s+1}(\Omega)}) \right)^{\frac{1}{2}}.$$

The space $H^{s+\frac{1}{2}}(\Omega)$ is then defined as the completion of all functions in $H^{s+1}(\Omega)$ with respect to the following norm:

$$\|v\|_{H^{s+\frac{1}{2}}(\Omega)} = \left(\int_0^\infty t^{-2} K^2(v,t) \, dt \right)^{\frac{1}{2}}.$$

Property A.1 - Relation between Sobolev spaces.
We have the following inclusion properties

$$H^{s+1}(\Omega) \subset H^{s+\frac{1}{2}}(\Omega) \subset H^{s}(\Omega).$$

Theorem A.1 - Relation between Sobolev spaces and continuous functions spaces.

$$H^{s}(\Omega) \subset \mathcal{C}^{r}(\Omega) \quad \text{if} \quad \frac{1}{2} < \frac{s-r}{d}.$$

In particular, in two dimensions

$$H^{s}(\Omega) \subset \mathcal{C}^{0}(\Omega) \text{ if } \begin{cases} s > \frac{1}{2} & \text{for} \quad d = 1, \\ s > \frac{2}{2} & \text{for} \quad d = 2, \\ s > \frac{3}{2} & \text{for} \quad d = 3. \end{cases}$$

Definition A.7 - Trace operators.
Let Ω be a bounded domain with polygonal boundary $\partial\Omega$ and outward normal vector $\mathbf{n}$. There exist trace operators $\gamma_0 \ : \ H^s(\Omega) \to H^{s-\frac{1}{2}}(\partial\Omega)$ for $s > \frac{1}{2}$ and $\gamma_1 \ : \ H^s(\Omega) \to H^{s-\frac{3}{2}}(\partial\Omega)$ for $s > \frac{3}{2}$ that are extensions of the boundary values and boundary normal derivatives, respectively. The operators γ_j are surjective. Furthermore, if $v \in \mathcal{C}^1(\bar{\Omega})$, then

$$\gamma_0 v = v|_{\partial\Omega}, \quad \gamma_1 v = \nabla v \cdot \mathbf{n}|_{\partial\Omega}$$

Definition A.8 - Subspace $H_0^s(\Omega)$.

$$H_0^s(\Omega) = \left\{ v \in H^s(\Omega) \ : \ \gamma_0 v = 0 \text{ on } \partial\Omega \right\}.$$

Definition A.9 - Subspace $\tilde{H}_0^s(\Omega)$.

$$\tilde{H}_0^s(\Omega) = \left\{ v \in H^s(\Omega) \ : \ \gamma_0 v = 0 \text{ on } \partial\Omega \cap \Gamma_D \right\}.$$

A.1 Useful formulas

Theorem A.2 - Green's formula.

$$\int_{\Omega} \frac{\partial w}{\partial x_i} = \int_{\partial\Omega} w \mathbf{n}_i$$

$$\int_{\Omega} u \frac{\partial v}{\partial x_i} = -\int_{\Omega} v \frac{\partial u}{\partial x_i} + \int_{\partial\Omega} uv\mathbf{n}_i$$

$$-\int_K w \Delta v = \int_K \nabla v \cdot \nabla w - \int_{\partial K} \nabla v \cdot \mathbf{n}_K w$$

Theorem A.3 - Cauchy-Schwarz's inequality.

$$\forall f, g \in L^2(\Omega), \quad |(f,g)_\Omega| \leq \|f\|_{L^2(\Omega)} \|g\|_{L^2(\Omega)}$$

Theorem A.4 - Young's inequality.

$$\forall \epsilon > 0, \quad \forall a, b \in \mathbb{R}, \quad ab < \frac{\epsilon}{2} a^2 + \frac{1}{2\epsilon} b^2$$

Appendix B

Elastodynamic Formulas

B.1 Elastodynamic Equations

$$\rho\frac{\partial^2 u}{\partial t^2} - div(2\mu e(u) + \lambda tr(e(u))Id) = f,$$

where $e(u) = \frac{1}{2}(\nabla u + \nabla u^t)$.

B.1.1 Two dimensional space case

In a two dimensional space case, we have

$$\nabla u = \begin{pmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} \\ \frac{\partial u_2}{\partial x} & \frac{\partial u_2}{\partial y} \end{pmatrix}, \quad e(u) = \frac{1}{2}\begin{pmatrix} 2\frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} \\ \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} & 2\frac{\partial u_2}{\partial y} \end{pmatrix}. \tag{B.1}$$

We have $\sigma(u) = 2\mu e(u) + \lambda tr(e(u))Id$, using (B.1) we have

$$\sigma(u) = \begin{pmatrix} (\lambda + 2\mu)\frac{\partial u_1}{\partial x} + \lambda\frac{\partial u_2}{\partial y} & \mu\frac{\partial u_1}{\partial y} + \mu\frac{\partial u_2}{\partial x} \\ \mu\frac{\partial u_1}{\partial y} + \mu\frac{\partial u_2}{\partial x} & \lambda\frac{\partial u_1}{\partial x} + (\lambda + 2\mu)\frac{\partial u_2}{\partial y} \end{pmatrix}$$

Thus,

$$\begin{aligned} div(\sigma(u)) &= \begin{pmatrix} (\lambda + 2\mu)\frac{\partial^2 u_1}{\partial x^2} + \lambda\frac{\partial^2 u_2}{\partial x\partial y} + \mu\frac{\partial^2 u_1}{\partial y^2} + \mu\frac{\partial^2 u_2}{\partial x\partial y} \\ \mu\frac{\partial^2 u_1}{\partial x\partial y} + \mu\frac{\partial^2 u_2}{\partial x^2} + \lambda\frac{\partial^2 u_1}{\partial x\partial y} + (\lambda + 2\mu)\frac{\partial^2 u_2}{\partial y^2} \end{pmatrix} \\ &= \begin{pmatrix} (\lambda + 2\mu)\frac{\partial^2 u_1}{\partial x^2} + \mu\frac{\partial^2 u_1}{\partial y^2} + (\lambda + \mu)\frac{\partial^2 u_2}{\partial x\partial y} \\ \mu\frac{\partial^2 u_2}{\partial x^2} + (\lambda + 2\mu)\frac{\partial^2 u_2}{\partial y^2} + (\lambda + \mu)\frac{\partial^2 u_1}{\partial x\partial y} \end{pmatrix} \end{aligned} \tag{B.2}$$

Hence, we have

$$\begin{aligned} \sigma(u)\cdot\nabla v =&(\lambda + 2\mu)\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x} + \lambda\frac{\partial u_2}{\partial y}\frac{\partial v_1}{\partial x} + \mu\frac{\partial u_1}{\partial y}\frac{\partial v_2}{\partial x} + \mu\frac{\partial u_2}{\partial x}\frac{\partial v_2}{\partial x} \\ &+ \mu\frac{\partial u_1}{\partial y}\frac{\partial v_1}{\partial y} + \mu\frac{\partial u_2}{\partial x}\frac{\partial v_1}{\partial y} + \lambda\frac{\partial u_1}{\partial x}\frac{\partial v_2}{\partial y} + (\lambda + 2\mu)\frac{\partial u_2}{\partial y}\frac{\partial v_2}{\partial y} \end{aligned}$$

B.1.2 Three dimensional space case

In a three dimensional space case, we have

$$\nabla u = \begin{pmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} & \frac{\partial u_1}{\partial z} \\ \frac{\partial u_2}{\partial x} & \frac{\partial u_2}{\partial y} & \frac{\partial u_2}{\partial z} \\ \frac{\partial u_3}{\partial x} & \frac{\partial u_3}{\partial y} & \frac{\partial u_3}{\partial z} \end{pmatrix}, \quad e(u) = \frac{1}{2} \begin{pmatrix} 2\frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} & \frac{\partial u_3}{\partial x} + \frac{\partial u_1}{\partial z} \\ \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} & 2\frac{\partial u_2}{\partial y} & \frac{\partial u_3}{\partial y} + \frac{\partial u_2}{\partial z} \\ \frac{\partial u_1}{\partial z} + \frac{\partial u_3}{\partial x} & \frac{\partial u_2}{\partial z} + \frac{\partial u_3}{\partial y} & 2\frac{\partial u_3}{\partial z} \end{pmatrix}. \tag{B.3}$$

We have $\sigma(u) = 2\mu e(u) + \lambda tr(e(u))Id$, using (B.3) we have

$$\sigma(u) = \begin{pmatrix} (\lambda+2\mu)\frac{\partial u_1}{\partial x} + \lambda\frac{\partial u_2}{\partial y} + \lambda\frac{\partial u_3}{\partial z} & \mu\frac{\partial u_1}{\partial y} + \mu\frac{\partial u_2}{\partial x} & \mu\frac{\partial u_3}{\partial x} + \mu\frac{\partial u_1}{\partial z} \\ \mu\frac{\partial u_1}{\partial y} + \mu\frac{\partial u_2}{\partial x} & \lambda\frac{\partial u_1}{\partial x} + (\lambda+2\mu)\frac{\partial u_2}{\partial y} + \lambda\frac{\partial u_3}{\partial z} & \mu\frac{\partial u_3}{\partial y} + \mu\frac{\partial u_2}{\partial z} \\ \mu\frac{\partial u_1}{\partial z} + \mu\frac{\partial u_3}{\partial x} & \mu\frac{\partial u_2}{\partial z} + \mu\frac{\partial u_3}{\partial y} & \lambda\frac{\partial u_1}{\partial x} + \lambda\frac{\partial u_2}{\partial y} + (\lambda+2\mu)\frac{\partial u_3}{\partial z} \end{pmatrix}$$

Thus,

$$\begin{aligned} div(\sigma(u)) &= \begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 u_1}{\partial x^2} + \lambda\frac{\partial^2 u_2}{\partial x\partial y} + \lambda\frac{\partial^2 u_3}{\partial x\partial z} + \mu\frac{\partial^2 u_1}{\partial y^2} + \mu\frac{\partial^2 u_2}{\partial x\partial y} + \mu\frac{\partial^2 u_3}{\partial x\partial z} + \mu\frac{\partial^2 u_1}{\partial z^2} \\ \mu\frac{\partial^2 u_1}{\partial x\partial y} + \mu\frac{\partial^2 u_2}{\partial x^2} + \lambda\frac{\partial^2 u_1}{\partial x\partial y} + (\lambda+2\mu)\frac{\partial^2 u_2}{\partial y^2} + \lambda\frac{\partial^2 u_3}{\partial z^2} + \mu\frac{\partial^2 u_3}{\partial y\partial z} + \mu\frac{\partial^2 u_2}{\partial z^2} \\ \mu\frac{\partial^2 u_1}{\partial x\partial z} + \mu\frac{\partial^2 u_3}{\partial x^2} + \mu\frac{\partial^2 u_2}{\partial y\partial z} + \mu\frac{\partial^2 u_3}{\partial y^2} + \lambda\frac{\partial^2 u_1}{\partial x\partial z} + \lambda\frac{\partial^2 u_2}{\partial y\partial z} + (\lambda+2\mu)\frac{\partial^2 u_3}{\partial z^2} \end{pmatrix} \\ &= \begin{pmatrix} (\lambda+2\mu)\frac{\partial^2 u_1}{\partial x^2} + \mu\frac{\partial^2 u_1}{\partial y^2} + \mu\frac{\partial^2 u_1}{\partial z^2} + (\lambda+\mu)\frac{\partial^2 u_2}{\partial x\partial y} + (\lambda+\mu)\frac{\partial^2 u_3}{\partial x\partial z} \\ (\lambda+\mu)\frac{\partial^2 u_1}{\partial x\partial y} + \mu\frac{\partial^2 u_2}{\partial x^2} + (\lambda+2\mu)\frac{\partial^2 u_2}{\partial y^2} + \mu\frac{\partial^2 u_2}{\partial z^2} + (\lambda+\mu)\frac{\partial^2 u_3}{\partial y\partial z} \\ (\lambda+\mu)\frac{\partial^2 u_1}{\partial x\partial z} + (\lambda+\mu)\frac{\partial^2 u_2}{\partial y\partial z} + \mu\frac{\partial^2 u_3}{\partial x^2} + \mu\frac{\partial^2 u_3}{\partial y^2} + (\lambda+2\mu)\frac{\partial^2 u_3}{\partial z^2} \end{pmatrix} \end{aligned} \tag{B.4}$$

Hence, we have

$$\begin{aligned} \sigma(u)\cdot\nabla v =& (\lambda+2\mu)\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x} + \lambda\frac{\partial u_2}{\partial y}\frac{\partial v_1}{\partial x} + \lambda\frac{\partial u_3}{\partial z}\frac{\partial v_1}{\partial x} + \mu\frac{\partial u_1}{\partial y}\frac{\partial v_2}{\partial x} \\ &+ \mu\frac{\partial u_2}{\partial x}\frac{\partial v_2}{\partial x} + \mu\frac{\partial u_3}{\partial x}\frac{\partial v_3}{\partial x} + \mu\frac{\partial u_1}{\partial z}\frac{\partial v_3}{\partial x} \\ &+ \mu\frac{\partial u_1}{\partial y}\frac{\partial v_1}{\partial y} + \mu\frac{\partial u_2}{\partial x}\frac{\partial v_1}{\partial y} + \lambda\frac{\partial u_1}{\partial x}\frac{\partial v_2}{\partial y} + (\lambda+2\mu)\frac{\partial u_2}{\partial y}\frac{\partial v_2}{\partial y} \\ &+ \lambda\frac{\partial u_3}{\partial z}\frac{\partial v_2}{\partial y} + \mu\frac{\partial u_3}{\partial y}\frac{\partial v_3}{\partial y} + \mu\frac{\partial u_2}{\partial z}\frac{\partial v_3}{\partial y} \\ &+ \mu\frac{\partial u_1}{\partial z}\frac{\partial v_1}{\partial z} + \mu\frac{\partial u_3}{\partial x}\frac{\partial v_1}{\partial z} + \mu\frac{\partial u_2}{\partial z}\frac{\partial v_2}{\partial z} + \mu\frac{\partial u_3}{\partial y}\frac{\partial v_2}{\partial z} \\ &+ \lambda\frac{\partial u_1}{\partial x}\frac{\partial v_3}{\partial z} + \lambda\frac{\partial u_2}{\partial y}\frac{\partial v_3}{\partial z} + (\lambda+2\mu)\frac{\partial u_3}{\partial z}\frac{\partial v_3}{\partial z} \end{aligned}$$

B.2 Dispersion Relation

$$\mathbf{u}_{tt} = A_1\mathbf{u}_{xx} + A_2\mathbf{u}_{yy} + A_3\mathbf{u}_{xy} \tag{B.5}$$

where $A_1 = \begin{pmatrix} \lambda+2\mu & 0 \\ 0 & \mu \end{pmatrix}$, $A_2 = \begin{pmatrix} \mu & 0 \\ 0 & \lambda+2\mu \end{pmatrix}$, $A_3 = \begin{pmatrix} 0 & \lambda+\mu \\ \lambda+\mu & 0 \end{pmatrix}$. If we consider plane wave solutions

$$\mathbf{u} = \mathbf{u_0}e^{i\mathbf{k}\cdot\mathbf{x}-st}. \tag{B.6}$$

Inserting (B.6) in (B.5) yields to the solvability condition called *dispersion relation*

$$det(s^2I + A_1k_x^2 + A_2k_y^2 + A_3k_xk_y) = 0,$$

which leads

$$s^4 + ((\lambda+3\mu)(k_x^2+k_y^2))s^2 + \lambda(\lambda+2\mu)(k_x^4+k_y^4) + 2\mu(\lambda+2\mu)k_x^2k_y^2 = 0. \tag{B.7}$$

Appendix C

PML

C.1 Three dimensional space case

C.1.1 PML Formulation

$$\rho \frac{\partial^2 u}{\partial t^2} - div(2\mu e(u) + \lambda tr(e(u))Id) = f, \tag{C.1}$$

where $e(u) = \frac{1}{2}(\nabla u + \nabla u^T)$.

Step 1: Laplace transform in the time domain.
By using B.4 and then applying the Laplace transform in time to C.1, by setting $f = 0$, we obtain

$$\rho s^2 \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \hat{u}_3 \end{pmatrix} = \begin{pmatrix} (\lambda + 2\mu)\frac{\partial^2 \hat{u}_1}{\partial \tilde{x}^2} + \mu \frac{\partial^2 \hat{u}_1}{\partial \tilde{y}^2} + \mu \frac{\partial^2 \hat{u}_1}{\partial \tilde{z}^2} + (\lambda + \mu)\frac{\partial^2 \hat{u}_2}{\partial \tilde{x} \partial \tilde{y}} + (\lambda + \mu)\frac{\partial^2 \hat{u}_3}{\partial \tilde{x} \partial \tilde{z}} \\ (\lambda + \mu)\frac{\partial^2 \hat{u}_1}{\partial \tilde{x} \partial \tilde{y}} + \mu \frac{\partial^2 \hat{u}_2}{\partial \tilde{x}^2} + (\lambda + 2\mu)\frac{\partial^2 \hat{u}_2}{\partial \tilde{y}^2} + \mu \frac{\partial^2 \hat{u}_2}{\partial \tilde{z}^2} + (\lambda + \mu)\frac{\partial^2 \hat{u}_3}{\partial \tilde{y} \partial \tilde{z}} \\ (\lambda + \mu)\frac{\partial^2 \hat{u}_1}{\partial \tilde{x} \partial \tilde{z}} + (\lambda + \mu)\frac{\partial^2 \hat{u}_2}{\partial \tilde{y} \partial \tilde{z}} + \mu \frac{\partial^2 \hat{u}_3}{\partial \tilde{x}^2} + \mu \frac{\partial^2 \hat{u}_3}{\partial \tilde{y}^2} + (\lambda + 2\mu)\frac{\partial^2 \hat{u}_3}{\partial \tilde{z}^2} \end{pmatrix} \tag{C.2}$$

Step 2: Integration by substitution.
We want to substitute the $\tilde{x}_i$ to the x_i through the coordinate transformation

$$\tilde{x} \, : \Omega \to \Omega_{PML}, \; x_i \to \tilde{x}(x_i) = x_i + \frac{1}{s}\int_0^{x_i} \zeta_i(\xi) d\xi, \quad i = 1, 2, 3,$$

Step 3: Relation between $\frac{\partial}{\partial x_i}$ and $\frac{\partial}{\partial \tilde{x}_i}$.

$$\forall i = 1, 2, \quad \frac{\partial}{\partial \tilde{x}_i} = \frac{s}{s + \zeta_i}\frac{\partial}{\partial x_i} = \frac{1}{\nu_i}\frac{\partial}{\partial x_i}. \tag{C.3}$$

Step 4: Continuation in a complex manifold.
By applying C.3 to C.2 we obtain

$$\begin{cases} \rho s^2 \hat{u}_1 = & (\lambda+2\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) + \mu\frac{1}{\nu_3}\frac{\partial}{\partial z}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_1}{\partial z}\right) \\ & + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_3}{\partial z}\right), \\ \rho s^2 \hat{u}_2 = & \mu\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) + \mu\frac{1}{\nu_3}\frac{\partial}{\partial z}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_2}{\partial z}\right) \\ & + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) + (\lambda+\mu)\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_3}{\partial z}\right), \\ \rho s^2 \hat{u}_3 = & \mu\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_1}\frac{\partial \hat{u}_3}{\partial x}\right) + \mu\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_2}\frac{\partial \hat{u}_3}{\partial y}\right) + (\lambda+2\mu)\frac{1}{\nu_3}\frac{\partial}{\partial z}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_3}{\partial z}\right) \\ & + (\lambda+\mu)\frac{1}{\nu_1}\frac{\partial}{\partial x}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_1}{\partial z}\right) + (\lambda+\mu)\frac{1}{\nu_2}\frac{\partial}{\partial y}\left(\frac{1}{\nu_3}\frac{\partial \hat{u}_2}{\partial z}\right). \end{cases}$$

Hence, by multiplying by $\nu_1\nu_2\nu_3$ we obtain

$$\begin{cases} \rho s^2 \nu_1\nu_2\nu_3 \hat{u}_1 = & (\lambda+2\mu)\frac{\partial}{\partial x}\left(\frac{\nu_2\nu_3}{\nu_1}\frac{\partial \hat{u}_1}{\partial x}\right) + \mu\frac{\partial}{\partial y}\left(\frac{\nu_1\nu_3}{\nu_2}\frac{\partial \hat{u}_1}{\partial y}\right) + \mu\frac{\partial}{\partial z}\left(\frac{\nu_1\nu_2}{\nu_3}\frac{\partial \hat{u}_1}{\partial z}\right) \\ & + (\lambda+\mu)\frac{\partial}{\partial x}\left(\nu_3\frac{\partial \hat{u}_2}{\partial y}\right) + (\lambda+\mu)\frac{\partial}{\partial x}\left(\nu_2\frac{\partial \hat{u}_3}{\partial z}\right), \\ \rho s^2 \nu_1\nu_2\nu_3 \hat{u}_2 = & \mu\frac{\partial}{\partial x}\left(\frac{\nu_2\nu_3}{\nu_1}\frac{\partial \hat{u}_2}{\partial x}\right) + (\lambda+2\mu)\frac{\partial}{\partial y}\left(\frac{\nu_1\nu_3}{\nu_2}\frac{\partial \hat{u}_2}{\partial y}\right) + \mu\frac{\partial}{\partial z}\left(\frac{\nu_1\nu_2}{\nu_3}\frac{\partial \hat{u}_2}{\partial z}\right) \\ & + (\lambda+\mu)\frac{\partial}{\partial x}\left(\nu_3\frac{\partial \hat{u}_1}{\partial y}\right) + (\lambda+\mu)\frac{\partial}{\partial y}\left(\nu_1\frac{\partial \hat{u}_3}{\partial z}\right), \\ \rho s^2 \nu_1\nu_2\nu_3 \hat{u}_3 = & \mu\frac{\partial}{\partial x}\left(\frac{\nu_2\nu_3}{\nu_1}\frac{\partial \hat{u}_3}{\partial x}\right) + \mu\frac{\partial}{\partial y}\left(\frac{\nu_1\nu_3}{\nu_2}\frac{\partial \hat{u}_3}{\partial y}\right) + (\lambda+2\mu)\frac{\partial}{\partial z}\left(\frac{\nu_1\nu_2}{\nu_3}\frac{\partial \hat{u}_3}{\partial z}\right) \\ & + (\lambda+\mu)\frac{\partial}{\partial x}\left(\nu_2\frac{\partial \hat{u}_1}{\partial z}\right) + (\lambda+\mu)\frac{\partial}{\partial y}\left(\nu_1\frac{\partial \hat{u}_2}{\partial z}\right). \end{cases}$$

Besides,

$$\begin{cases} \forall i = 1,2,3, \quad \nu_i = 1 + \frac{\zeta_i}{s} \\ \frac{\nu_2\nu_3}{\nu_1} = 1 + \frac{(\zeta_2+\zeta_3-\zeta_1)s + \zeta_2\zeta_3}{(s+\zeta_1)s}, \\ \frac{\nu_1\nu_3}{\nu_2} = 1 + \frac{(\zeta_1+\zeta_3-\zeta_2)s + \zeta_1\zeta_3}{(s+\zeta_2)s}, \\ \frac{\nu_1\nu_2}{\nu_3} = 1 + \frac{(\zeta_1+\zeta_2-\zeta_3)s + \zeta_1\zeta_2}{(s+\zeta_3)s}, \\ \nu_1\nu_2\nu_3 = \frac{s^3 + s^2(\zeta_1+\zeta_2+\zeta_3) + s(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3) + \zeta_1\zeta_2\zeta_3}{s^3}. \end{cases}$$

Thus, we obtain

$$
\left\{
\begin{aligned}
&\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\frac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_1 = \\
&(\lambda+2\mu)\frac{\partial^2\hat{u}_1}{\partial x^2}+\mu\frac{\partial^2\hat{u}_1}{\partial y^2}+\mu\frac{\partial^2\hat{u}_1}{\partial z^2}+(\lambda+\mu)\frac{\partial^2\hat{u}_2}{\partial x\partial y}+(\lambda+\mu)\frac{\partial^2\hat{u}_3}{\partial x\partial z} \\
&\;+(\lambda+2\mu)\frac{\partial}{\partial x}\left(\frac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\frac{\partial\hat{u}_1}{\partial x}\right)+\mu\frac{\partial}{\partial y}\left(\frac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\frac{\partial\hat{u}_1}{\partial y}\right) \\
&\;+\mu\frac{\partial}{\partial z}\left(\frac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\frac{\partial\hat{u}_1}{\partial z}\right)+(\lambda+\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_3}{s}\frac{\partial\hat{u}_2}{\partial y}\right)+(\lambda+\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_2}{s}\frac{\partial\hat{u}_3}{\partial z}\right), \\
&\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\frac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_2 = \\
&\mu\frac{\partial^2\hat{u}_2}{\partial x^2}+(\lambda+2\mu)\frac{\partial^2\hat{u}_2}{\partial y^2}+\mu\frac{\partial^2\hat{u}_2}{\partial z^2}+(\lambda+\mu)\frac{\partial^2\hat{u}_1}{\partial x\partial y}+(\lambda+\mu)\frac{\partial^2\hat{u}_3}{\partial y\partial z} \\
&\;+\mu\frac{\partial}{\partial x}\left(\frac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\frac{\partial\hat{u}_2}{\partial x}\right)+(\lambda+2\mu)\frac{\partial}{\partial y}\left(\frac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\frac{\partial\hat{u}_2}{\partial y}\right) \\
&\;+\mu\frac{\partial}{\partial z}\left(\frac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\frac{\partial\hat{u}_2}{\partial z}\right)+(\lambda+\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_3}{s}\frac{\partial\hat{u}_1}{\partial y}\right)+(\lambda+\mu)\frac{\partial}{\partial y}\left(\frac{\zeta_1}{s}\frac{\partial\hat{u}_3}{\partial z}\right), \\
&\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\frac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_3 = \\
&\mu\frac{\partial^2\hat{u}_3}{\partial x^2}+\mu\frac{\partial^2\hat{u}_3}{\partial y^2}+(\lambda+2\mu)\frac{\partial^2\hat{u}_3}{\partial z^2}+(\lambda+\mu)\frac{\partial^2\hat{u}_1}{\partial x\partial z}+(\lambda+\mu)\frac{\partial^2\hat{u}_2}{\partial y\partial z} \\
&\;+\mu\frac{\partial}{\partial x}\left(\frac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\frac{\partial\hat{u}_3}{\partial x}\right)+\mu\frac{\partial}{\partial y}\left(\frac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\frac{\partial\hat{u}_3}{\partial y}\right) \\
&\;+(\lambda+2\mu)\frac{\partial}{\partial z}\left(\frac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\frac{\partial\hat{u}_3}{\partial z}\right)+(\lambda+\mu)\frac{\partial}{\partial x}\left(\frac{\zeta_2}{s}\frac{\partial\hat{u}_1}{\partial z}\right)+(\lambda+\mu)\frac{\partial}{\partial y}\left(\frac{\zeta_1}{s}\frac{\partial\hat{u}_2}{\partial z}\right).
\end{aligned}
\right.
$$

Step 5: Defining the auxiliary variables.
By defining the auxiliary variables

$$\begin{cases}
\tilde{\psi} = \dfrac{\hat{u}}{s}, \\
\tilde{\phi}_{11} = \dfrac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\dfrac{\partial \hat{u}_1}{\partial x}, \\
\tilde{\phi}_{12} = \dfrac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\dfrac{\partial \hat{u}_1}{\partial y}, \\
\tilde{\phi}_{13} = \dfrac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\dfrac{\partial \hat{u}_1}{\partial z}, \\
\tilde{\phi}_{21} = \dfrac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\dfrac{\partial \hat{u}_2}{\partial x}, \\
\tilde{\phi}_{22} = \dfrac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\dfrac{\partial \hat{u}_2}{\partial y}, \\
\tilde{\phi}_{23} = \dfrac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\dfrac{\partial \hat{u}_2}{\partial z}, \\
\tilde{\phi}_{31} = \dfrac{(\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3}{(s+\zeta_1)s}\dfrac{\partial \hat{u}_3}{\partial x}, \\
\tilde{\phi}_{32} = \dfrac{(\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3}{(s+\zeta_2)s}\dfrac{\partial \hat{u}_3}{\partial y}, \\
\tilde{\phi}_{33} = \dfrac{(\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2}{(s+\zeta_3)s}\dfrac{\partial \hat{u}_3}{\partial z}.
\end{cases}$$

we obtain the following system of equations

$$\begin{cases}
\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\dfrac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_1 = \\
(\lambda+2\mu)\dfrac{\partial^2\hat{u}_1}{\partial x^2}+\mu\dfrac{\partial^2\hat{u}_1}{\partial y^2}+\mu\dfrac{\partial^2\hat{u}_1}{\partial z^2}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_2}{\partial x\partial y}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_3}{\partial x\partial z} \\
+(\lambda+2\mu)\dfrac{\partial\tilde{\phi}_{11}}{\partial x}+\mu\dfrac{\partial\tilde{\phi}_{12}}{\partial y}+\mu\dfrac{\partial\tilde{\phi}_{13}}{\partial z} \\
+(\lambda+\mu)\dfrac{\partial}{\partial x}\left(\zeta_3\dfrac{\partial\tilde{\psi}_2}{\partial y}\right)+(\lambda+\mu)\dfrac{\partial}{\partial x}\left(\zeta_2\dfrac{\partial\tilde{\psi}_3}{\partial z}\right), \\
\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\dfrac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_2 = \\
\mu\dfrac{\partial^2\hat{u}_2}{\partial x^2}+(\lambda+2\mu)\dfrac{\partial^2\hat{u}_2}{\partial y^2}+\mu\dfrac{\partial^2\hat{u}_2}{\partial z^2}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_1}{\partial x\partial y}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_3}{\partial y\partial z} \\
+\mu\dfrac{\partial\tilde{\phi}_{21}}{\partial x}+(\lambda+2\mu)\dfrac{\partial\tilde{\phi}_{22}}{\partial y}+\mu\dfrac{\partial\tilde{\phi}_{23}}{\partial z} \\
+(\lambda+\mu)\dfrac{\partial}{\partial x}\left(\zeta_3\dfrac{\partial\tilde{\psi}_1}{\partial y}\right)+(\lambda+\mu)\dfrac{\partial}{\partial y}\left(\zeta_1\dfrac{\partial\tilde{\psi}_3}{\partial z}\right), \\
\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)+\dfrac{\zeta_1\zeta_2\zeta_3}{s})\hat{u}_3 = \\
\mu\dfrac{\partial^2\hat{u}_3}{\partial x^2}+\mu\dfrac{\partial^2\hat{u}_3}{\partial y^2}+(\lambda+2\mu)\dfrac{\partial^2\hat{u}_3}{\partial z^2}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_1}{\partial x\partial z}+(\lambda+\mu)\dfrac{\partial^2\hat{u}_2}{\partial y\partial z} \\
+\mu\dfrac{\partial\tilde{\phi}_{31}}{\partial x}+\mu\dfrac{\partial\tilde{\phi}_{32}}{\partial y}+(\lambda+2\mu)\dfrac{\partial\tilde{\phi}_{33}}{\partial z} \\
+(\lambda+\mu)\dfrac{\partial}{\partial x}\left(\zeta_2\dfrac{\partial\tilde{\psi}_1}{\partial z}\right)+(\lambda+\mu)\dfrac{\partial}{\partial y}\left(\zeta_1\dfrac{\partial\tilde{\psi}_2}{\partial z}\right), \\
s\tilde{\psi}=\hat{u}, \\
(s+\zeta_1)s\tilde{\phi}_{11}=((\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3)\frac{\partial\hat{u}_1}{\partial x}, \\
(s+\zeta_2)s\tilde{\phi}_{12}=((\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3)\frac{\partial\hat{u}_1}{\partial y}, \\
(s+\zeta_3)s\tilde{\phi}_{13}=((\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2)\frac{\partial\hat{u}_1}{\partial z}, \\
(s+\zeta_1)s\tilde{\phi}_{21}=((\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3)\frac{\partial\hat{u}_2}{\partial x}, \\
(s+\zeta_2)s\tilde{\phi}_{22}=((\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3)\frac{\partial\hat{u}_2}{\partial y}, \\
(s+\zeta_3)s\tilde{\phi}_{23}=((\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2)\frac{\partial\hat{u}_2}{\partial z}, \\
(s+\zeta_1)s\tilde{\phi}_{31}=((\zeta_2+\zeta_3-\zeta_1)s+\zeta_2\zeta_3)\frac{\partial\hat{u}_3}{\partial x}, \\
(s+\zeta_2)s\tilde{\phi}_{32}=((\zeta_1+\zeta_3-\zeta_2)s+\zeta_1\zeta_3)\frac{\partial\hat{u}_3}{\partial y}, \\
(s+\zeta_3)s\tilde{\phi}_{33}=((\zeta_1+\zeta_2-\zeta_3)s+\zeta_1\zeta_2)\frac{\partial\hat{u}_3}{\partial z}.
\end{cases}$$

or equivalently

$$\begin{cases}
\rho(s^2+s(\zeta_1+\zeta_2+\zeta_3)+(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3))\hat{u}+\zeta_1\zeta_2\zeta_3\tilde{\psi} = \\
div(\underline{\underline{\sigma}}(\hat{u}))+div(\Phi_1:\underline{\underline{\tilde{\phi}}})+div(\Phi_2:\nabla\tilde{\psi}), \\
s\underline{\underline{\tilde{\phi}}}=\underline{\underline{\tilde{\phi}}}\Psi_1+\nabla\hat{u}\Psi_2+\nabla\tilde{\psi}\Psi_3, \\
s\tilde{\psi}=\hat{u}.
\end{cases}$$

where

$$\Phi_1 = \begin{pmatrix} \lambda+2\mu & \mu & \mu \\ \mu & \lambda+2\mu & \mu \\ \mu & \mu & \lambda+2\mu \end{pmatrix}$$

$$\Phi_2 = (\lambda+\mu)\begin{pmatrix} 0 & \zeta_3 & \zeta_2 \\ \zeta_3 & 0 & \zeta_1 \\ \zeta_2 & \zeta_1 & 0 \end{pmatrix}$$

$$\Psi_1 = \begin{pmatrix} -\zeta_1 & 0 & 0 \\ 0 & -\zeta_2 & 0 \\ 0 & 0 & -\zeta_3 \end{pmatrix}$$

$$\Psi_2 = \begin{pmatrix} \zeta_2+\zeta_3-\zeta_1 & 0 & 0 \\ 0 & \zeta_1+\zeta_3-\zeta_2 & 0 \\ 0 & 0 & \zeta_1+\zeta_2-\zeta_3 \end{pmatrix}$$

$$\Psi_3 = \begin{pmatrix} \zeta_2\zeta_3 & 0 & 0 \\ 0 & \zeta_1\zeta_3 & 0 \\ 0 & 0 & \zeta_1\zeta_2 \end{pmatrix}$$

Step 6: Inverse Laplace transformation.
Finally, we apply the inverse Laplace transformation to the time domain and obtain the PML equations for the elastodynamic equations

$$\begin{cases} \rho\dfrac{\partial^2 u}{\partial t^2} + \rho(\zeta_1+\zeta_2+\zeta_3)\dfrac{\partial u}{\partial t} + \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)u = \\ div(\underline{\underline{\sigma}}(u)) + div(\Phi_1 : \underline{\underline{\phi}}) + div(\Phi_2 : \nabla\psi) - \zeta_1\zeta_2\zeta_3\psi, \\ \dfrac{\partial \underline{\underline{\phi}}}{\partial t} = \underline{\underline{\phi}}\Psi_1 + \nabla u\Psi_2 + \nabla\psi\Psi_3, \\ \dfrac{\partial\psi}{\partial t} = u. \end{cases} \tag{C.4}$$

where

$$\Phi_1 = \begin{pmatrix} \lambda+2\mu & \mu & \mu \\ \mu & \lambda+2\mu & \mu \\ \mu & \mu & \lambda+2\mu \end{pmatrix}$$

$$\Phi_2 = (\lambda+\mu)\begin{pmatrix} 0 & \zeta_3 & \zeta_2 \\ \zeta_3 & 0 & \zeta_1 \\ \zeta_2 & \zeta_1 & 0 \end{pmatrix}$$

$$\Psi_1 = \begin{pmatrix} -\zeta_1 & 0 & 0 \\ 0 & -\zeta_2 & 0 \\ 0 & 0 & -\zeta_3 \end{pmatrix}$$

$$\Psi_2 = \begin{pmatrix} \zeta_2+\zeta_3-\zeta_1 & 0 & 0 \\ 0 & \zeta_1+\zeta_3-\zeta_2 & 0 \\ 0 & 0 & \zeta_1+\zeta_2-\zeta_3 \end{pmatrix}$$

$$\Psi_3 = \begin{pmatrix} \zeta_2\zeta_3 & 0 & 0 \\ 0 & \zeta_1\zeta_3 & 0 \\ 0 & 0 & \zeta_1\zeta_2 \end{pmatrix}$$

C.1.2 Variational Formulation

Step 1: Multiply all equations by test functions.
We multiply the first equation of C.4 by a test function $v \in H^s(\mathcal{T}_h)^d$, the second equation by a test function $\varphi \in H^s(\mathcal{T}_h)^{d^2}$, and the third equation by v also, we obtain the system

$$\begin{cases} \rho\dfrac{\partial^2 u}{\partial t^2}\cdot v + \rho(\zeta_1+\zeta_2+\zeta_3)\dfrac{\partial u}{\partial t}\cdot v + \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3)u\cdot v = \\ div(\underline{\underline{\sigma}}(u))\cdot v + div(\Phi_1 : \underline{\underline{\phi}})\cdot v + div(\Phi_2 : \nabla\psi)\cdot v - \zeta_1\zeta_2\zeta_3\psi\cdot v, \\ \dfrac{\partial \underline{\underline{\phi}}}{\partial t}\cdot\varphi = (\underline{\underline{\phi}}\Psi_1)\cdot\varphi + (\nabla u\Psi_2)\cdot\varphi + (\nabla\psi\Psi_3)\cdot\varphi, \\ \dfrac{\partial\psi}{\partial t}\cdot v = u\cdot v. \end{cases}$$

Step 2: Integration on domain Ω.

$$\begin{cases}
\int_\Omega \rho \frac{\partial^2 u}{\partial t^2} \cdot v\, dx + \int_\Omega \rho(\zeta_1+\zeta_2+\zeta_3)\frac{\partial u}{\partial t} \cdot v\, dx + \int_\Omega \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3) u \cdot v\, dx = \\
\int_\Omega div(\underline{\underline{\sigma}}(u)) \cdot v\, dx + \int_\Omega div(\Phi_1 : \underline{\underline{\phi}}) \cdot v\, dx + \int_\Omega div(\Phi_2 : \nabla\psi) \cdot v\, dx - \int_\Omega \zeta_1\zeta_2\zeta_3 \psi \cdot v\, dx, \\
\int_\Omega \frac{\partial \underline{\underline{\phi}}}{\partial t} \cdot \varphi\, dx = \int_\Omega (\underline{\underline{\phi}}\Psi_1) \cdot \varphi\, dx + \int_\Omega (\nabla u \Psi_2) \cdot \varphi\, dx + \int_\Omega (\nabla\psi\Psi_3) \cdot \varphi\, dx, \\
\int_\Omega \frac{\partial \psi}{\partial t} \cdot v\, dx = \int_\Omega u \cdot v\, dx.
\end{cases}$$

As $\Omega = \bigcup_{K\in\mathcal{T}_h} K$, we have

$$\begin{cases}
\sum_{K\in\mathcal{T}_h} \left(\int_K \rho \frac{\partial^2 u}{\partial t^2} \cdot v\, dx + \int_K \rho(\zeta_1+\zeta_2+\zeta_3)\frac{\partial u}{\partial t} \cdot v\, dx + \int_K \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3) u \cdot v\, dx \right) = \\
\sum_{K\in\mathcal{T}_h} \left(\int_K div(\underline{\underline{\sigma}}(u)) \cdot v\, dx + \int_K div(\Phi_1 : \underline{\underline{\phi}}) \cdot v\, dx + \int_K div(\Phi_2 : \nabla\psi) \cdot v\, dx - \int_K \zeta_1\zeta_2\zeta_3 \psi \cdot v\, dx \right), \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \underline{\underline{\phi}}}{\partial t} \cdot \varphi\, dx = \sum_{K\in\mathcal{T}_h} \left(\int_K (\underline{\underline{\phi}}\Psi_1) \cdot \varphi\, dx + \int_K (\nabla u \Psi_2) \cdot \varphi\, dx + \int_K (\nabla\psi\Psi_3) \cdot \varphi\, dx \right), \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \psi}{\partial t} \cdot v\, dx = \sum_{K\in\mathcal{T}_h} \int_K u \cdot v\, dx.
\end{cases}$$

Step 3: Green's Formula.

$$\int_K div(\sigma(u)) \cdot v\, dx = -\int_K \sigma(u) \cdot \nabla v\, dx + \int_{\partial K} (\sigma(u)n) \cdot v\, ds.$$

As for classical IPDG formulation we have

$$\sum_{K\in\mathcal{T}_h} \int_{\partial K} (\sigma(u)n) \cdot v\, ds = \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\sigma(u)n\}\!\} \cdot [\![v]\!]\, ds$$

Thus, we obtain

$$\begin{cases}
\sum_{K\in\mathcal{T}_h} \left(\int_K \rho \frac{\partial^2 u}{\partial t^2} \cdot v\, dx + \int_K \rho(\zeta_1+\zeta_2+\zeta_3)\frac{\partial u}{\partial t} \cdot v\, dx + \int_K \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3) u \cdot v\, dx \right) = \\
- \sum_{K\in\mathcal{T}_h} \int_K \underline{\underline{\sigma}}(u) \cdot \nabla v\, dx + \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\} \cdot [\![v]\!]\, ds \\
+ \sum_{K\in\mathcal{T}_h} \int_K div(\Phi_1 : \underline{\underline{\phi}}) \cdot v\, dx + \sum_{F\in\mathcal{F}_h} \int_F [\![(\Phi_1 : \underline{\underline{\phi}})n]\!] \cdot \{\!\{v\}\!\}\, ds \\
- \sum_{K\in\mathcal{T}_h} \int_K (\Phi_2 : \nabla\psi) \cdot \nabla v\, dx + \sum_{F\in\mathcal{F}_h} \int_F \{\!\{(\Phi_2 : \nabla\psi)n\}\!\} \cdot [\![v]\!]\, ds - \sum_{K\in\mathcal{T}_h} \int_K \zeta_1\zeta_2\zeta_3 \psi \cdot v\, dx, \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \underline{\underline{\phi}}}{\partial t} \cdot \varphi\, dx = \sum_{K\in\mathcal{T}_h} \int_K (\underline{\underline{\phi}}\Psi_1) \cdot \varphi\, dx + \sum_{K\in\mathcal{T}_h} \int_K (\nabla u \Psi_2) \cdot \varphi\, dx \\
+ \sum_{F\in\mathcal{F}_h} \int_F [\![u]\!] \cdot \{\!\{(\varphi\Psi_2)n\}\!\}\, ds + \sum_{K\in\mathcal{T}_h} \int_K (\nabla\psi\Psi_3) \cdot \varphi\, dx + \sum_{F\in\mathcal{F}_h} \int_F [\![\psi]\!] \cdot \{\!\{(\varphi\Psi_3)n\}\!\}\, ds, \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \psi}{\partial t} \cdot v\, dx = \sum_{K\in\mathcal{T}_h} \int_K u \cdot v\, dx.
\end{cases}$$

We add the classical IPDG symmetric term $\int_F [\![u]\!] \cdot \{\!\{\underline{\underline{\sigma}}(v)n\}\!\}\, ds$ and the penalization term $-\int_F \alpha_F [\![u]\!] \cdot [\![v]\!]\, ds$, thus, we obtain

$$\begin{cases}
\sum_{K\in\mathcal{T}_h} \left(\int_K \rho \frac{\partial^2 u}{\partial t^2} \cdot v\, dx + \int_K \rho(\zeta_1+\zeta_2+\zeta_3)\frac{\partial u}{\partial t} \cdot v\, dx + \int_K \rho(\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3) u \cdot v\, dx \right) = \\
-\sum_{K\in\mathcal{T}_h} \int_K \underline{\underline{\sigma}}(u) \cdot \nabla v\, dx + \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\underline{\underline{\sigma}}(u)n\}\!\} \cdot [\![v]\!]\, ds + \sum_{F\in\mathcal{F}_h} \int_F [\![u]\!] \cdot \{\!\{\underline{\underline{\sigma}}(v)n\}\!\}\, ds \\
-\sum_{F\in\mathcal{F}_h} \int_F \alpha_F [\![u]\!] \cdot [\![v]\!]\, ds + \sum_{K\in\mathcal{T}_h} \int_K div(\Phi_1 : \underline{\underline{\phi}}) \cdot v\, dx + \sum_{F\in\mathcal{F}_h} \int_F [\![(\Phi_1 : \underline{\underline{\phi}})n]\!] \cdot \{\!\{v\}\!\}\, ds \\
-\sum_{K\in\mathcal{T}_h} \int_K (\Phi_2 : \nabla\psi) \cdot \nabla v\, dx + \sum_{F\in\mathcal{F}_h} \int_F \{\!\{(\Phi_2 : \nabla\psi)n\}\!\} \cdot [\![v]\!]\, ds - \sum_{K\in\mathcal{T}_h} \int_K \zeta_1\zeta_2\zeta_3 \psi \cdot v\, dx, \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \underline{\underline{\phi}}}{\partial t} \cdot \varphi\, dx = \sum_{K\in\mathcal{T}_h} \int_K (\underline{\underline{\phi}}\Psi_1) \cdot \varphi\, dx + \sum_{K\in\mathcal{T}_h} \int_K (\nabla u \Psi_2) \cdot \varphi\, dx \\
+\sum_{F\in\mathcal{F}_h} \int_F [\![u]\!] \cdot \{\!\{(\varphi\Psi_2)n\}\!\}\, ds + \sum_{K\in\mathcal{T}_h} \int_K (\nabla\psi\Psi_3) \cdot \varphi\, dx + \sum_{F\in\mathcal{F}_h} \int_F [\![\psi]\!] \cdot \{\!\{(\varphi\Psi_3)n\}\!\}\, ds, \\
\sum_{K\in\mathcal{T}_h} \int_K \frac{\partial \psi}{\partial t} \cdot v\, dx = \sum_{K\in\mathcal{T}_h} \int_K u \cdot v\, dx.
\end{cases}$$

where

$$\Phi_1 = \begin{pmatrix} \lambda+2\mu & \mu & \mu \\ \mu & \lambda+2\mu & \mu \\ \mu & \mu & \lambda+2\mu \end{pmatrix}$$

$$\Phi_2 = (\lambda+\mu) \begin{pmatrix} 0 & \zeta_3 & \zeta_2 \\ \zeta_3 & 0 & \zeta_1 \\ \zeta_2 & \zeta_1 & 0 \end{pmatrix}$$

$$\Psi_1 = \begin{pmatrix} -\zeta_1 & 0 & 0 \\ 0 & -\zeta_2 & 0 \\ 0 & 0 & -\zeta_3 \end{pmatrix}$$

$$\Psi_2 = \begin{pmatrix} \zeta_2+\zeta_3-\zeta_1 & 0 & 0 \\ 0 & \zeta_1+\zeta_3-\zeta_2 & 0 \\ 0 & 0 & \zeta_1+\zeta_2-\zeta_3 \end{pmatrix}$$

$$\Psi_3 = \begin{pmatrix} \zeta_2\zeta_3 & 0 & 0 \\ 0 & \zeta_1\zeta_3 & 0 \\ 0 & 0 & \zeta_1\zeta_2 \end{pmatrix}$$

C.1.3 Space Discretization

C.1.3.1 Global Formulation of the Space Discretization

The global space discretization of the PML is

$$\begin{cases}
M\frac{\partial^2 U}{\partial t^2} + M_{\zeta_1+\zeta_2+\zeta_3}\frac{\partial U}{\partial t} + M_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3} U = K_\sigma U + K_{\Phi_1}\phi + K_{\Phi_2}\psi, \\
M\frac{\partial \underline{\underline{\phi}}}{\partial t} = K_{\Psi_1}\phi + K_{\Psi_2} U + K_{\Psi_3}\psi, \\
\frac{\partial \psi}{\partial t} = U.
\end{cases}$$

This global formulation is really neat, but has a major drawback, it's hiding all the locality of the discontinuous Galerkin and consequently all the attractiveness and difficulties of the method. For this reason, we prefer to rewrite these equations in a local form.

C.1.3.2 Local Formulation of the Space Discretization

To obtain the local formulation of the variational formulation we have to consider a test function which is not null only on a reference element K, thus we obtain the following local variational formulation

$$\begin{cases} \rho_K M^K \dfrac{\partial^2 u^K}{\partial t^2} + \rho_K M^K_{\zeta_1+\zeta_2+\zeta_3} \dfrac{\partial u^K}{\partial t} + \rho_K M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3} u^K = -K^K_\sigma u^K + \sum\limits_{F\in\mathcal{F}_K} F^{V_F(K)}_\sigma u^{V_F(K)} \\ + K^K_{\Psi_1}\phi^K + \sum\limits_{F\in\mathcal{F}_K} F^{V_F(K)}_{\Psi_1}\phi^{V_F(K)} + K^K_{\Psi_2}\psi^K + \sum\limits_{F\in\mathcal{F}_K} F^{V_F(K)}_{\Psi_2}\psi^{V_F(K)}, \\ M^K \dfrac{\partial \phi^K}{\partial t} = K^K_{\Psi_1}\phi^K + K^K_{\Psi_2} u^K + \sum\limits_{F\in\mathcal{F}_K} F^{V_F(K)}_{\Psi_2} u^{V_F(K)} + K^K_{\Psi_3}\psi^K + \sum\limits_{F\in\mathcal{F}_K} F^{V_F(K)}_{\Psi_3}\psi^{V_F(K)}, \\ \dfrac{\partial \psi^K}{\partial t} = u^K, \end{cases}$$

where

$$\begin{aligned} M^K_{\zeta_1+\zeta_2+\zeta_3} &= M^K_{\zeta_1} + M^K_{\zeta_2} + M^K_{\zeta_3} \\ &= h^N_K \left(a^K_{\zeta_1}\tilde{M}_{x^2} + a^K_{\zeta_2}\tilde{M}_{y^2} + a^K_{\zeta_3}\tilde{M}_{z^2} + b^K_{\zeta_1}\tilde{M}_x + b^K_{\zeta_2}\tilde{M}_y + b^K_{\zeta_3}\tilde{M}_z + (c^K_{\zeta_1} + c^K_{\zeta_2} + c^K_{\zeta_3})\tilde{M} \right), \end{aligned}$$

and the matrix $M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3}$ can be decomposed the same way. The Stiffness and flux matrices, K and F, can be decomposed as previously.
Thus we can decompose each element matrices in a linear sum of reference matrices.

$$\begin{aligned} M^K_{\zeta_1+\zeta_2} &= M^K_{\zeta_1} + M^K_{\zeta_2} \\ &= h^N_K \left(a^K_{\zeta_1}\tilde{M}_{x^2} + a^K_{\zeta_2}\tilde{M}_{y^2} + b^K_{\zeta_1}\tilde{M}_x + b^K_{\zeta_2}\tilde{M}_y + (c^K_{\zeta_1} + c^K_{\zeta_2})\tilde{M} \right), \end{aligned}$$

and the matrix $M^K_{\zeta_1\zeta_2}$ can be decomposed the same way. The Stiffness and flux matrices, K and F, can be decomposed as previously.
Thus we can decompose each element matrices in a linear sum of reference matrices.

C.1.4 Time Discretization

$$\begin{cases} \rho_K M^K \dfrac{u^K_{n+1} - 2u^K_n + u^K_{n-1}}{\Delta t^2} + \rho_K M^K_{\zeta_1+\zeta_2+\zeta_3} \dfrac{u^K_{n+1} - u^K_{n-1}}{\Delta t} \\ + \rho_K M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3} \dfrac{u^K_{n+1} + 2u^K_n + u^K_{n-1}}{4} = \Theta_1(u_n, \dfrac{\phi_{n+\frac{1}{2}} + \phi_{n-\frac{1}{2}}}{2}, \dfrac{\psi_{n+\frac{1}{2}} + \psi_{n-\frac{1}{2}}}{2}), \\ M^K \dfrac{\phi^K_{n+\frac{1}{2}} - \phi^K_{n-\frac{1}{2}}}{\Delta t} = \Theta_2(u_n, \dfrac{\phi_{n+\frac{1}{2}} + \phi_{n-\frac{1}{2}}}{2}, \dfrac{\psi_{n+\frac{1}{2}} + \psi_{n-\frac{1}{2}}}{2}), \\ \dfrac{\psi^K_{n+\frac{1}{2}} - \psi^K_{n-\frac{1}{2}}}{\Delta t} = u^K_n. \end{cases}$$

Hence, if we rewrite these equations in an iterative manner, we obtain

$$\begin{cases}
\rho_K(M^K + \Delta t M^K_{\zeta_1+\zeta_2+\zeta_3} + \dfrac{\Delta t^2}{4} M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3}) u^K_{n+1} = \\
\rho_K(2M^K - \dfrac{\Delta t^2}{2} M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3}) u^K_n \\
+ \rho_K(-M^K + \Delta t M^K_{\zeta_1+\zeta_2+\zeta_3} - \dfrac{\Delta t^2}{4} M^K_{\zeta_1\zeta_2+\zeta_1\zeta_3+\zeta_2\zeta_3}) u^K_{n-1} \\
+ \Delta t^2 \Theta_1(u_n, \dfrac{\phi_{n+\frac{1}{2}} + \phi_{n-\frac{1}{2}}}{2}, \dfrac{\psi_{n+\frac{1}{2}} + \psi_{n-\frac{1}{2}}}{2}), \\
M^K \phi^K_{n+\frac{1}{2}} = M^K \phi^K_{n-\frac{1}{2}} + \Delta t \Theta_2(u_n, \dfrac{\phi_{n+\frac{1}{2}} + \phi_{n-\frac{1}{2}}}{2}, \dfrac{\psi_{n+\frac{1}{2}} + \psi_{n-\frac{1}{2}}}{2}), \\
\psi^K_{n+\frac{1}{2}} = \psi^K_{n-\frac{1}{2}} + \Delta t u^K_n.
\end{cases}$$