Full Waveform Inversion with Firedrake

April 2020

1 Purpose

These notes detail the evolving work on using finite elements to perform Full Waveform inversion.

We are using the Firedrake domain specific language [1] to perform a two-dimensional (2D), P-wave acoustic time-domain inversion of the Marmousi 2 model [2]. This work demonstrates the ongoing development of our *hp*-Galerkin seismological Python program called *Spyro*.

Spyro represents a collective effort between workstreams 3 and 4 of the STMI project. It is being developed with WS4's mesh generation approaches, high-order timestepping schemes to reduce the memory storage requirements for the adjoint calculation, Perfectly Matched Layer (PML) boundary conditions to reduce artificial reflections, and a variety of finite element discretizations. All capabilities are designed to function harmoniously and are exposed to the user at a *friendly* and high level of abstraction.

Here we highlight the simplest possible usage case of the software to perform Full Waveform Inversion (FWI) using linear finite elements with a 2^{nd} order in time scheme.

2 Full waveform inversion using Firedrake

2.1 Governing equations

The isotropic, acoustic wave equation (Eq. 1.0) was used to perform the inversion. In this section, we detail the governing equations and how they were discretized in time and space. Briefly, this equation 1.0 can be derived from the isotropic elastic wave equation under the assumption that the shear stress is effectively zero [3]. While being strictly valid only in fluid and gaseous media, the acoustic wave equation is frequently used in active-source full waveform inversion because its numerical solution is computationally inexpensive compared to the solution of the elastic wave equation.

The *strong form* of the acoustic wave equation is:

$$\kappa^{-1}\frac{\partial^2 p(t,\boldsymbol{x})}{\partial t^2} - \nabla\cdot(\rho^{-1}\nabla p(t,\boldsymbol{x})) = -\nabla\cdot(\rho^{-1}q(t,\boldsymbol{x})) + \gamma(\boldsymbol{x}) \text{ in } \Omega \in \mathbb{R}^2 \quad (1.0)$$

$$p(0,\boldsymbol{x}) = 0 \text{ in } \Omega \quad (1.1)$$

$$\frac{\partial p(0,\boldsymbol{x})}{\partial t} = 0 \text{ in } \Omega \quad (1.2)$$

$$\frac{\partial p}{\partial t} + \frac{\partial p}{\partial \mathbf{n}} = 0 \text{ on } \Gamma \in \Omega \quad (1.3)$$

where ρ is density, κ is the bulk modulus, γ is a term representing the absorbing boundary layer, p is the pressure of the P-wave, $q(t,x,y,z)$ is the acoustic source, Γ is defined as the boundary of the mesh, and $\mathbf{n}$ is the unit normal to Γ.

2.2 Spatio-temporal discretization

We arrived at the weak form of equation 1.0 by multiplying by a suitable test function $w : \Omega \to \mathbb{R}$ and then integrating-by-parts once over the spatial derivative term from 1.0 leading to:

$$\int_\Omega w\frac{\partial^2 p}{\partial t^2}dx + \int_\Omega c^2\frac{\partial w}{\partial x}\frac{\partial p}{\partial x}dx = -\int_\Omega c^2 w\nabla\cdot q dx + c^2\int_\Omega w\gamma dx \quad (2)$$

Note that ρ was removed from the spatial derivative (under the assumption it is constant) after integrating-by-parts and was replaced with the P-wave speed $c = \sqrt{\frac{\kappa}{\rho}}$. Also, note that the source term q can be seen as $\tilde{q} = \nabla\cdot q$.

The test function w in equation 2 were chosen to be $P = 1$ linear Lagrange basis functions. However, it's important to point out that the usage of Firedrake enables the selection of variably high-order schemes at the user's discretion.

The algebro-differential (Eq. 2) was temporally discretized such that $t = n\Delta t$ timesteps:

$$\underbrace{\int_\Omega \frac{p^{n+1} - 2p^n + p^{n-1}}{\Delta t^2}w dx}_{\text{mass matrix}} + \underbrace{\int_\Omega c^2\frac{\partial w}{\partial x}\frac{\partial p^{n+1}}{\partial x}dx}_{\text{stiffness matrix}} = -\int_\Omega c^2 w\nabla\cdot q^n dx + \int_\Omega c^2 w\gamma dx \quad (3.1)$$

$$\frac{p^{n+1} - p^{n-1}}{2\Delta t} - \frac{\partial p^n}{d\boldsymbol{n}} = 0 \quad (3.2)$$

The time derivative in equation 2 (mass matrix) was discretized with a central 2^{nd} order accurate finite difference scheme. For the stiffness matrix, we choose to represent the ∇p variable at time $n+1$ for numerical stability.

For the boundary conditions, a non-reflective condition was implemented (Eq. 1.3). Consistent with the other spatio-temporal discretizations, the term was treated implicitly using a second order central difference in time.

The non-reflective Neuman boundary condition approach only damps waves traveling in the normal direction to the boundary so it cannot eliminate boundary reflections altogether. As a result, the domain was partitioned $\Omega = \Omega_{interior} \cup$

Ω_{absorb} with $\Omega_{interior}$ representing the *true* domain and Ω_{absorb} a 1-km width surrounding three sides of the domain (except for the top of the domain) (Fig. 1). The 1-km width of this absorbing boundary layer was chosen through trial-and-error by observing the solutions' reflection.

In Ω_{absorb}, γ was:

$$\gamma = -\alpha(\boldsymbol{x})\frac{\partial p}{\partial t} \tag{4}$$

where the time derivative was discretized again using a 2^{nd} order accurate central finite difference. The damping coefficient α ranged between 0 and 1.0. The α coefficient can vary either linearly or exponentially in the absorbing layer. Note, α was only non-zero in Ω_{absorb}. For this experiment we chose a exponentially decay rate of 0.40.

2.3 Unstructured mesh

To generate a high-geometric quality triangular mesh, we utilized a modified version of the *DistMesh* algorithm. The algorithm was implemented in a mixed language environment (Python + CPP) and is called *SeismicMesh*. This program represents an automatic workflow to build meshes; in other words, the user only supplies a set of input files and parameters and it outputs a simulation-ready mesh and relevant input files. For example, the inputs to the mesh generator were a SEG-y file and a set of user-defined parameters that control mesh sizing heuristics which have been tuned for seismology. The mesh generator algorithm is currently serial, however, a distributed-memory parallel version of this mesh generator is in development for 3D applications in mind.

The mesh was built to resolve an initial P-wave velocity model of the domain. This initial velocity model featured a linear variation in P-wave speed from $4,200$ m/s at the bottom to $1,500$ m/s at the top of the domain (Fig. 1). Specifically, the mesh resolved the wavelength of the P-wave assuming a maximum Ricker source frequency of 20 hz while ensuring the element size respected a maximum Courant number of 0.20. This resulted in a variation in element size (diameter of inscribed circumcirlce of each triangle) ranging from approximately 20 m to 100 m. Further, element size transitions were bounded by solving a Hamilton-Jacobi type equation. We note that this mesh is relatively computationally lightweight considering the size of the domain (e.g., 17 km wide by 3.5 km deep) containing $39,346$ vertices and $77,649$ elements.

In the absorbing layer (i.e., domain extension), the mesh was coarsened to an element size of approximately 80 m. The mesh sizes entering into the border were graded using a Partial-Differential-Equation (PDE)-based approach (following [4]). This PDE-approach preserves mesh sizes inside the domain but relaxes them in the absorbing layer to ensure the transition is sufficiently smooth to avoid numerical artifacts. It's important to point out that this mesh size strategy dramatically reduces the absorbing layers' computational cost as

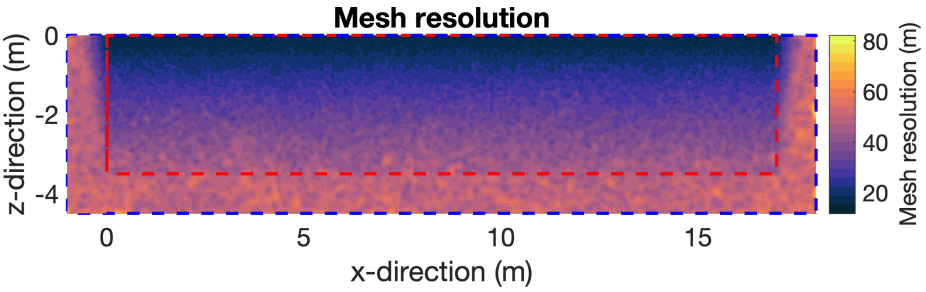

Figure 1: Depicts the mesh, the size of the elemental resolution, along with the absorbing boundary layer. $\Omega_{interior}$ is defined as the area enclosed by the red dotted line. Ω_{absorb} is defined as the area between the red and blue dotted lines.

compared to traditional finite difference methods, which would require a resolution 4x finer resolution (i.e., 20 m). In the border layer, a constant velocity of 1.5 km-s was interpolated. (Fig. 1).

2.4 Optimization

2.4.1 Problem statement

With the numerical aspects clarified, we carry on with the details about how Full Waveform Inversion (FWI) was implemented.

FWI in general can be seen as an optimization problem in which the model parameter (e.g., $\boldsymbol{m}(c)$) is iteratively updated such that the difference in the misfit (Eq. 5) is minimized:

$$\chi(p) = \frac{1}{2}\int_T \sum_{k=1}^{N_p} [p(\boldsymbol{x}_k, t) - p^0(\boldsymbol{x}_k, t)]^2 dt \tag{5}$$

where u^0 is the observed/recorded data from the field campaign at the same set of receiver points $\boldsymbol{x}_k$.

2.4.2 Derivation of the gradient - Discrete

Generally, this optimization problem is solved using a gradient-based method, where the gradient represents how we can optimally change the control variables (here the velocity model) such that the cost functional χ changes the quickest, in a linear fashion. In other words, we are looking for $d\chi/d\mathbf{m} = \nabla_{\boldsymbol{m}}\chi$. We remark we cannot directly compute this field since the cost functional χ depends implicitly on the velocity model under the wave-equation constraint. This requirement falls precisely into the Lagrangian multiplier framework, where we allow the solution (here denoted generically by ϕ) and the velocity model to vary independently and we add another variable (called Lagrangin multiplier or

adjoint, denoted $\phi^\dagger$) so that the wave-equation constraint is satisfied. To do so, we define the Lagrangian:

$$\begin{aligned}\mathcal{L}(\phi,\phi^\dagger,\boldsymbol{m}) = \chi(\phi) + (\phi^\dagger)^{0,T}\cdot(\mathbb{B}_1\phi^1 + \mathbb{B}_0\phi^0 + S_0)\\ + \sum_{n=1}^{N_t}(\phi^\dagger)^{n,T}\cdot(\mathbb{A}_{n+1}\phi^{n+1} + \mathbb{A}_n\phi^n + \mathbb{A}_{n-1}\phi^{n-1} + S_n)\end{aligned} \tag{6}$$

We remark that the equation multiplied with $\phi^{\dagger T}$ are the discrete-version of the wave equation where the matrices $\mathbb{A}_{n+1,n,n-1}$ are combinations of mass and stiffness matrices, depending on the time-scheme employed. The external (or source) term is lumped in S_n, taking into account both initial conditions and volume forcing terms. Notice that a special treatment for the first time-iteration is given so that we can respect the second-order in time accuracy even at $n = 1$.

It is interesting as well to rewrite the cost functional in a discrete manner:

$$\chi(\phi) = \frac{1}{2}\sum_{n=1}^{N_t}\sum_{k=1}^{N_p}[\mathbb{H}_k\phi^n - r_{n,k}]^2 = \frac{1}{2}\sum_{n=1}^{N_t}[\mathbb{H}\phi^n - r_n]^2 \tag{1}$$

where $r_n = \{p^0(\boldsymbol{x}_k, t_n)\}$ is the reference data at time iteration n, evaluated a a set of points $\mathbf{x}_k$. The matrix $\mathbb{H}$ is the so-called measurement operator and outputs, from a given solution ϕ the measurement. For example, if we consider ϕ to be the discrete pressure P, we have that $\mathbb{H}$ is composed by lines of all test functions evaluated at all measurement points, $\phi_k(\mathbf{x}_k)$.

Before deriving the adjoint and the gradient, we rewrite the Lagrangian functional in terms of the concatenated variables such as $\phi = (\phi^0, \phi^1, \phi^2, \phi^3, \cdots)^T$ and $\phi^\dagger = (\phi^{\dagger 0}, \phi^{\dagger 1}, \phi^{\dagger 2}, \phi^{\dagger 3}, \cdots)^T$ as:

$$\mathcal{L}(\phi,\phi^\dagger,\boldsymbol{m}) = \chi + \phi^{\dagger T}\cdot(\mathcal{A}\phi - \mathcal{S}) \tag{2}$$

where:

$$\mathcal{A} = \begin{bmatrix} \mathbb{B}_0 & \mathbb{B}_1 & 0 & 0 & \dots \\ \mathbb{A}_{n-1} & \mathbb{A}_n & \mathbb{A}_{n+1} & 0 & \dots \\ 0 & \mathbb{A}_{n-1} & \mathbb{A}_n & \mathbb{A}_{n+1} & \dots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix}, \tag{3}$$

and, accordingly, $\mathcal{S}$ groups all external information, including initial conditions and volume forcing terms. By taking the variation of this Lagrangian with respect to the adjoint, we recover the equation $\mathcal{A}\phi = \mathcal{S}$, equivalent to equation 3.1. Its variation with respect to the sate leads to:

$$\frac{\partial\mathcal{L}}{\partial\phi}\mathcal{M}\delta\phi = \frac{\partial\chi}{\partial\phi}\delta\phi + \phi^{\dagger T}\cdot\mathcal{A}\delta\phi = (\nabla_\phi\chi + \mathcal{A}^T\phi^\dagger)\cdot\delta\phi = 0 \Rightarrow \mathcal{A}^T\phi^\dagger = -\nabla_\phi\chi \tag{4}$$

where

$$\mathcal{A}^T = \begin{bmatrix} \mathbb{B}_0^T & \mathbb{A}_{n-1}^T & 0 & 0 & \dots \\ \mathbb{B}_1^T & \mathbb{A}_n^T & \mathbb{A}_{n-1}^T & 0 & \dots \\ 0 & \mathbb{A}_{n+1}^T & \mathbb{A}_n^T & \mathbb{A}_{n-1}^T & \dots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix}, \quad \nabla_\phi \chi = \begin{bmatrix} \mathbb{H}^T(\mathbb{H}\phi_0 - r_0) \\ \mathbb{H}^T(\mathbb{H}\phi_1 - r_1) \\ \mathbb{H}^T(\mathbb{H}\phi_2 - r_2) \\ \vdots \end{bmatrix}, \tag{5}$$

which is the linearization of the cost functional with respect to the state. It is worth noticing that when the matrix $\mathbb{H}$ is the operator that "measures" the solution ϕ on the probing locations $\mathbf{x}_k$, its transpose $\mathbb{H}^T$ represents an operator that takes as input information (typically the residual $(\mathbb{H}\phi_n - r_n)$) on the probing points and outputs functions in the whole space. This is the numerical representation of the Dirac functional, evaluated at the points $\mathbf{x}_k$.

For the gradient computation, we take the variation of the Lagrangian with respect to the control parameter, which is implicitly present in the formulation via the matrices $\mathbb{A}_{n-1}, \mathbb{A}_n, \mathbb{A}_{n+1}$ and $\mathbb{B}_0, \mathbb{B}_1$. For simplicity, let us consider that we are dealing with a Leapfrog scheme, where the stiffness matrix (which has the dependency on the velocity model) in only included in the matrices $\mathbb{A}_n$ and $\mathbb{B}_0$. In this scenario, we have that the variation of the Lagrangian with respect to the velocity model is:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \boldsymbol{m}} \mathcal{M} \delta \boldsymbol{m} &= \frac{d\chi}{d\boldsymbol{m}} \mathcal{M} \delta \boldsymbol{m} \\ &= (\phi^\dagger)^{0,T} \cdot (\partial_{\boldsymbol{m}} \mathbb{B}_0 \delta \boldsymbol{m}) \phi^0 + \sum_{n=1}^{N_t} (\phi^\dagger)^{n,T} \cdot (\partial_{\boldsymbol{m}} \mathbb{A}_n \delta \boldsymbol{m}) \phi^n \end{aligned} \tag{6}$$

where $\partial_{\boldsymbol{m}} \mathbb{A}_n$ is variation of the matrix with respect to the velocity model. For the Leapfrog scheme, this quantity is equal to the variation of the stiffness matrix, namely:

$$\partial_{\boldsymbol{m}} \mathbb{A}_n = \partial_{\boldsymbol{m}} \mathbb{B}_0 = \partial_{\boldsymbol{m}} \mathbb{K} \tag{7}$$

where $\mathbb{K}_{i,j} = \int_\Omega c^2 \nabla \psi_i \cdot \nabla \psi_j d\mathbf{x}$, ψ_i being the test/trial functions on the finite-element space. If we expand as well the velocity model with respect to its coefficients $\boldsymbol{m} = c(\mathbf{x}) = \sum_n c_n \psi_n$, we have:

$$\mathbb{K}_{i,j} = \int_\Omega c^2 \nabla \psi_i \cdot \nabla \psi_j d\mathbf{x} = \sum_{n,m} c_m c_n \int_\Omega \psi_n \psi_m \nabla \psi_i \cdot \nabla \psi_j \tag{8}$$

Taking now the variation of the i,j-th coefficient of this matrix with respect to the k-th coefficient c_k, we have:

$$
\begin{aligned}
\partial_{c_k}\mathbb{K}_{i,j}\delta c_k &= \sum_{n,m}(\partial_{c_k}c_m\delta_{k,m}c_n + c_m\partial_{c_k}c_n\delta_{k,n})\delta c_k\int_\Omega \psi_n\psi_m\nabla\psi_i\cdot\nabla\psi_j \\
&= \sum_m 2c_m\delta c_k\int_\Omega \psi_k\psi_m\nabla\psi_i\cdot\nabla\psi_j \\
&= \int_\Omega\left(2\sum_m c_m\psi_m\right)(\psi_k\delta c_k)\nabla\psi_i\cdot\nabla\psi_j \\
&= \int_\Omega 2c(\psi_k\delta c_k)\nabla\psi_i\cdot\nabla\psi_j
\end{aligned}
\tag{9}
$$

which gives that:

$$
\partial_{\boldsymbol{m}}\mathbb{K}_{i,j}\delta\boldsymbol{m} = \partial_c\mathbb{K}_{i,j}\delta c = \int_\Omega 2c\delta c\nabla\psi_i\cdot\nabla\psi_j \tag{10}
$$

leading to the following gradient:

$$
\frac{d\chi}{dc}\mathcal{M}\delta c = \sum_{n=0}^{N_t}(\phi^\dagger)^{n,T}\cdot(\partial_{\boldsymbol{m}}\mathbb{K}\delta\boldsymbol{m})\phi^n = \sum_{n=0}^{N_t}\int_\Omega 2c\delta c\nabla\phi^\dagger\cdot\nabla\phi \tag{11}
$$

Leading to :

$$
\mathcal{M}\frac{d\chi}{dc} = \sum_{n=0}^{N_t}\int_\Omega 2c\delta c\nabla\phi^\dagger\cdot\nabla\phi \tag{12}
$$

By inverting the mass matrix onto the right-hand-side of the above equation, the gradient is derived, necessary component to perform gradient-based optimization.

2.4.3 Derivation of the gradient

Generally this optimization problem is solved using adjoint-based methods where the change in the velocity model $\delta\boldsymbol{m}$ that optimally changes the misfit χ is computed [5].

We derive the adjoint from the Lagrangian formalism. The Lagrangian functional is given by:

$$
\mathcal{L}(p,p^\dagger,\boldsymbol{m}) = \chi(p) + \int_T\int_\Omega p^\dagger\cdot(\partial_{tt}p - \nabla\cdot(\boldsymbol{m}^2\nabla p) - q)d\boldsymbol{x}dt \tag{6}
$$

In this context, the adjoint's purpose is to multiply the equation that establishes the constraint(s) the optimization must obey. Although we write this functional for continuous variables, we adopted a Discretize-and-then-Optimize approach [3].

Spatio-temporal discretized equations are enforced using the Lagrange multiplier. The Lagrange multiplier theory states that the variation of the functional

with respect to any change in the adjoint $p^\dagger$ solution δp must vanish. This remark determines the equations that the adjoint must satisfy.

By taking the Fréchet derivative of this functional, we have:

$$\lim_{\varepsilon\to 0}\frac{\mathcal{L}(p+\varepsilon\delta p,p^\dagger,\boldsymbol{m})-\mathcal{L}(p,p^\dagger,\boldsymbol{m})}{\varepsilon}=\int_\Omega\frac{\partial\mathcal{L}}{\partial p}\delta p \tag{7.0}$$

$$=\int_\Omega\frac{\partial\chi}{\partial p}\delta p+\int_T\int_\Omega p^\dagger\cdot(\kappa^{-1}\partial_{tt}\delta p-\nabla\cdot(\boldsymbol{m}\nabla\delta p))d\boldsymbol{x}dt=0,\ \ \forall\delta u \tag{7.1}$$

and by applying integration by parts to arrive at the equationa above (to be repeated depending on the numerical discretization). Equation 7.0 dictates how the adjoint is solved from the final time step to the initial time step. For this reason, we often refer to the adjoint as the *backward* solution. We note as well that the variation of the Lagrangian with respect to the adjoint variables (that must also vanish) gives the forward (or direct) problem. Lastly, if we take the variation of the Lagrangian with respect to the velocity model:

$$\lim_{\varepsilon\to 0}\frac{\mathcal{L}(p,p^\dagger,\boldsymbol{m}+\varepsilon\delta\boldsymbol{m})-\mathcal{L}(p,p^\dagger,\boldsymbol{m})}{\varepsilon}=\int_\Omega\frac{\partial\mathcal{L}}{\partial\boldsymbol{m}}\delta\boldsymbol{m}:=\int_\Omega\frac{d\chi}{d\boldsymbol{m}}\delta\boldsymbol{m} \tag{8.0}$$

$$=-\int_T\int_\Omega p^\dagger\cdot\nabla\cdot(2\boldsymbol{m}\delta\boldsymbol{m}\nabla p) \tag{8.1}$$

$$=2\int_T\int_\Omega\boldsymbol{m}\nabla p^\dagger\cdot\nabla p\ \delta\boldsymbol{m}-\int_T\int_{\partial\Omega}p^\dagger\boldsymbol{m}\boldsymbol{n}\cdot\nabla p\ \delta\boldsymbol{m} \tag{8.2}$$

$$=2\int_T\int_\Omega\boldsymbol{m}\nabla p^\dagger\cdot\nabla p\ \delta\boldsymbol{m} \tag{8.4}$$

Note that the last boundary term was set to zero as it was observed to pollute the gradient with high values at the first elements bordering the boundary. In other words, the gradient $\nabla_m\chi = d\chi/dm$ is computed by solving the linear system:

$$\int_\Omega\underbrace{\frac{d\chi}{d\boldsymbol{m}}\delta\boldsymbol{m}}_{Gradient}=2\int_T\int_\Omega\boldsymbol{m}\nabla p^\dagger\cdot\nabla p\ \delta\boldsymbol{m},\ \forall\delta m \tag{9}$$

we can see here that the arbitrary variation of the velocity model δm plays the role of the test function in a finite element formulation. Indeed, this is exactly how we solve the system. We remark that, to be able to compute the gradient, we need not only the forward solution p but also the backwards $p^\dagger$ at the same time This means one needs to have them both in memory to compute the gradient. In practice, we store in memory only the forward solution and during the backward propagation of the adjoint we compute at every time-step the contribution on the right-hand-side term in equation 9.

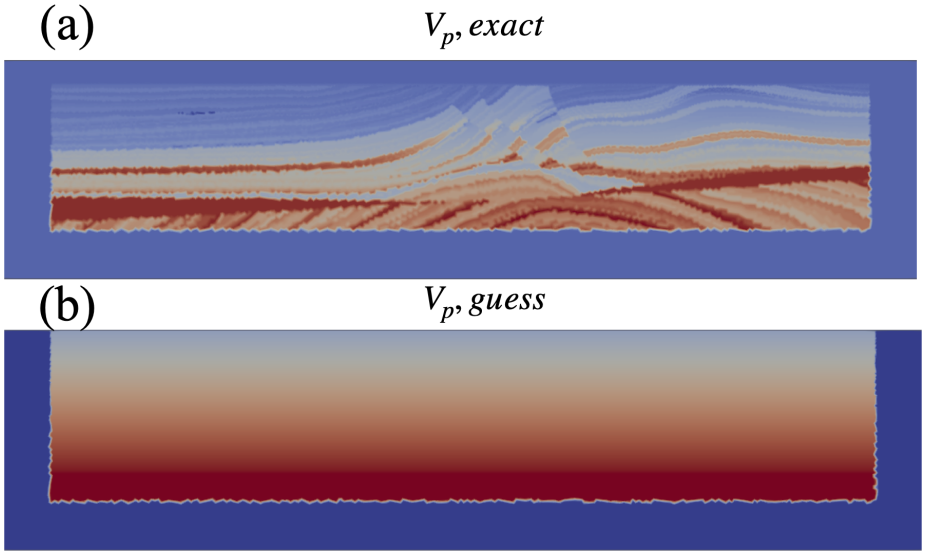

Figure 2: (a) P-wave velocity data for the Marmousi 2 domain and (b) the initial guess both linearly interpolated onto the vertices of an unstructured mesh. Note that the color scales are identical.

2.4.4 Mesh independence

At the end of the adjoint simulation, we perform a mass-matrix inversion to calculate the gradient of the functional. We also remark that the inversion of the mass matrix, as opposed to use the right-hand-side function as the gradient (correlation), is very important since its output is mesh independent (in the sense where mesh refinement will make the gradient converge).

Numerical experiments using both versions of the gradient will be shown later.

2.5 Experimental configuration

To perform the FWI, we used 24 sources equi-spaced in the x-direction located 50-m below the top of the domain. To record the shot, 300 receivers equi-spaced in the x-direction were located 100-m below the surface (Fig. 1). Each source was forced with a Ricker wavelet using a maximum frequency of 3 hz. Note that for this experiment, we used a single frequency band; however, the code supports a multiscale strategy whereby the maximum source frequency is increased in steps using the previous optimization result from the lower frequency wavelet.

Each shot was integrated for 5 seconds with a timestep of 0.005 seconds. The full forward simulation was kept in random access memory (RAM) occupying about 50 gigabtyes of RAM for all forward simulations/shots combined.

The calculation was run on a virtual computer created on Amazon Cloud Services with 24 processors. Each shot was sent to a separate processor acceler-

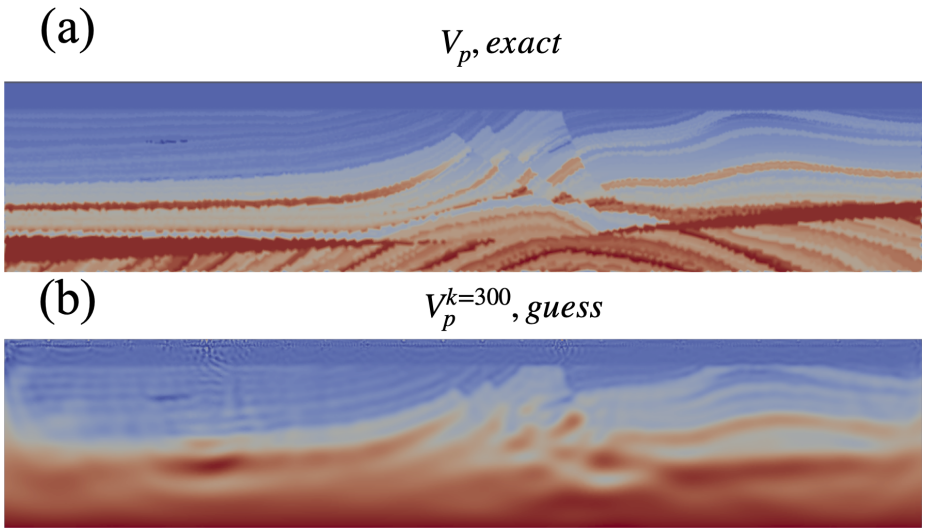

Figure 3: (a) The exact P-wave velocity and (b) after 300 iterations of FWI with *Spyro* from the initial guess depicted in Figure 2(b). Note the absorbing layer is not shown.

ating the calculation of each FWI iteration by a factor of 24. Distributing each shot to each processor is referred to as ensemble parallelism.

To perform the model updates (i.e., non-linear optimization), we used the L-BFGS routine from the Python *SciPy* package with 1 and 10 km velocity bounds. This quasi-Newton optimization method is theoretically 2^{nd} order accurate in the vicinity of a local or global minimum. The *SciPy* wrapper requires the user write a call-back function to return the gradient of the functional and the functional itself.

Due to the synthetic nature of this experiment, we simulated the "observed" shot record data (used to compute the misfit function Eq. **??**) with the same mesh and numerical code as the forward guess.

2.6 Results

The FWI proceeded for 300 iterations, which was a user-defined amount. As is seen in Figure 3, the end result appears qualitatively similar to the Marmousi 2 P-wave velocity model albeit smoother. These features are expected to become better resolved with the usage of higher source frequencies. The total FWI program lasted 15 wall-clock hours.

2.7 Future directions

We realize the work documented here is an initial step forward and relatively simplistic given the constant density acoustic wave equation. In the future, we

would like to incorporate more complex aspects into the numerics and algorithms. Here we detail what aspects of work we would like to improve on:

1. Use observed shot records to evaluate the misfit (Eq. **??**) instead of calculating the synthetic data. This can in part circumvent the inverse crime. Additionally this will also require we perform some kind of low-pass filtering of shot records to enable a practical multiscale FWI strategy since the observed shot records typically contain high-frequency data and noise.

2. Integrating more mesh generation technology directly into the FWI process. Specifically, performing periodic mesh connectivity updates (mesh adaptation) during the FWI iterations. For instance starting with a coarse mesh and refining/coarsening after each frequency band during time-domain FWI.

3. Investigating the performance (both computational and numerical) when using different timestepping schemes and spatial order of accuracy along with variations in the mesh sizes. The timestepping scheme used in FWI problems in the seismological community is almost exclusively 2^{nd} order accurate. However, we note that this is not necessarily memory-efficient in the context of FWI since the full forward simulation needs to be saved. The lack of sufficient memory to store the forward problem is normally circumvented through checkpointing schemes that rely on file input-output.

3 Ongoing Work

Here, we are going to show the new developments and efforts that WS3 and WS4 are facing in order to improve the FWI model. In the next section, our implementation of a perfectly matched layer (PML) is presented, which is based on the mathematical formulation detailed in the work of Kaltenbacher *et al.* [6]. This PML requires only four auxiliary variables in three dimensions inside the absorbing layer, which avoids the need for convolution integrals. In addition, our team formulated a high-order interior penalty discontinuous Galerkin (IPDG) method for the spatial discretization which will be briefly described in this manuscript, so that some steps will be omitted.

3.1 Perfectly matched layer

The second-order acoustic wave equation is considered in a three-dimensional domain Ω_0. In aero-acoustics, this method can approximate an unbounded domain efficiently by minimizing wave reflections on the boundary of Ω_0. A perfectly matched layer, which is a small domain extension Ω_{PML} that attenuates the wave propagation inside it, surrounds Ω_0 truncated by an outer boundary Γ. The computational domain Ω, therefore, is composed by $\Omega_0 \cup \Omega_{PML}$.

The coupled system of partial differential equations for the PML and the initial conditions are:

$$\frac{\partial^2 u}{\partial t^2} + \operatorname{tr}\Psi_1 \frac{\partial u}{\partial t} + \operatorname{tr}\Psi_3 u + \det \Psi_1 \omega - \nabla \cdot (c^2 \nabla u) - \nabla \cdot \mathbf{p} = \bar{f}, \tag{13}$$

$$\frac{\partial \mathbf{p}}{\partial t} + \Psi_1 \mathbf{p} + \Psi_2 (c^2 \nabla u) - \Psi_3 \nabla \omega = 0, \tag{14}$$

$$\frac{\partial \omega}{\partial t} - u = 0, \tag{15}$$

$$u\big|_{t_0} = u_0, \tag{16}$$

$$\partial_t u\big|_{t_0} = v\big|_{t_0} = v_0, \tag{17}$$

$$\mathbf{p}\big|_{t_0} = \mathbf{0}, \tag{18}$$

$$\omega\big|_{t_0} = 0, \tag{19}$$

where u, $\omega : (0,T) \times \Omega \to \mathbb{R}$ and $\mathbf{p}$ $(0,T) \times \Omega \to \mathbb{R}^3$, and the matrices are given by

$$\Psi_1 = \begin{pmatrix} \sigma_x & 0 & 0 \\ 0 & \sigma_y & 0 \\ 0 & 0 & \sigma_z \end{pmatrix}; \Psi_3 = \begin{pmatrix} \sigma_y \sigma_z & 0 & 0 \\ 0 & \sigma_x \sigma_z & 0 \\ 0 & 0 & \sigma_x \sigma_y \end{pmatrix}, \tag{20}$$

$$\Psi_2 = \begin{pmatrix} \sigma_x - \sigma_y - \sigma_z & 0 & 0 \\ 0 & \sigma_y - \sigma_x - \sigma_z & 0 \\ 0 & 0 & \sigma_z - \sigma_x - \sigma_y \end{pmatrix}, \tag{21}$$

and a general Robin boundary condition was applied

$$\beta \frac{\partial u}{\partial t} + c \nabla u \cdot \mathbf{n} = 0, \quad \text{on } \Gamma \times (0,T). \tag{22}$$

The non-reflective (first-order absorbing boundary condition), Neumann, and Dirichlet boundary conditions were carried out in this study in order to determine the most appropriate boundary condition for this type of problem and explore their differences. The coefficient β sets the boundary condition depending of its value. For the non-reflective boundary condition its value is $\beta = 1$, Neumann boundary condition is $\beta = 0$ and Dirichlet boundary condition is $\beta = \infty$.

We can define the domains as $\Omega = (x_1 - l_x, x_2 + l_x) \times (y_1 - l_y, y_2 + l_y) \times (z_1 - l_z, z_2 + l_z)$, $\Omega_0 = (x_1, x_2) \times (y_1, y_2) \times (z_1, z_2)$ and $\Omega_{PML} = \Omega \backslash \Omega_0$. The variables l_x, l_y and l_z are the thickness of the PML layers. The damping function are all positive in Ω_{PML}.

The damping function is based on the work of [7] and is defined for the x-axis as

$$\sigma_x(x) = \begin{cases} 0, & x_1 \leq x \leq x_2. \\ \frac{|x - x_1|^m}{l_x} \sigma_x^{max}, & x_1 - l_x \leq x < x_2, \\ \frac{|x - x_2|^m}{l_x} \sigma_x^{max}, & x_1 < x \leq x_2 + l_x. \end{cases} \tag{23}$$

where m is an exponent which determines the PML spatial growth and σ_x^{max} is a constant which sets the PML maximum value in the x-axis. Similar scaling functions are used for σ_y and σ_z.

For further details about the PML formulation and its two-dimensional reformulation see Baffet *et al.* [8].

3.2 Interior penalty discontinous Galerkin method

The interior penalty discontinuous Galerkin (IPDG) formulation for the acoustic wave equation coupled with the PML is described in this section.

3.2.1 Spatial discretization

The interior penalty discontinuous Galerkin weak formulation of the acoustic wave equation coupled with the PML (13)-(19) and (22) is given by the following statement: find $(u_h, \mathbf{p}_h, \omega_h) \in V_h \times W_h \times Z_h$ as a solution of

$$(\frac{\partial^2 u_h}{\partial t^2}, v)_{\mathcal{T}_h} + (\text{tr}\,\Psi_1 \frac{\partial u_h}{\partial t}, v)_{\mathcal{T}_h} + (\text{tr}\,\Psi_3 u_h + \det \Psi_1 \omega_h, v)_{\mathcal{T}_h} + a_h^{(DG)}(u, v)_{\mathcal{T}_h} + \langle c^2 \nabla u_h, \mathbf{n} \rangle_{\Gamma_h} - (\nabla \cdot \mathbf{p}_h, v)_{\mathcal{T}_h} = (\bar{f}, v)_{\mathcal{T}_h}, \quad (24)$$

$$(\frac{\partial \mathbf{p}_h}{\partial t}, \mathbf{q})_{\mathcal{T}_h} + (\Psi_1 \mathbf{p}_h, \mathbf{q}_h)_{\mathcal{T}_h} + (\Psi_2 (c^2 \nabla u_h), \mathbf{q}_h)_{\mathcal{T}_h} - (\Psi_3 \nabla \omega_h, \mathbf{q}_h)_{\mathcal{T}_h} = 0, \quad (25)$$

$$(\frac{\partial \omega_h}{\partial_t}, \theta)_{\mathcal{T}_h} - (u_h, \theta)_{\mathcal{T}_h} = 0, \quad (26)$$

for all $(v_h, \mathbf{q}_h, \theta_h) \in V_h \times W_h \times Z_h$ and all $t \in (0, T)$, where $\mathcal{T}_h$ is the set of elements e that discretize Ω and the bilinear form is

$$a_h^{(DG)}(u, v) = a_h^{(CG)}(u, v) - a_h^{(D)}(u, v) + S a_h^{(D)}(v, u) + a_h^{(IP)}(u, v)$$

with

$$a_h^{(CG)}(u, v) := \sum_{e \in \mathcal{T}_h} \int_e c^2 \nabla u \cdot \nabla v \mathrm{d}\mathbf{x}, \quad (27)$$

$$a_h^{(D)}(u, v) := \sum_{f \in \mathcal{F}_{h,in} \cup \mathcal{F}_{h,b}} \int_f [[u]] \cdot \{\{c^2 \nabla v\}\} \mathrm{d}\mathbf{s}, \quad (28)$$

$$a_h^{(IP)}(u, v) := \sum_{f \in \mathcal{F}_{h,in} \cup \mathcal{F}_{h,b}} \alpha_h \int_f c^2 [[u]] \cdot [[v]] \mathrm{d}\mathbf{s}, \quad (29)$$

where $f \in \mathcal{F}_h$ represents all faces, $\mathcal{F}_{h,in}$ and $\mathcal{F}_{h,b}$ are the internal faces and the boundary faces on Γ, respectively, α_h is the penalty function and the parameter S takes the values 0, -1 or 1 depending on the particular formulation of IPDG

- $S = 0$ for the Incomplete Interior Penalty (IIPDG) [9]

- $S = -1$ for the Symmetric Interior Penalty (SIPDG) [10]
- $S = 1$ for the Non-symmetric Interior Penalty (NIPDG) [11].

The penalty function was defined as

$$\alpha_h = \frac{P(P+2)}{min_{e\in\mathcal{T}_h} d_e} \tag{30}$$

for tetrahedral elements [12], where d_e is the diameter of the inscribed sphere (or circle). In the two-dimensional case, the penalty was defined as

$$\alpha_h = \frac{P}{2}\frac{(P+1)}{min_{e\in\mathcal{T}_h} d_e} \tag{31}$$

for triangular elements [13].

In order to complete the IPDG formulation, the developments made in some terms of equations (24) and (25) are given below

$$(\nabla\cdot\mathbf{p}_h, v)_{\mathcal{T}_h} = -\sum_{e\in\mathcal{T}_h}\int_e \mathbf{p}\cdot\nabla v \mathrm{d}\mathbf{x} + \sum_{f\in\mathcal{F}_{h,in}\cup\mathcal{F}_{h,b}}\int_f \{\{\mathbf{p}\}\}\cdot[[v]]\mathrm{d}\mathbf{s},$$

$$(\Psi_2(c^2\nabla u_h), \mathbf{q}_h)_{\mathcal{T}_h} = -\sum_{e\in\mathcal{T}_h}\int_e c^2\nabla u\cdot(\Psi_2\mathbf{q})\mathrm{d}\mathbf{x} + \sum_{f\in\mathcal{F}_{h,in}\cup\mathcal{F}_{h,b}}\int_f c^2[[u]]\cdot\{\{\Psi_2\mathbf{q}\}\}\mathrm{d}\mathbf{s},$$

$$(\Psi_3(c^2\nabla\omega_h), \mathbf{q}_h)_{\mathcal{T}_h} = -\sum_{e\in\mathcal{T}_h}\int_e c^2\nabla\omega\cdot(\Psi_3\mathbf{q})\mathrm{d}x + \sum_{f\in\mathcal{F}_{h,in}\cup\mathcal{F}_{h,b}}\int_f c^2[[\omega]]\cdot\{\{\Psi_3\mathbf{q}\}\}\mathrm{d}\mathbf{s}.$$

The average of the function $\{\{\cdot\}\}$ and the function jump $[[\cdot]]$ are defined as follows

$$[[u]] := u^+\mathbf{n}^+ + u^-\mathbf{n}^-, \{\{u\}\} := \frac{1}{2}(u^+ + u-),$$

where $\mathbf{n}^\pm$ denotes the outward pointing normal unit vectors on the boundaries $\mathcal{F}^\pm$.

3.2.2 Temporal discretization

For the temporal discretization, we let $\Delta t > 0$ denote time step size and set $t^n = n\Delta t$. The implicit Newmark time integration scheme was carried out for the temporal discretization for the equation (24), which is a second-order accurate scheme in time [10]. The equations (25) and (26) with a first-order accurate finite difference scheme are:

$$(\frac{\mathbf{p}_h^{n+1}}{\Delta t^n}, \mathbf{q})_{\mathcal{T}_h} = (\frac{\mathbf{p}_h^n}{\Delta t^n}, \mathbf{q})_{\mathcal{T}_h} - (\Psi_1\mathbf{p}_h^n, \mathbf{q})_{\mathcal{T}_h} + (\Psi_2(c^2\nabla u_h^n), \mathbf{q}_h)_{\mathcal{T}_h} - (\Psi_3\nabla\omega_h^n, \mathbf{q}_h)_{\mathcal{T}_h},$$

$$(\frac{\omega_h^{n+1}}{\Delta t}, \theta)_{\mathcal{T}_h} = (\frac{\omega_h^n}{\Delta t}, \theta)_{\mathcal{T}_h} + (u_h^n, \theta)_{\mathcal{T}_h}.$$

Furthermore, when the non-reflective boundary condition were used, a first-order finite difference scheme in time was applied to discretize the boundary integral

$$\langle c^2\nabla u_h, \mathbf{n}\rangle_{\Gamma_h} = \langle c\frac{u_h^{n+1} - u_h^n}{\Delta t}, v\rangle_{\Gamma_h}.$$

3.3 Results

In this section, the preliminary results are presented concerning the IPDG and PML. The Ricker wavelet was used as the source term for the numerical simulation. The starting point for the Ricker wavelet was starting at its peak (i.e., $t = 0$), we considered only half the wave's period.

3.3.1 Boundary condition evaluation

Three different boundary conditions was applied on the outer boundary Γ, in order to understand their differences and find the most appropriate one for a seismology application. Numerical simulations with dirichlet, neumann and non-reflective boundary conditions were performed, which were derived from the general Robin boundary condition (equation (22)). We applied the SIPDG method coupled with PML. The domain used was a square $\Omega = (0, 1) \times (0, 1)$. The two-dimensional mesh was made of regular triangles with $n = 2^7$ elements per side. The polynomial order chose for the interpolation was $P = 3$ (i.e. fourth-order accuracy in space) and equi-spaced quadrature points was adopted. The CFL condition was based on previous studies (Report 2 - WS3), so we chose $q = 0.05$. For the damping parameters, we use $\sigma_x^{max} = 50$ (maximum damping function), and a PML length of $l_x = l_y = 0.1$. The velocity of the P-wave is an unit. The frequency of the Ricker wavelet was $\nu = 40$ hz and was integrated by 0.075 seconds.

Figures 4, 5 and 6 show the numerical simulations performed with dirichlet, neumann and non-reflective boundary conditions, respectively. The measures were made at $y = 0.5$. It can be seen that, before $t = 0.5$ there is no great differences between the boundary conditions, however for $t > 0.5$ the reflections on the boundary of the domain depicted remarkable variations. Nevertheless, it is evident that the non-reflective boundary condition demonstrated to be the best solution as shown in figure 6. It is worth noting that, since the measures were carried out perpendicular to the boundary Γ, the non-reflective boundary condition properly absorbed the waves which touched the borders, hence, here we are not accounting the waves that touch the borders in a non-perpendicular way. In other words, the boundary condition absorbs the waves due to the term $\langle c^2 \nabla u, \mathbf{n} \rangle_\Gamma$. Furthermore, note that, for the dirichlet boundary condition in figure 4, it literally reflects the wave when it touches the border due to its reflective feature.

Regarding the parameter m (exponent), an important aspect to notice is that, high values of m do not absorb the wave as much as low values at least in this homogeneous medium. The best results were observed for $m = 0$ and $m = 1$, however, $m = 0$ presented an early reflection when the wave touches the PML, because the damping function is constant and the wave feels it as a discontinuity in the medium. This reflection appeared for all boundary condition at $t = 0.8$ around $x = 0.0$, although it may disappear with a better discretization or higher polynomial order.

Finally, it should be noted that these conclusions are for this particular

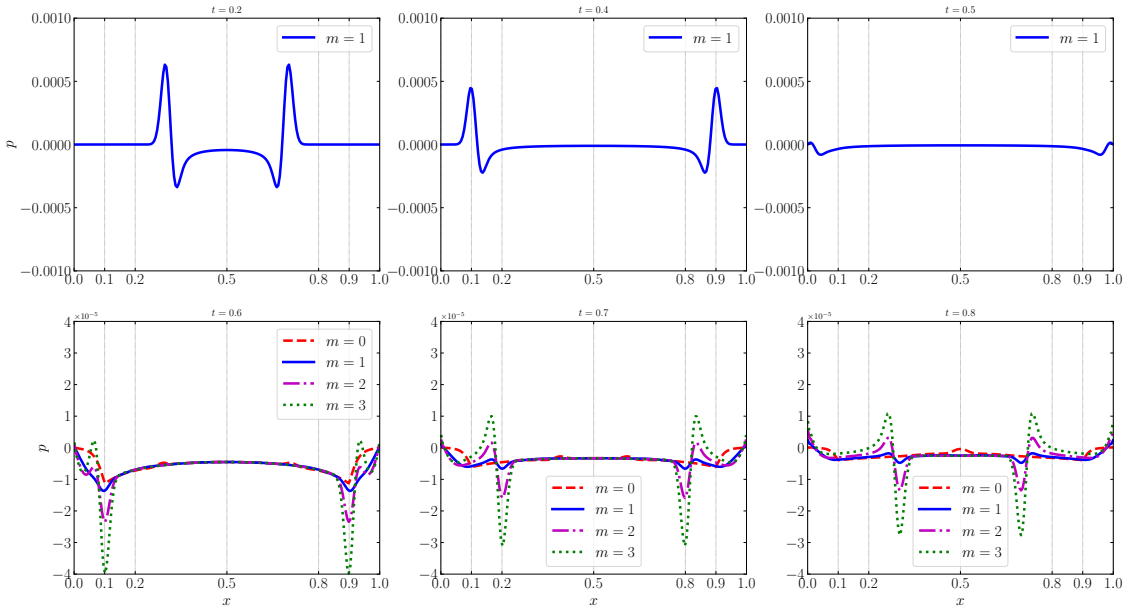

Figure 4: Profiles of the wave amplitude along the line $y = 0.5$ as a function of time. Results were obtained from a two-dimensional numerical simulation of the acoustic wave equation discretized with SIPDG method coupled with PML using Dirichlet boundary conditions.

homogeneous medium, which may not be true for a heterogeneous case. Thus, the next step, in this research, is consider a heterogeneous medium.

3.3.2 Three-dimensional simulation

A numerical simulation was carried out with the SIPDG method for the three-dimensional acoustic wave equation using tetrahedral elements (regular elements) and equi-spaced quadrature points. This experiment was performed with a perfectly matched layer combined with the non-reflective boundary condition on the outer boundary Γ. It was performed in a cubic domain $\Omega = (0,1) \times (0,1) \times (0,1)$ and discretized with a mesh of $n = 2^6$ elements. A value of $q = 0.2$ was used for the CFL condition and P=1 for the polynomial order. For the damping parameters, we use $m = 3$ and $\sigma_x^{max} = 15$, and a PML length of $l_x = l_y = l_z = 0.1$. The velocity of the P-wave is an unit. The frequency of the Ricker wavelet was $\nu = 10$ hz and was integrated by 0.2 seconds. Figure 7 shows the numerical simulations of the acoustic wave discretize with SIPDG method coupled with the PML for four different instants in time. As we can see, the wave propagation almost disappears completely in $T = 0.788$.

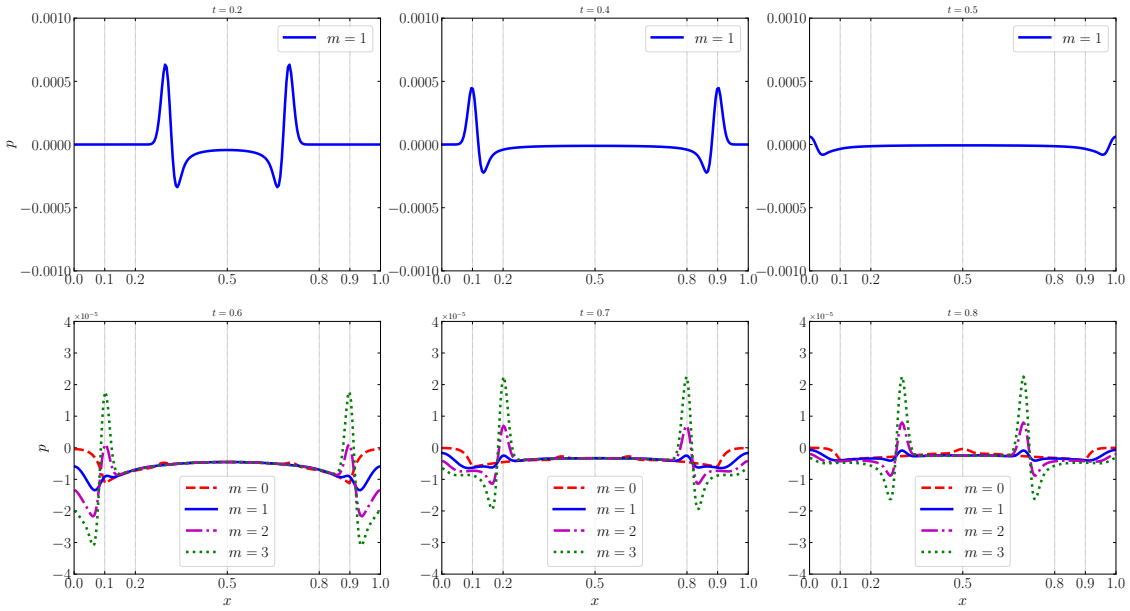

Figure 5: Profiles of the wave amplitude along the line $y = 0.5$ as a function of time. Results were obtained from a two-dimensional numerical simulation of the acoustic wave equation discretized with SIPDG method coupled with PML using Neumann boundary condition.

3.4 Next Steps for PML

In the next steps, we will carry out a seismology applications for our implementation of the PML (i.e., Full Waveform Inversion). In a typical seismic approach, the medium is heterogeneous. This necessitates a special treatment of the velocity model so that it varies sufficient smoothly into the PML region.

In addition, we will evaluate the computational performance in order to study the viability and efficiency of this formulation. We will evaluate the PML formulation using varying layer thicknesses' and adapating the tetrahedral element size into the PML to conserve computational resources by reducing the total number of DoF. These results may give us guidelines for how to setup the forward problem for example, to perform Full-Waveform Inversion.

We notice that an important step for the FWI using the PML is its adjoint since the extra equations added to the problem make it no longer self adjoint. Just as in the case where no PML was considered, we employ the discrete adjoint. We notice that the formalism presented before (section 2.4.1) is still applicable if now our system has more state variables, grouped in the vector $\psi = (u, \boldsymbol{p})$ in 2D or $\psi = (u, \boldsymbol{p}, \omega)$ in 3D.

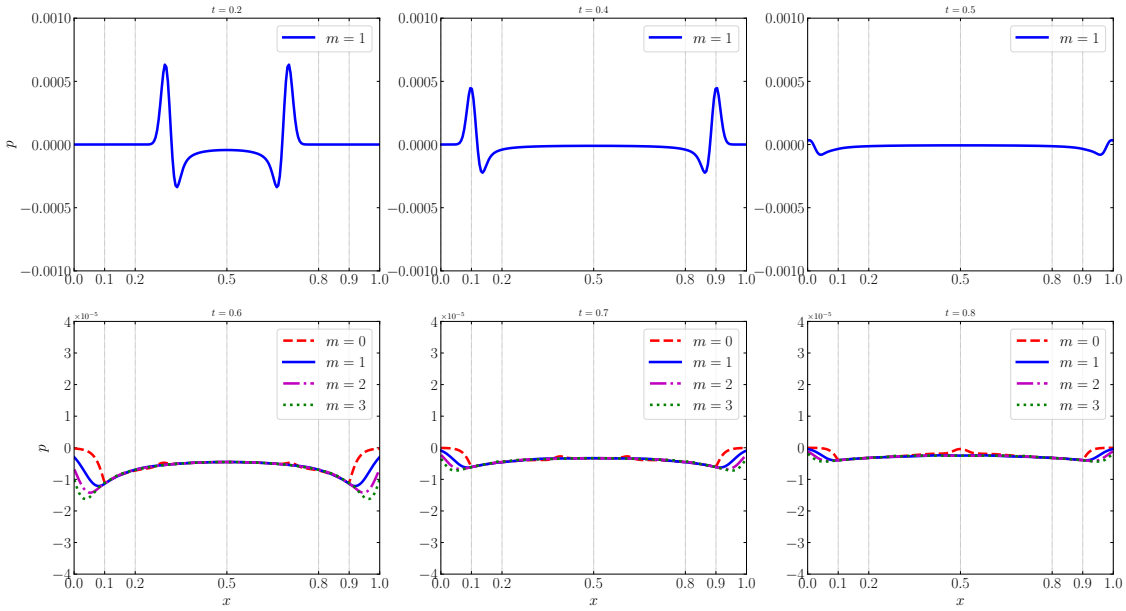

Figure 6: Profiles of the wave amplitude along the line $y = 0.5$ as a function of time. Results were obtained from a two-dimensional numerical simulation of the acoustic wave equation discretized with SIPDG method coupled with PML using non-reflective boundary condition.

References

[1] F. Rathgeber, D. A. Ham, L. Mitchell, M. Lange, F. Luporini, A. T. T. Mcrae, G.-T. Bercea, G. R. Markall, and P. H. J. Kelly, "Firedrake: Automating the finite element method by composing abstractions," vol. 43, no. 3, 2016.

[2] G. S. Martin, R. Wiley, and K. J. Marfurt, "Marmousi2: An elastic upgrade for marmousi," *The Leading Edge*, vol. 25, no. 2, pp. 156–166, 2006.

[3] A. Fichtner, *Full Seismic Waveform Modelling and Inversion*. 01 2011.

[4] P.-O. Persson, "Pde-based gradient limiting for mesh size functions," in *IMR*, 2004.

[5] J. Virieux and S. Operto, "An overview of full-waveform inversion in exploration geophysics," *Geophysics*, vol. 74, no. 6, pp. WCC1–WCC26, 2009.

[6] B. Kaltenbacher, M. Kaltenbacher, and I. Sim, "A modified and stable version of a perfectly matched layer technique for the 3-d second order wave equation in time domain with an application to aeroacoustics," *Journal of Computational Physics*, vol. 235, pp. 407–422, 2013.

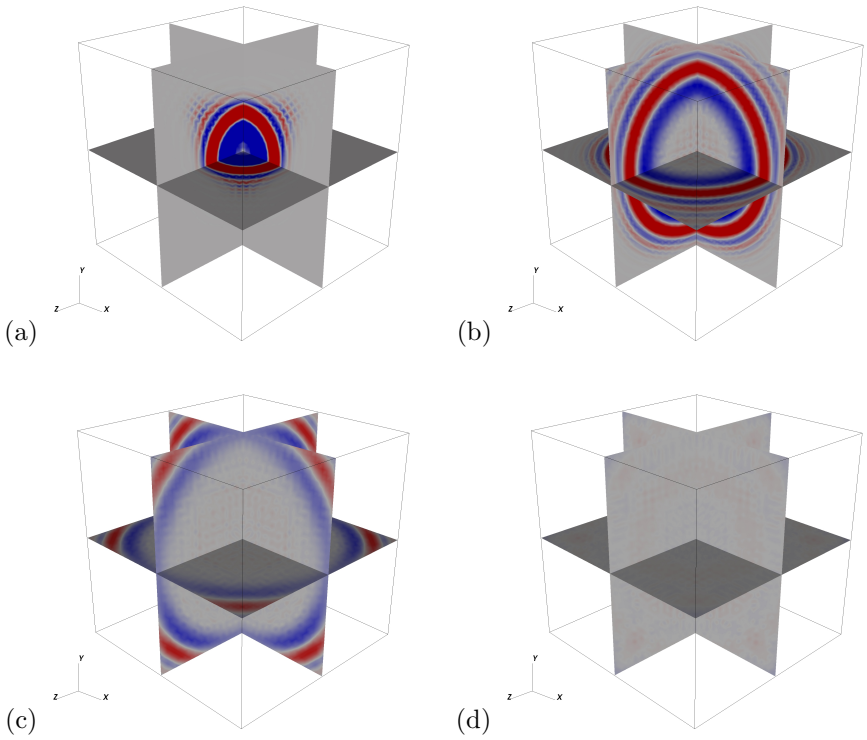

Figure 7: Numerical simulation of the acoustic wave equation discretized with SIPDG method and coupled with PML. (a) T=0.20, (b) T=0.396, (c) T=0.592, (d) T=0.788.

[7] M. Stanglmeir, N. C. Nguyen, J. Peraire, and B. Cockburn, "An explicit hybridizable discontinuous galerkin method for the acoustic wave equation," *Computer Methods in Applied Mechanics and Engineering*, vol. 300, pp. 748–769, 2016.

[8] D. H. Baffet, M. J. Grote, S. Imperiale, and M. Kachanovska, "Energy decay and stability of a perfectly matched layer for the wave equation," *Journal of Scientific Computing*, vol. 81, pp. 2237–2270, 2019.

[9] C. Dawson, S. Sun, and M. F. Wheeler, "Compatible algorithms for coupled flow and transport," *Comput. Methods Appl. Mech. Eng.*, vol. 193, pp. 2565–2580, 2004.

[10] M. J. Grote, A. Schneebeli, and D. Schötzau, "Discontinuous galerkin finite element method for the wave equation," *SIAM Journal on Numerical Analysis*, vol. 44, no. 6, pp. 2408–2431, 2006.

[11] B. Rivière, M. Wheeler, and V. Girault, "Improved energy estimates for interior penalty, constrained and discontinuous galerkin methods for elliptic problems. part 1," *Computational Geosciences*, vol. 3, no. 3, pp. 337–360, 1999.

[12] S. Geevers, W. A. Mulder, and J. J. W. van der Vegt, "Dispersion properties of explicit finite element methods for wave propagation modelling on tetrahedral meshes," *Journal of Scientific Computing*, vol. 77, pp. 372–396, 2018.

[13] C. Agut and J. Diaz, "Stability analysis of the interior penalty discontinuous galerkin method for the wave equation," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 47, pp. 903–932, 2013.