UNIVERSITÀ DEGLI STUDI DI MILANO

Dottorato di Ricerca in Scienze della Terra
Ciclo XXX

Modelling and optimization techniques for acoustic Full Waveform Inversion in seismic exploration

Ph.D. Thesis

Bruno Giovanni Galuzzi
Matricola R10913-R28

Tutor
Prof. Eusebio Stucchi

Academic Year
2017-2018

Coordinator
Prof. ssa Elisabetta Erba

Co- Tutor
Prof. ssa Elena Zampieri

Abstract

Full Waveform Inversion has become an important research field in the context of seismic exploration, due to the possibility to estimate a high-resolution model of the subsurface in terms of acoustic and elastic parameters. To this aim, issues such as an efficient implementation of wave equation solution for the forward problem, and optimization algorithms, both local and global, for this high non-linear inverse problem must be tackled. In this thesis, in the framework of 2D acoustic approximation, I implemented an efficient numerical solution of the wave equation based on a local order of approximation of the spatial derivatives to reduce the computational time and the approximation error. Moreover, for what concerns the inversion, I studied two different global optimization algorithms (Simulated Annealing and Genetic Algorithms) on analytic functions that represent different possible scenarios of the misfit function to estimate an initial model for local optimization algorithm in the basin of attraction of the global minimum. Due to the high number of unknowns in seismic exploration context, of the order of some thousands or more, different strategies based on the adjoint method must be used to compute the gradient of the misfit function. By this procedure, only three wave equation solutions are required to compute the gradient instead of a number of solutions proportional to the unknown parameters.

The FWI approach developed in this thesis has been applied first on a synthetic inverse problem on the Marmousi model to validate the whole procedure, then on two real seismic datasets. The first is a land profile with two expanding spread experiments and is characterized by a low S/N ratio. In this case, the main variations of the estimated P-wave velocity model well correspond to the shallow events observed on the post-stack depth migrated section. The second is a marine profile extracted from a 3D volume where the local optimization, based on the adjoint method, allows to estimate a high-resolution velocity model whose reliability has been checked by the alignment of the CIGs computed by pre-stack depth migration.

Summary

Introduction

Full Waveform Inversion (FWI) represents an essential variant of seismic tomography for the estimation of a high-resolution subsurface model, through a minimization procedure on a misfit function, which measures the differences between the observed and the modelled seismograms. These are given by the numerical solution of the seismic wave equation, requiring a high computational cost if the simulation of seismic wave propagation is made in complex 2D and 3D Earth structures. The minimization procedure consists of any optimization algorithm able to reach the global minimum of the misfit function, despite the high number of possible unknows and the non-linearity relation between the seismic data and the subsurface model parameters.

During my Ph.D. I have investigated three central topics regarding FWI in case of 2D acoustic approximation and in the context of seismic exploration. The first topic concerns the seismic modelling, in particular, the implementation of an efficient algorithm to simulate synthetic seismograms, using a numerical approach with low numerical dispersion and low computational time. The second regards the implementation of the adjoint method to compute the gradient of a misfit function, for a local optimization procedure, including some basic processing operations on the synthetic data. The third deals with the possibility to use global optimization strategies to determinate a suitable starting model for local FWI.

As demonstration of the FWI feasibility, I have applied the FWI procedure for the estimation of an acoustic macro-model of the subsurface for three very different scenarios: a classical problem of FWI on synthetic marine data, a FWI on real land data, and a FWI on real marine data.

The thesis is divided into nine Chapters and an Appendix. Chapter 1 introduces the Full Waveform Inversion in the context of seismic exploration, underlining the limits and the difficulties of its applications.

Chapter 2 presents the new numerical scheme I implemented to model the generation and propagation of seismic waves in the subsurface under a 2D acoustic approximation assumption, whereas Chapter 3 describes an efficient numerical scheme to compute the gradient of a misfit function using the adjoint method.

A general description of the software for the modelling and the gradient computation is given in Chapter 4, with more details of the implementation in Appendix A.

Chapter 5 is intended to introduce the study I have done on the global optimization, and in particular on two global optimization algorithms: the Adaptive Simulated Annealing (ASA) and the Genetic algorithms (GA). In Chapter 6, both algorithms have been used to find the global minimum of some analytic test functions, that represent very different scenarios of the misfit function.

In the last chapters, I present the results of three different FWI experiences. Chapter 7 illustrates the results obtained by the application of ASA and GA algorithms for a classical FWI problem, which consists of the inversion of synthetic seismograms of the Marmousi model.

Chapter 8 deals with the results of an experience of acoustic FWI on a 2D seismic line data acquired onshore, using the GA algorithm for the estimation of a smooth velocity model of the area of interest, and a subsequent local minimization.

Finally, in Chapter 9 a local FWI procedure is applied on the data of a 2D marine seismic profile to estimate an acoustic velocity field for a pre-stack depth migration of the seismic data.

Chapter 1.
Full Waveform Inversion

The first chapter is intended to introduce the Full Waveform Inversion in the context of seismic exploration.

1.1 Seismic exploration

Seismic exploration is an active geophysical exploration method, which has the aim of obtaining structural subsurface information from active seismic data. The main application of this method is in oil and gas industry, in search of oil and gas reservoirs, but many other applications exist such as the search of geothermal reservoirs, site investigation for civil engineering, and the exploration of mineral deposits and underground water supplies. For a complete and exhaustive description of seismic exploration, it is possible to see, for example, [1] and [2].

In a **seismic acquisition** the seismic waves created by a controlled **source**, that generates a band-limited time-varying impulse signal, propagate through the subsurface. Some of them return to the surface as refractions or reflections from geological boundaries within the subsurface. The recording instruments, or **receivers**, distributed on the surface along a line (2D seismic) or a grid (3D seismic), detect these returning waves and measure the arrival times and the amplitudes of the waves at a different distance, or **offset,** from the source. The record of a single receiver is called **seismic trace**, whereas the **seismogram** is classically the set of all the seismic traces relative to the same shot. Finally, the seismic data are processed and converted into depth information to map the distribution of subsurface geological structures.

In general, we distinguish between a **land acquisition** and a **marine acquisition**. In the first case, the seismic source produces elastic energy that propagates into the Earth, and the receivers are distributed on the ground to detect the ground motion caused by the returning waves. The source can be explosive (**dynamite, seismic gun**) or non-explosive (**vibroseis**). The receivers, called **geophones**, produce an output voltage proportional to the velocity or acceleration of the ground particles as the wave passes. In the second case, the source and receivers are towed behind a vessel, in the water. The source radiates acoustic energy into the water, which propagates down to the geological structures located under the sea floor. The receivers are distributed near the sea surface to detect the pressure waves caused by the energy of the returning waves. The source is usually one or more **air-guns**, and the receivers are **hydrophones** that can produce an output voltage proportional to pressure variations associated with the passage of compressional seismic waves through the water. Figure 1.1 displays a sketch of a marine seismic acquisition from [3].

Figure 1.2a and Figure 1.2b show two raw seismograms, made on land and at sea, respectively. The land one shows the presence of strong noisy events, that depends on the ground-roll or surface waves that propagate along the surface, and can hide the reflections events. The sea one is cleaner than the previous one, but the multiple strong waves are present that can hide much information.

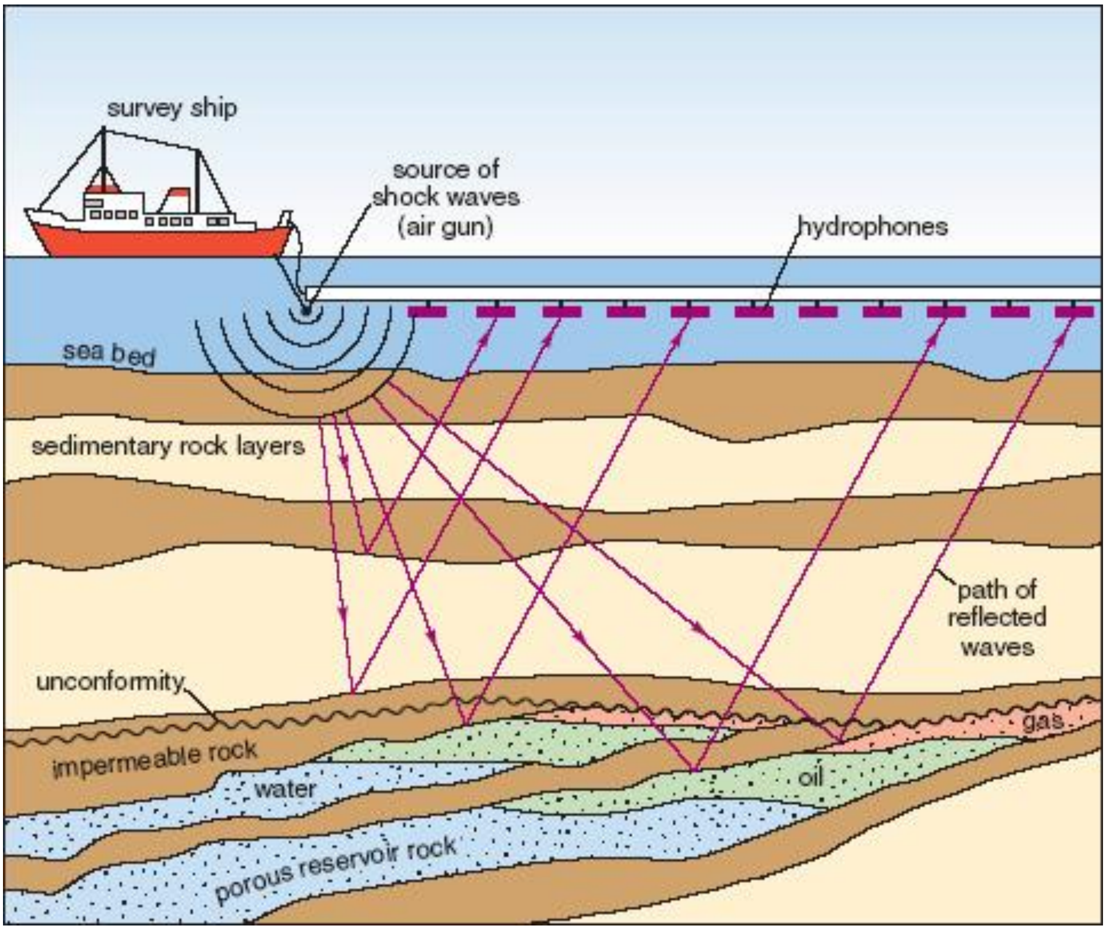


Figure 1.1: Example of a marine seismic acquisition from [3].

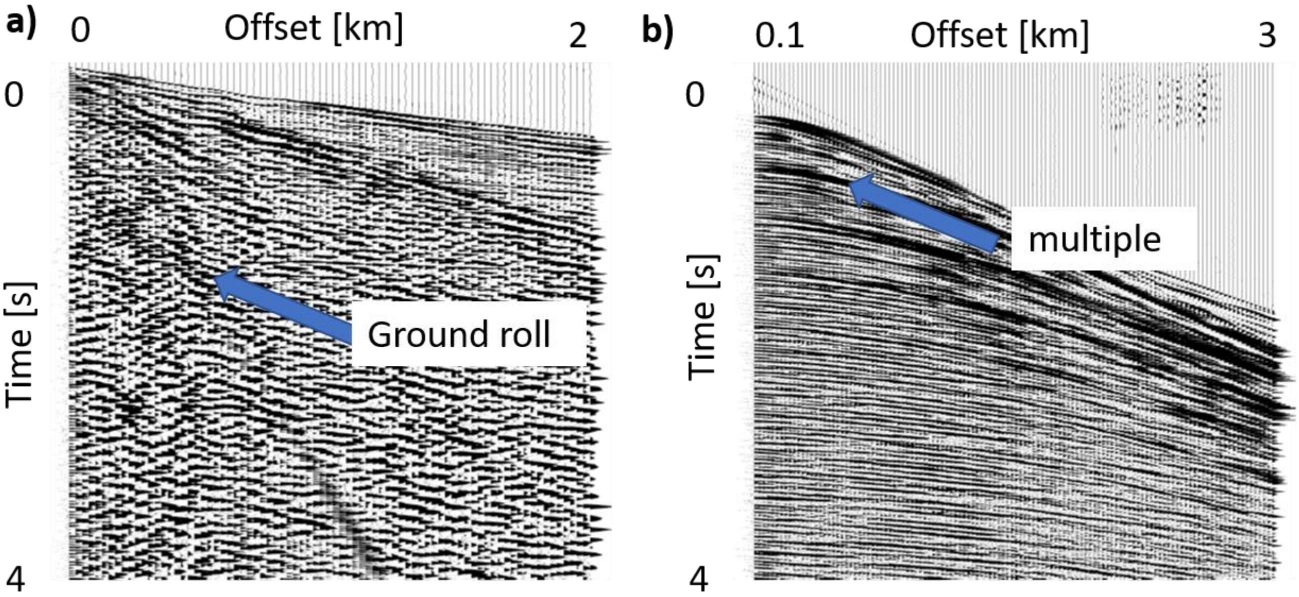


Figure 1.2: Example of: a) raw land seismogram; b) raw marine seismogram.

1.2 What is the Full Waveform Inversion?

When the elastic properties of the rocks in the subsurface are highly variable, it is difficult to estimate information from the seismic data, using classical tomography techniques based on ray theory approximation. Vice versa, if we have a perfect knowledge of the elastic properties of the subsurface, it is possible to simulate the synthetic seismograms of a seismic acquisition, by solving the numerical solution of the wave equation.

Full Waveform Inversion (FWI) is a seismic imaging technique that estimates a high-resolution model of the subsurface by a **minimization** of a **misfit functional** $F(m)$. This function measures the difference between **observed seismograms**, obtained during a seismic acquisition, and the **predicted seismograms** obtained by the numerical solution of the seismic wave equation (acoustic or elastic). This approach was proposed about 30 years ago [4], but it has gained success recently, thanks to the enormous improvements in the computational time and memory storage of computers, that nowadays can model the propagation of seismic waves through 2D and 3D Earth structures, e.g., basins and reservoirs. Recent reviews of FWI can be found in [5] and in [6].

1.3 FWI as a minimization problem

Figure 1.3 displays a possible scheme of the algorithm FWI, characterized by an iterative procedure that is based on:

- a **modelling algorithm**, to simulate predicted seismograms using the numerical solution of the wave equation;
- a **misfit function** to measure the difference between the predicted and the observed data;
- a **stopping criterium**, to determine when to stop the iterative procedure;
- an **inversion algorithm** step**,** to estimate the predicted model that minimizes the misfit function.

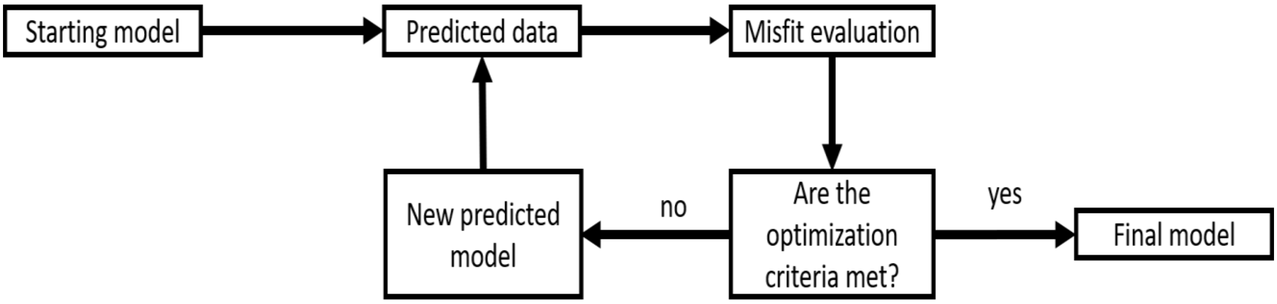


Figure 1.3: Full Waveform Inversion iterative procedure.

The modelling algorithm depends on the type of seismic wave propagation we want to simulate (elastic or acoustic? Isotropic or anisotropic?1D, 2D or 3D?) and will be discuss in the next chapter.

The misfit function represents some function quantifying the difference between the observed and the modelled data. It reads:

$$\overline{m} = arg \min_{m \in M} \hat{F}(u(m), u_0), \tag{1.1}$$

where M is the set of all possible geological models, u_0 is the observed data, and $u(m)$ is the predicted data. The Earth model m consists of a set of geophysical properties such as the spatial distribution of the P-wave velocity $v_p(\boldsymbol{x})$, the S-wave velocity $v_s(\boldsymbol{x})$ and the density $\rho(\boldsymbol{x})$.

In the context of FWI different misfit functions can be chosen based, e.g., on differences between waveforms, travel times, amplitudes, envelopes, and phases. See [7] and [8] for the description and application of many misfit functions. However, a classical one remains the **L^2-norm difference** between the observed and the synthetic seismogram [4] [9]:

$$\hat{F}(m) = \frac{1}{2}\sum_{r=1}^{n_r}\left(\int_0^T [u(m,\boldsymbol{x},t) - u_0(\boldsymbol{x},t)]^2 \delta(\boldsymbol{x} - \boldsymbol{x}^r)\, dt\right), \tag{1.2}$$

where n_r is the number of receivers, and $\{\boldsymbol{x}^r\}$ represent their positions. The integral in (1.2) is approximated by the following quadrature formula:

$$\hat{F}(m) \approx F(m) = \frac{1}{2}\sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}[u(m,\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)]^2\ \delta(\boldsymbol{x} - \boldsymbol{x}^r)\right) dt, \tag{1.3}$$

where n_t is the number of trace samples, and dt is the time sampling.

However, some processing steps such as band-pass filtering, windowing operations, and gain recovery operations can be applied to the seismograms before the building of the misfit functions to isolate and invert only specific events. For example, in [10] a low-pass filter and a time windowing on the diving waves for a FWI on marine data are used.

In Chapter 2 we will observe that the space of propagation of the seismic waves must be discretized on a modelling grid to use a modelling algorithm. This fact causes the discretization of geological models, and a FWI problem becomes a minimization problem with the number of possible unknowns given by the number of nodes in the modelling grid multiplied by the number of geophysical properties we consider for each node.

The stopping criterion consists of a practical test to determine when to stop the iterative procedure, such as a threshold value of the misfit function or a maximum number of iterations.

1.4 Local inversion algorithm for FWI

Since $F(m)$ is a complicated non-linear function of the model m, an iterative minimization procedure must be used to obtain the **global minimum** $\overline{m}$. Starting from a plausible **initial model** m^0, an iterative minimization updates the current model m^k, to a new model m^{k+1}, given by

$$m^{k+1} = m^k + \bar{\gamma}_k h_k \tag{1.4}$$

where $F(m^{k+1}) < F(m^k)$, and $h_k \in \mathbb{R}^n$ and $\bar{\gamma}_k \in \mathbb{R}^+$ are the **descend direction** and the **step length**, respectively. For a local descend direction we must have

$$h_k \cdot \nabla_m F(m^k) < 0, \tag{1.5}$$

where $\nabla_m F(m^k)$ represents the gradient of the misfit function $F(m)$ and " $\cdot$ " denotes the scalar (or inner) product between h_k and $\nabla_m F(m^k)$. Table 1.1 lists some possible descend directions, classically used in the context of local optimization algorithms. A complete and detailed description of them is in [11].

Algorithm	h_k
a) Steepest descend method	$-\nabla_m F(m^k)$
b) Newton's method	$-H_F(m^k) \cdot \nabla_m F(m^k)$
c) Conjugate gradient method	$-\nabla_m F(m^k) + \beta_k h_{k-1}$, *with* $\beta_k \geq 0$

*Table 1.1: Some possible descend directions used for local optimization algorithms. a) The **Steepest descend method** uses the gradient of the misfit function at m^k as descend direction. b) **The Newton's method** uses also the knowledge of Hessian matrix $H_F(m^k)$ at m^k. c) The **Conjugate gradient method** uses a combination of the gradient of the misfit function at m^k and the previous descend direction h_{k-1}.*

The step length $\bar{\gamma}_k$ should be chosen such that $F(m^k + \bar{\gamma}_k h_k)$ is minimal along the direction h_k. However, to assure the convergence of the algorithm it is sufficient to find a step length $\bar{\gamma}_k$ that satisfies these two following conditions, called strong **Wolfe conditions** [12]:

$$\begin{cases} F(m^k + \bar{\gamma}_k h_k) < F(m^k) + c_1 \bar{\gamma}_k \nabla_m F(m^k) \cdot h_k \\ |\nabla_m F(m^k + \bar{\gamma}_k h_k) \cdot h_k| < c_2 |\nabla_m F(m^k) \cdot h_k| \end{cases} \tag{1.6}$$

with $0 < c_1 < c_2 < 1$ [11].

The first condition assures a sufficient decrease of the misfit function along the direction h_k, whereas the second one that the slope has been reduced sufficiently.

1.5 Limits and difficulties of local FWI

Solving a FWI problem by local optimization algorithms can be a good strategy. However, many difficulties arise in such a problem, such as:

1. no a priori information about the structure of the misfit function;
2. high number of unknows of the optimization problem;
3. high computation cost of a misfit evaluation;
4. a non-trivial way to obtain local information of the misfit function, such as the value of gradient or the Hessian;
5. the presence of many local minima for the misfit function;

The first difficulty is related to the fact that there is no explicit analytic expression for the misfit function as a function of the model parameters.

The second one is related to the fact that a high number of parameters can be necessary to describe the variations of the physical properties of complex geological structures.

The third difficulty is related to the fact that to evaluate the misfit function for a model m, is necessary to solve the wave equation by using a modelling algorithm, whose computational time depends on the size of the problem, but also on the efficiency of the numerical method used.

The fourth problem is related to the fact that it is not possible to compute the gradient of the Hessian of the misfit function in a reasonable time, using some finite difference approximation formula, because the number of unknowns is large, and the solution of the wave equation is computationally expensive. The last one is because seismic inverse problems such as FWI are characterized by non-linear relation between a model m and the synthetic seismograms $u(m)$. Besides, the well-known **cycle skipping effect** causes the fact that, if the starting model produces predicted seismograms very different from the observed ones, then the local optimization could converge to a **local minimum** rather than the global one. From a mathematical point of view this means that, if the starting model m_0 of a local procedure is not in the basin of attraction of the global minimum, the algorithm cannot converge to the global minimum, but rather to a local one [13]. Figure 1.4 displays an example of possible misfit function with a high number of local minima, in which a local procedure converges to the global minimum, starting from the green points in the basin of attraction of the global minimum. On the other hand, it converges to a local minimum, if it starts from the red points, which are not in the basin of attraction of the global minimum.

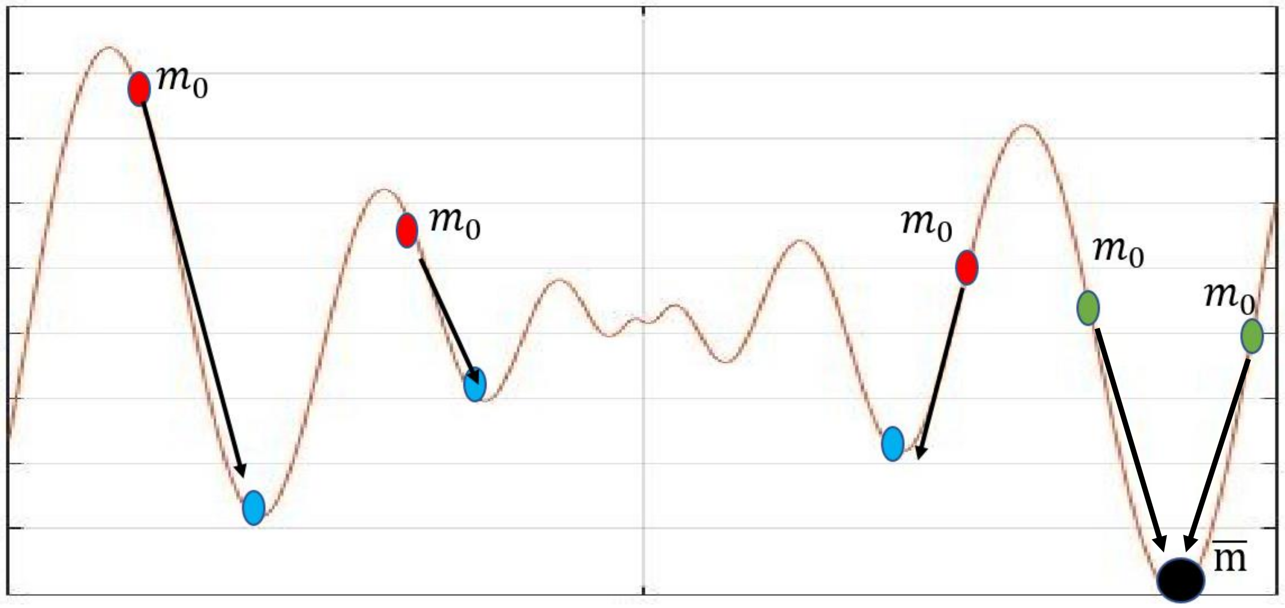


Figure 1.4: Example of misfit function with many local minima. If the starting model m_0 is outside the basin of attraction of global minimum $\overline{m}$, the local algorithm converges to a local minimum.

All these facts make the FWI problem rich of challenges, many of which have not been resolved yet.

1.6 My research about FWI

During my Ph.D. I have studied some of the central aspects regarding FWI, in case of 2D acoustic approximation, such as:

1. the study and the implementation of an efficient modelling algorithm to simulate synthetic seismograms, using a new numerical approach with low numerical dispersion and low computational time, that includes the possibility to execute some basic processing operations on the data (Chapter 2 and Chapter 4);
2. the study and the implementation of an efficient tool to compute the gradient of a misfit function using the well-known adjoint method, but that also includes the possibility to execute some basic processing operations on the synthetic data (Chapter 3 and Chapter 4);

3. the study (Chapter 5) and the comparison (Chapter 6) of the performances of the two global optimization algorithms on finding the global minimum of some analytic objective functions;
4. the application of global optimization procedure for a classical problem of FWI on synthetic seismograms (Chapter 7);
5. the application of global optimization procedure for a FWI on a real land data (Chapter 8);
6. the application of local optimization procedure for a FWI on real marine data (Chapter 9).

Chapter 2.
Numerical implementation of the seismic wave equation for acoustic FWI

In this chapter, I present the essential elements to model the generation and propagation of seismic waves in the subsurface. Under the hypothesis of a 2D acoustic approximation, I describe the numerical scheme I implemented. See [14] for more details.

2.1 Modelling the seismic wave equation

In this section, the acoustic wave equation is introduced as a particular case of the elastic one. For a complete and exhaustive description of the mathematical model of the seismic wave propagation, the reader can refer, for example, [15].

2.1.1 Elastic wave equation

The propagation of seismic waves in the Earth can be modelled by the **elastic wave equation**

$$\ddot{\boldsymbol{u}}(\boldsymbol{x},t) - \rho(\boldsymbol{x})^{-1}\nabla\cdot\boldsymbol{\sigma}(\boldsymbol{x},t) = \rho(\boldsymbol{x})^{-1}\boldsymbol{f}(\boldsymbol{x},t), \quad x \in D \subset \mathbb{R}^3,\ \ t \in [0,T] \subset \mathbb{R}^+, \tag{2.1}$$

that relates the **displacement field** u in a region of the Earth D to its **density** ρ, the **stress tensor** σ, and an **external force** f, representing the source of seismic waves. At the surface ∂D, the normal components of the stress tensor $\boldsymbol{\sigma}$ is zero, that is

$$\boldsymbol{\sigma}(\boldsymbol{x},t)\cdot\boldsymbol{n} = \boldsymbol{0}, \tag{2.2}$$

where $\boldsymbol{n}$ is the unit normal on ∂D. This condition is called **free surface boundary condition**.
As initial condition, both the displacement field $\boldsymbol{u}$ and the velocity field $\dot{\boldsymbol{u}}$ are zero before $t = 0$,

$$\boldsymbol{u}_{t\leq 0} = \dot{\boldsymbol{u}}_{t\leq 0} = \boldsymbol{0}, \tag{2.3}$$

that is when the source $\boldsymbol{f}$ starts to act. The stress tensor $\boldsymbol{\sigma}$ is related to the displacement vector $\boldsymbol{u}$ by the **linear visco-elastic equation**

$$\boldsymbol{\sigma}(\boldsymbol{x},t) = \int_{-\infty}^{+\infty} \dot{\boldsymbol{C}}(\boldsymbol{x},t-t') : \nabla\mathbf{u}(\mathbf{x},t')\,dt', \tag{2.4}$$

where $\boldsymbol{C}$ is the 4th-order **elastic tensor**, with 21 independent components and $\boldsymbol{C}(t)_{t\leq 0} = \boldsymbol{0}$. In the previous equation " : " denotes the tensor product between $\dot{\boldsymbol{C}}$ and the displacement gradient $\nabla\mathbf{u}$.
Equations (2.1) and (2.4) are very general and include several effects, such as anisotropic effects and visco-elastic dissipation. See [16] for more details about the theory of elasticity. However, in macroscopic scale, the Earth or a region D of our interest can be sufficiently well described as an isotropic non-dissipative medium. In this case, the equation (2.4) becomes

$$\sigma_{i,j}(\boldsymbol{x},t) = \delta_{i,j}\left(\kappa(\boldsymbol{x}) - \tfrac{2}{3}\mu(\mathbf{x})\right)\nabla\mathbf{u}(\mathbf{x},\mathrm{t}) + 2\mu(\mathbf{x})\varepsilon_{i,j}, \tag{2.5}$$

where $\delta_{i,j}$ is the Kronecker delta symbol, the quantity $\varepsilon_{i,j} = \left(\frac{\partial u_j(x,t)}{\partial x_i} + \frac{\partial u_i(x,t)}{\partial x_j}\right)$ is called **strain tensor**, κ is the **Bulk modulus**, and μ is the **shear modulus**. Equations (2.1) and (2.5) constitute the displacement-stress formulation of the elastic wave equation, in the absence of dissipation.

2.1.2 Acoustic wave equation

In case of **acoustic approximation** of the propagation of the seismic waves, we assume that $\mu = 0$. Even if this assumption is strictly valid only in fluid and gaseous media, the acoustic approximation is justifiable when the seismic source is isotropic and radiates little S-wave energy, and the data analysis is restricted to the analysis of the P-waves. Examples of applications are in [6]. Therefore, for our applications, the acoustic approximation can be used to model the propagation of seismic waves in case of:

- the marine seismic acquisition, in which the source and receivers are in the water, where $\mu = 0$ and little conversion between P to S waves is present;
- the land seismic acquisition, if the source is explosive (e. g. dynamite) and we restrict the analysis to the first-arriving P waves, or specific P-wave reflections.

The assumption $\mu = 0$, simplifies significantly (2.5), giving

$$\sigma_{i,j}(\boldsymbol{x},t) = -p(\mathbf{x},\mathrm{t})\delta_{i,j}, \tag{2.6}$$

where $p(\mathbf{x},\mathrm{t}) = -\kappa(\boldsymbol{x})\nabla\mathbf{u}(\mathbf{x},\mathrm{t})$ is the scalar pressure.

Inserting (2.6) in (2.1) we obtain

$$\ddot{\mathbf{u}}(\boldsymbol{x},t) + \rho(\boldsymbol{x})^{-1}\nabla p(\boldsymbol{x},t) = \rho(\boldsymbol{x})^{-1}\boldsymbol{f}(\boldsymbol{x},t), \tag{2.7}$$

Now, if we consider the divergence of the last equation and use the definition of the pressure p, we obtain

$$-\kappa(\boldsymbol{x})^{-1}\ddot{p}(\boldsymbol{x},t) + \nabla\cdot\left(\rho(\boldsymbol{x})^{-1}\nabla p(\boldsymbol{x},t)\right) = -\nabla\cdot\left(\rho(\boldsymbol{x})^{-1}\boldsymbol{f}(\boldsymbol{x},t)\right), \tag{2.8}$$

Finally, if the density $\rho(\boldsymbol{x})$ varies much more slowly than the pressure field p and the source $\boldsymbol{f}$, we have $\nabla\cdot(\rho^{-1}\nabla p) \approx \rho^{-1}\Delta p$ and $\nabla\cdot(\rho^{-1}\boldsymbol{f}) \approx \rho^{-1}\nabla\cdot\boldsymbol{f}$), obtaining the **3D acoustic wave equation**:

$$\ddot{p}(\boldsymbol{x},t) - v_a(\boldsymbol{x})^2\Delta \mathrm{p}(\boldsymbol{x},\mathrm{t}) = v_a(\boldsymbol{x})^2 g(\mathbf{x},\mathrm{t}), \tag{2.9}$$

where $v = \sqrt{\frac{\kappa}{\rho}}$ is the **acoustic velocity** and $g(\mathbf{x},\mathrm{t}) = \nabla\cdot\boldsymbol{f}$ is the **acoustic source**. Equation (2.9) describes the seismic wave motion with a single scalar variable p, which depends only on the source term $\mathrm{g}(\mathbf{x},\mathrm{t})$ and the spatial distribution of the acoustic velocity $v(\boldsymbol{x})$. The range of the possible values of the acoustic velocity depends on the region of the Earth we consider. An example of velocity ranges for some geological materials is listed in Table 2.1.

Material	***v (km/s)***
Water	1.45-1.5
Oil	1.2-1.25
Chalk	2.3-2.6
Limestone	3.5-6.0
Dolomite	3.5-6.5

Gneiss	4.4-5.2
Granite	4.5-6.0
Basalt	5.0-6.0

Table 2.1: Example of acoustic velocities of common materials, from [17].

Since the source-receiver geometries considered are confined to a plane ($y = 0$) and assuming a 2.5D geometry of subsurface, it is possible to approximate (2.9) with the **2.5D acoustic wave equation** [18]

$$\ddot{p}(\boldsymbol{x},t) - v(x,z)^2\Delta\mathrm{p}(\boldsymbol{x},\mathrm{t}) = v(x,z)^2\mathrm{g}(\mathbf{x},\mathrm{t}), \qquad (2.10)$$

in which v_a varies only as a function of the depth z and the length x. Finally, because of the large computational cost of 3D modeling, in my work I consider the **2D acoustic wave equation**:

$$\ddot{p}(x,z,t) - v(x,z)^2\Delta\mathrm{p}(x,z,\mathrm{t}) = v(x,z)^2\mathrm{g}(x,z,\mathrm{t}), \qquad (2.11)$$

with $(x,z) \in D \subset \mathbb{R}^2$ and $\Delta\mathrm{p}(x,z,\mathrm{t}) = \left(\frac{\partial^2 p(x,z,t)}{\partial x^2} + \frac{\partial^2 p(x,z,t)}{\partial z^2}\right)$.

2.1.3 Seismic source

Generally, in seismic exploration, the dimension of the source is much smaller than the region of Earth considered. Therefore, in (2.11) it is possible to assume a ***δ*-source** in space

$$\mathrm{g}(\mathrm{x},\mathrm{z},\mathrm{t}) = \delta(x - x_0)\delta(z - z_0)s(t), \qquad (2.12)$$

where (x_0, z_0) is the location of the source in space and $s(t)$ is the seismic wavelet, describing seismic source in time. The wavelet estimation represents an essential challenge for Full Waveform Inversion because its choice considerably influences the quality of the results. Different functions exist that try to describe the wavelet of a seismic acquisition. See, e.g. [19] for a comparison of various wavelets. However, in case of real data, the wavelet usually must be estimated from the seismograms. A comparison of various methods to estimate the wavelet is, e.g., in [20].

2.2 Implementation of 2D acoustic wave equation

In this section, I describe the numerical approach used for the solution of the 2D acoustic wave equation ((2.11*)* to model the synthetic seismograms in the context of seismic exploration. This method is based on a finite difference (FD) approximation of the time and spatial derivatives [21] [22] [23].

2.2.1 Discretization in time

If we set $\dot{p} = q$, the 2D acoustic wave equation (2.11) can be written as a couple of two first order differential equations:

$$\begin{cases} \dot{q}(\boldsymbol{x},t) = v_a(\boldsymbol{x})^2\Delta\mathrm{p}(\boldsymbol{x},\mathrm{t}) + v(\boldsymbol{x})^2\delta(\boldsymbol{x} - \boldsymbol{x}_0)s(t) \\ \dot{p}(\boldsymbol{x},t) = q(\boldsymbol{x},t) \end{cases}, \qquad (2.13)$$

To approximate the time derivatives, I use the **Leap-Frog method**, an explicit second order in time approximation with uniform time sampling dt, obtaining the following system

$$\begin{cases} q\big(\boldsymbol{x}, t_{k+1/2}\big) = q\big(\boldsymbol{x}, t_{k-1/2}\big) + dt\, v(\boldsymbol{x})^2\big(\Delta \mathrm{p}(\boldsymbol{x}, t_k) + \delta(\boldsymbol{x} - \boldsymbol{x}_0)s(t_k)\big) \\ p(\boldsymbol{x}, t_{k+1}) = p(\boldsymbol{x}, t_k) + dt\, q\big(\boldsymbol{x}, t_{k+1/2}\big) \end{cases}, \tag{2.14}$$

where the number of time samples is $n_t = \frac{T}{dt} + 1$, with $k = 1, \dots, n_t$.

2.2.2 Discretization in space

For the discretization of the space, D is assumed to be a rectangular domain, where the top side ($z = 0$) represents the surface level, the bottom ($z = Z$) represents the maximum depth of the true physical domain of our interest, and the lateral sides, $x = 0$ and $x = X$ limit the domain along the x-direction. Therefore, we sample D with a uniform space sampling dx along the horizontal and vertical direction, obtaining a regular grid $D_{i,j}$ of $n_x \cdot n_z$ grid nodes, with $i = 1, \dots, n_x$ and $j = 1, \dots, n_z$, obtaining the following finite difference system

$$\begin{cases} q\big(x_{i,j}, t_{k+1/2}\big) = q\big(x_{i,j}, t_{k-1/2}\big) + dt\, v\big(x_{i,j}\big)^2 \Big(\hat{\Delta} \mathrm{p}\big(x_{i,j}, t_k\big) + \delta\big(x_{i,j} - \boldsymbol{x}_0\big)s(t_k)\Big) \\ p\big(x_{i,j}, t_{k+1}\big) = p\big(x_{i,j}, t_k\big) + dt\, q\big(x_{i,j}, t_{k+1/2}\big) \end{cases}, \tag{2.15}$$

where $\hat{\Delta}p\big(x_{i,j}\big)$ represents any finite difference operator used to approximate the Laplacian operator. An efficient way to implement the FD system consists of reordering the space grid nodes $x_{i,j}$ as a row vector x_i with $i = 1, \dots, n_x * n_z$, scrolling the grid matrix along the rows. Thus, I obtain:

$$\begin{cases} q_i^{k+1/2} = q_i^{k-1/2} + dt\, {v_i}^2\big(\hat{\Delta} p_i^k + \delta(x_i - \boldsymbol{x}_0)s^k\big) \\ p_i^{k+1} = p_i^k + dt\, q_i^{k+1/2} \end{cases}. \tag{2.16}$$

with $p_i^k = p(x_i, t_k)$, $q_i^{k+1/2} = q(x_i, t_{k+\frac{1}{2}})$, $v_i = v\big(x_{i,j}\big)$ and $s^k = s(t_k)$. Using this notation, the operator $\hat{\Delta} p_i^k$ corresponds to the application of a matrix K_Δ, called **stiffness matrix**, on the vector $\boldsymbol{p}^k$ where the elements of each row depend on the way the value of the spatial derivatives for each grid node x_i is computed. In general, the Laplacian operator is approximated by the following finite difference formula:

$$\hat{\Delta}p\big(x_{i,j}\big) = \frac{\left(\sum_{l=1}^{\frac{p_{ord}}{2}} c_l\big(p(x_{i+l,j}) + p(x_{i-l,j}) + p(x_{i,j+l}) + p(x_{i,j-l}) \big) - 4\left(\sum_{l=1}^{\frac{p_{ord}}{2}} c_l\right) p(x_{i,j})\right)}{dx^2}, \tag{2.17}$$

where the c_l are the coefficients for the p_{ord}-order approximation of the spatial derivatives [24], and p_{ord} is an even number. Table 2.2 lists the values of c_l, obtained as a function of the order of approximation $p_{ord} = 2,4, \dots ,12$.

p_{ord}	c_1	c_2	c_3	c_4	c_5	c_6
2	1	0	0	0	0	0
4	4/3	-1/12	0	0	0	0
6	3/2	-3/20	1/60	0	0	0
8	8/5	-1/5	8/135	-1/560	0	0
10	5/3	-5/21	5/126	-5/1008	1/3150	0

12	12/7	-15/56	10/189	-1/112	2/1925	-1/16632

Table 2.2: Values of the coefficients c_l for different orders of approximation of the spatial derivatives.

The row elements of K_Δ corresponds to the coefficients obtained for a p_{ord}- order approximation of the spatial derivatives. The matrix K_Δ is sparse, symmetric, banded, with the number of non-zeros elements N_Δ that is of the order of $(2p_{ord} + 1)n_x n_z$.

2.2.3 Local adaptive scheme for the stiffness matrix

In general, for an explicit finite difference scheme, two numerical conditions need to be satisfied. The first one is related to the **numerical stability relation** [25] and imposes a limit on the time sampling dt:

$$dt < \frac{dx}{v_{max}} \lambda, \tag{2.18}$$

where $\lambda \approx 0.5$ is the Courant number, and v_{max} is the maximum velocity. The second one is due to the **numerical dispersion relation** [26] and enforces a limit on dx and the order p_{ord} :

$$n < \frac{v_{min}}{dx \cdot f_{max}}, \tag{2.19}$$

where $n = n(\,p_{ord})$ is the number of points per wavelength, v_{min} is the minimum velocity of the model, and f_{max} is the maximum frequency of the wavelet $\mathrm{s(t)}$. Table 2.3 lists the values of n, as a function of the order of approximation $\mathrm{p_{ord}}$, required in (2.19) [27].

p_{ord}	2	4	6	8	10	12
n	18	6.3	4.5	3.75	3.5	3.25

Table 2.3: Values of n as a function of p_{ord}.

The numerical dispersion relation can cause very inefficient forward modelling for FWI applications because only the minimum velocity v_{min} of the model is considered to fix the grid parameters dx and p_{ord}. An example of this is in Figure 2.1a, where there is a constant velocity model of 7 km/s superimposed by a thin layer of 1.5 km/s. For this case, the choice of the grid parameters is influenced only by a small part of the model having the lowest velocity value. Even if the examined model is clearly unrealistic, we can have similar consequences also for realistic models. For example, in a marine seismic acquisition (Figure 2.1b), the minimum velocity is usually fixed to the water velocity situated on the top of the studied region, but the geological model situated below has, in general, a higher velocity.

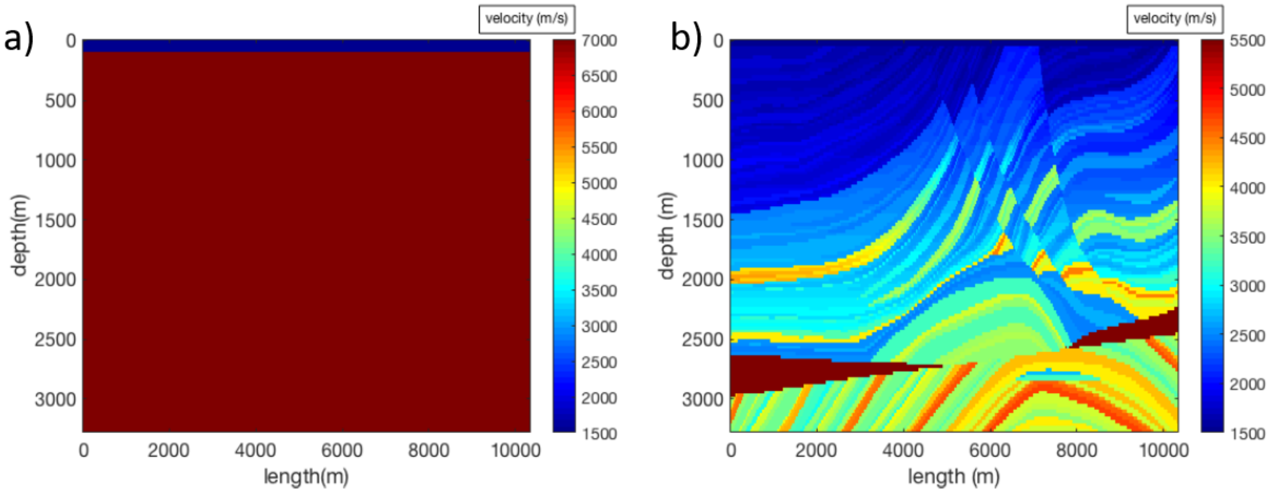


Figure 2.1: a) An example of a velocity model where the lowest velocity layer compromises the computational efficiency. b) A synthetic but realistic geological model situated below a water layer in which the mean velocity is higher than the water one.

Therefore, the choice of a classical finite difference approach to model the seismic wave propagation seems inefficient. Thus, I have implemented a new approach based on the use of different orders of approximation of the spatial derivatives in different parts of the model, instead of a global one p_{ord}. Starting from the numerical dispersion relation (2.18), and setting dx and f_{max}, I replace the minimum velocity v_{min}, with $v_i = v(x_i)$:

$$n_{loc}^{i} < \frac{v_i}{dx \cdot f_{max}}. \tag{2.20}$$

This **local numerical dispersion relation** applied to any velocity model, allows us to compute the parameter n_{loc}^{i}, i.e., the local number of points for the wavelength that must be necessary to satisfy Equation (2.20). Afterwards, I estimate the local order of the approximation of the spatial derivatives p_{loc}^{i}, from n_{loc}^{i} according to the results in Table 2.3. Therefore p_{loc}^{i} will be high in those parts of the model with low velocities, and low in the ones with high-velocity values. Figure 2.2 shows an example of the computation of p_{loc}^{i}, with $dx = 27m$, $f_{max} = 20\,Hz$ and the velocity model of Figure 2.1b. Starting from the analysis of the velocity value v_i, it is possible to compute the local number of points for wavelength n_{loc}^{i} (Figure 2.2a), and the local order of the approximation of the spatial derivatives p_{loc}^{i} (Figure 2.2b).

This new approach modifies the structure of the stiffness matrix K_Δ to a new local stiffness matrix $K_{\hat{\Delta}}$, that depends on the velocity model of each grid nodes $v_i = v(x_i)$, yielding an asymmetric matrix. Moreover, its number of non-zero elements $N_{\hat{\Delta}}$, depends on the model velocity and is in general smaller than N_Δ. In particular, I obtain:

$$N_{\hat{\Delta}} \approx \sum_{k=1}^{\frac{p_{max}}{2}} (2 \cdot 2k + 1) n_{2k}, \tag{2.21}$$

where p_{max} is the maximum implemented order, n_{2i} is the number of grid nodes with $p_{loc} = 2k$, and $n_2 + n_4 + \cdots + n_{p_{max}} = n_x \cdot n_z$.

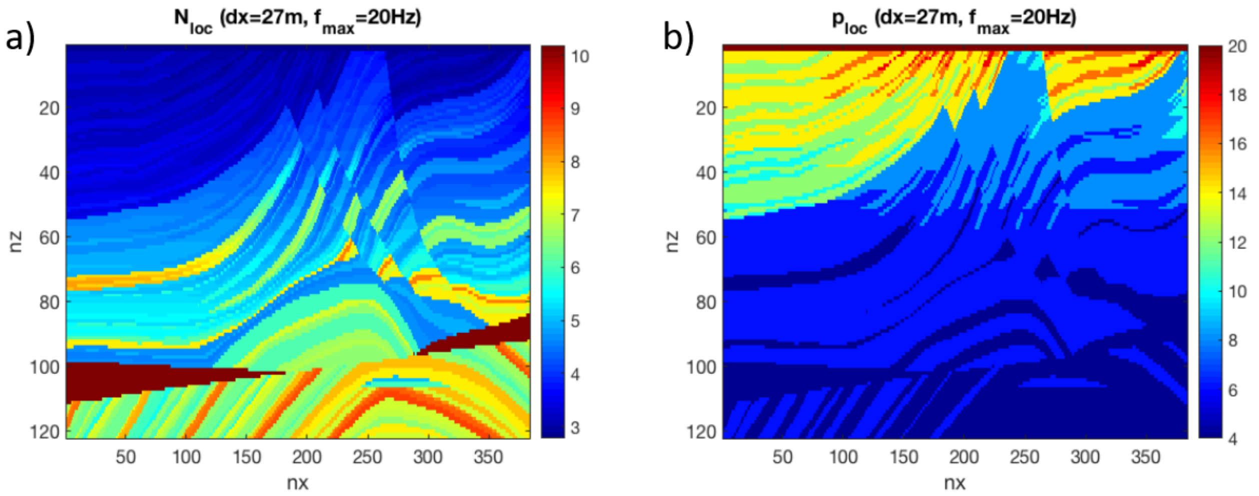


Figure 2.2:a) n_{loc} and b) p_{loc} computation for the model of Figure 2.1b, with dx=27m and f_{max} =20 Hz.

2.2.4 Point source

Since the source is a δ-source in space $\delta(\boldsymbol{x}-\boldsymbol{x}_0)$, I distinguish two cases:

- if the position of the source $\boldsymbol{x}_0$ coincides with the position of a grid point x_i (Figure 2.3b), I put
$$\delta(\boldsymbol{x}-\boldsymbol{x}_0)=\begin{cases}1, & if\ x_i=\boldsymbol{x}_0\\ 0, & if\ x_i\neq\boldsymbol{x}_0\end{cases}, \tag{2.22}$$
at each time step t_k, during the computation of the numerical solution of (2.16).
- if the position of the source $\boldsymbol{x}_0$ does not coincide with the position of any grid point x_i (Figure 2.3a), I approximate the space component of the source using the formula in [28]:
$$\delta(\boldsymbol{x}-\boldsymbol{x}_0)\approx W(\boldsymbol{x}-\boldsymbol{x}_0)\cdot\frac{1}{dx}sinc\left[\frac{\pi(x-x_0)}{dx}\right], \tag{2.23}$$
and $W(\boldsymbol{x}-\boldsymbol{x}_0)$ is the **Kaiser windowing function** [29]:
$$W(\boldsymbol{x}-\boldsymbol{x}_0)=\begin{cases}\dfrac{I_0\left(\beta\sqrt{1-\dfrac{|x-x_0|}{k_w dx}}\right)I_0\left(\beta\sqrt{1-\dfrac{|z-z_0|}{k_w dx}}\right)}{I_0(\beta)^2}, & |\boldsymbol{x}-\boldsymbol{x}_0|_\infty<k_w dx\\ 0, & otherwise\end{cases}, \tag{2.24}$$
where I_0 denotes the **zero-order Bessel function of the first kind** [30]. The parameter $k_w\in\mathbb{N}$ controls the width of the Kaiser window, whereas the parameter $\beta\in\mathbb{R}^+$ controls its shape. Typical values of k_w are between 1 and 10. The value of β is fixed as a function of k_w, using Table 2.4 [28]. Thus, the location of the point source is approximated with its band-limited version distributed through a small square of $4k_w^2$ grid nodes surrounding the exact source position.

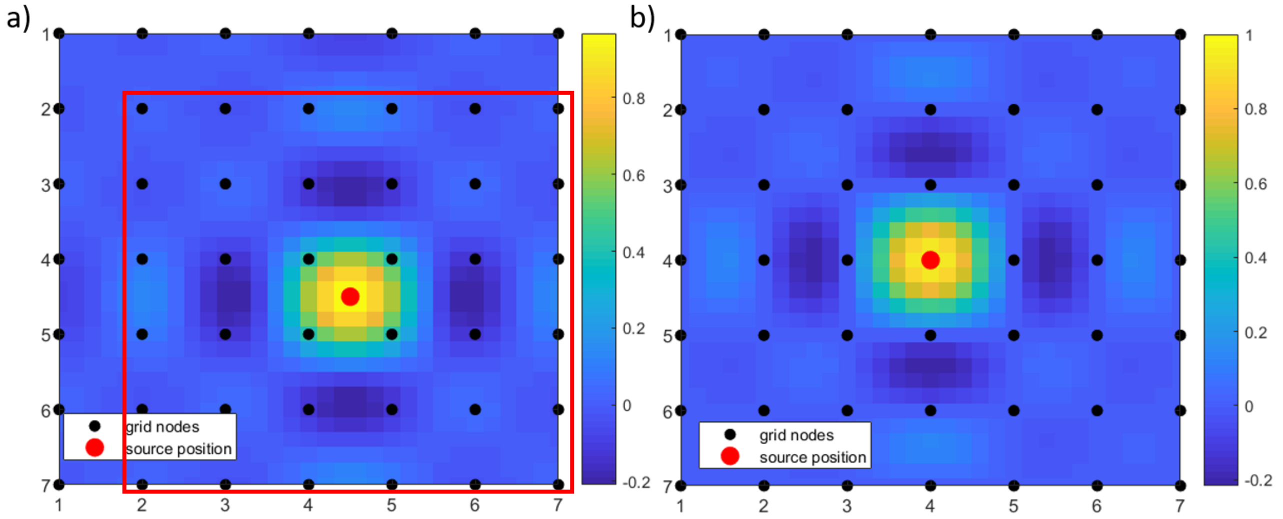


Figure 2.3: Examples of interpolation of a point source: a) the source position does not coincide with the position of any grid point and a Keiser window (the red rectangle), with k_w=3 is used; b) the source position coincides with the position of a grid point.

k_w	1	2	3	4	5	6	7	8	9	10
β	1.24	2.94	4.53	6.31	7.91	9.42	10.95	12.53	14.09	14.18

Table 2.4: Values of β as a function of k_w, taken from [28].

2.2.5 Boundary conditions

The reduction of the computational domain to only a part of the true physical domain entails artificial reflecting boundaries. To suppress these undesired reflections, an **absorbing boundary condition** is used based on the introduction of a thin narrow strip along the artificial boundary. In particular, I used the **Gaussian taper method** [31], a very popular and robust technique, that consists of a multiplication of the solution inside the absorbing layers at each time step times a **Gaussian taper factor**. This factor is 1 along the boundaries between the computational domain and the narrow strip, and decreases until the value 0.85 along the outside boundaries of the narrow strip. Thus, the previous scheme of (2.16) becomes:

$$\begin{cases} q_i^{k+1/2} = G_i\left(q_i^{k-1/2} + dt\; v_i{}^2\left(\hat{\Delta} p_i^k + \delta(x_i - \boldsymbol{x}_0)s^k\right)\right) \\ \qquad\qquad p_i^{k+1} = G_i\left(p_i^k + dt\; q_i^{k+1/2}\right) \end{cases}, \tag{2.25}$$

where $G_{i,j} \in [0.85,1]$ is the Gaussian taper factor. Clearly, each side can have a different number of absorbing grid nodes and the total number $\bar{n}_{abs}$ is given by:

$$\bar{n}_{abs} = (\bar{n}_t + \bar{n}_b)\, n_x + (\bar{n}_l + \bar{n}_r)\, n_z + (\bar{n}_t + \bar{n}_b)\, (\bar{n}_l + \bar{n}_r), \tag{2.26}$$

where $\bar{n}_t$, $\bar{n}_b$, $\bar{n}_l$, $\bar{n}_r$ are the grid node numbers of the top, bottom, left and right absorbing layers, respectively. Figure 2.4a shows an example of the computational domain with $n_x = n_z = 500$, surrounded on the four sides by absorbing boundaries layers with $\bar{n}_r$ =$\bar{n}_b$ =$\bar{n}_t$ =$\bar{n}_l$ =50, while Figure 2.4b displays the values of the Gaussian taper factor from 1 to 0.85 inside the absorbing boundary layers.

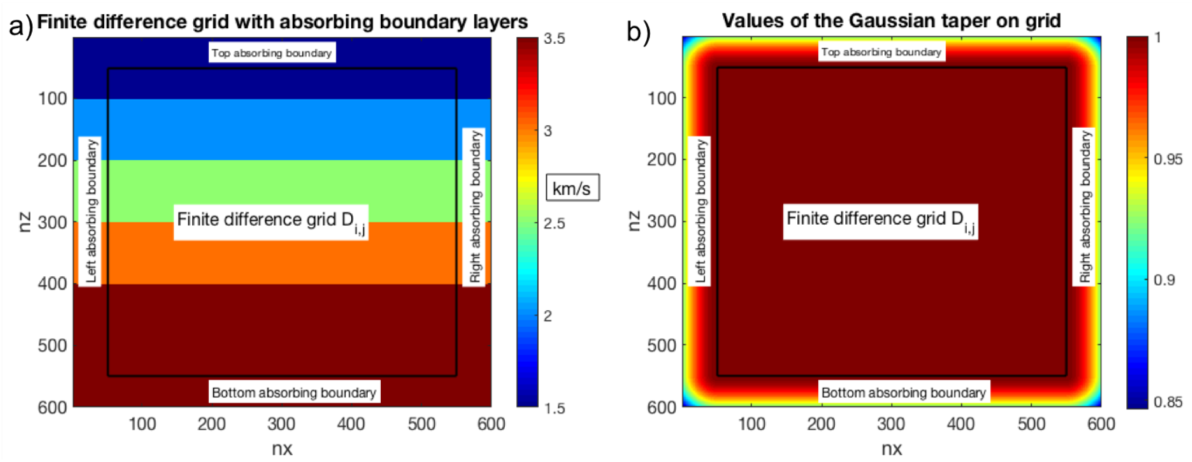


Figure 2.4: a) Example of the computational domain (n_x=n_z=500), with $\bar{n}_r$ =$\bar{n}_b$ =$\bar{n}_t$ =$\bar{n}_l$ =50 and, b) the value of the Gaussian taper. Note that $G_{i,j}$<1 inside the absorbing boundaries and $G_{i,j}$=1 otherwise.

2.2.6 The synthetic seismograms

The synthetic seismograms represent the restriction of the solution of the acoustic wave equation (*2.11*) on that part of the computational domain where the receivers are located. As in case of the seismic source, also a receiver can be approximated by a point source in space. Precisely, considering $X^r = \{\boldsymbol{x}^r\}$ the set of points related to the position of the receivers, then the predicted seismogram $u(v_p, \boldsymbol{x}, t_k)$ is the restriction of the solution of the acoustic wave equation on the X^r:

$$u(v_p, \boldsymbol{x}, t) = p(\boldsymbol{x}, t)|_{x\epsilon\, X^r}\,, \tag{2.27}$$

using v_p as the acoustic velocity model. Also in this case, we distinguish between the cases in which the position of a receiver $\boldsymbol{x}^r$ coincides with the position a grid point $\boldsymbol{x}_i$ or it does not:

- if the position of a receiver $\boldsymbol{x}^r$ coincides with the position a grid point x_i, then
$$u(v_p, \boldsymbol{x}^r, t_k) = p_i^{k+1}, \tag{2.28}$$
at each time step t_k, during the computation of the numerical solution of (2.16).
- if the position of a receiver $\boldsymbol{x}^r$ does not coincide with the position of any grid point x_i, the signal at the receiver is approximated using a formula analogous to that applied for the approximation of the source:
$$u(v_p, \boldsymbol{x}^r, t_k) = \sum_{i=1}^{n_x \cdot n_z}\left(W(\boldsymbol{x}_i - \boldsymbol{x}^r)\cdot \frac{1}{dx} sinc\left[\frac{\pi(x_i - x^r)}{dx}\right]\cdot p_i^{k+1}\right). \tag{2.29}$$

Note that the number of non-zero elements of the summation corresponds to $4k_w^2$.

2.3 Conclusions and results

In this Chapter, I have described the numerical scheme used for the solution of the 2D acoustic wave equation. This scheme models the predicted seismograms in the field of the seismic exploration, with a low computational time and a low approximation error, using also some devices for the treatment of the boundary conditions and the approximation of the sources and the receiver positions.

Chapter 3.
Implementation of the adjoint method

In this chapter, I describe the numerical scheme I implemented for the computation of the gradient of a misfit function, in the framework of 2D acoustic approximation, using the adjoint method.

3.1 An introduction to the adjoint method

The **adjoint method** is a mathematical tool proposed in control theory [32] and introduced in the theory of inverse problems [33] that considerably reduces the computational time for the gradient of a misfit function. Several authors have illustrated the adjoint method in the geophysical literature, e.g. [34], [35], [36], [7], [37], and [38] among others. It requires the computation of a time integral that depends on the solution of two differential equations: the first is the regular wave equation (elastic or acoustic) and the second is the **adjoint wave equation**, where the source term, or **adjoint source**, depends on the solution of the regular wave equation. An important issue is that to compute this time integral, both solutions must be known at the same computational time over the whole physical domain. A classical way to obviate this problem is to re-compute the solution of the regular wave equation reverse in time, during the computation of the adjoint solution which is carried out backward in time, see e.g. [7], [6], [39] and [40].

3.2 Gradient of the misfit function

Generalizing Equation (1.3), we can consider as a misfit function the L^n-norm difference between the observed and the synthetic seismogram, that is:

$$F(v) = \frac{1}{n}\sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}|u(v,\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^n \;\delta(\boldsymbol{x}-\boldsymbol{x}^r)\right)dt. \tag{3.1}$$

The gradient of (3.1) with respect to the model v gives:

$$\left[\frac{\delta F(v)}{\delta v}\right] = \sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}|u(v,\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^{n-1} sign\big(u(v,\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\big)\frac{\delta u(v_p,\boldsymbol{x},t_k)}{\delta v_p}\delta(\boldsymbol{x}-\boldsymbol{x}^r)dt\right), \tag{3.2}$$

and, inserting equation (2.27) we obtain:

$$\left[\frac{\delta F(v)}{\delta v}\right] = \sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}|p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^{n-1} sign\big(p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\big)\cdot\frac{\delta p(\boldsymbol{x},t_k)}{\delta v_p}\delta(\boldsymbol{x}-\boldsymbol{x}^r)dt\right), \tag{3.3}$$

where $p(\boldsymbol{x},t_k)$ represents the solution of the wave equation at time t_k and point $\boldsymbol{x}$, using v as velocity model. Due to the discretization of the space domain, the gradient of the misfit function is given by the $n_x \cdot n_z$ partial derivatives of F with respect to the components $v_i = v(x_i)$ of the velocity model v on the grid nodes:

$$\left[\frac{\partial F(v)}{\partial v_i}\right] = \sum_{l=1}^{n_t}\left(\sum_{r=1}^{n_r}|p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^{n-1} sign\big(p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\big)\cdot\frac{\partial p(\boldsymbol{x},t_k)}{\partial v_i}\delta(\boldsymbol{x}-\boldsymbol{x}^r)dt\right). \tag{3.4}$$

The difficulty of (3.4) lies in the appearance of the term $\frac{\partial p(\boldsymbol{x},t_k)}{\partial v_i}$, which is hard to evaluate numerically. If a finite difference approximation is used to evaluate the partial derivatives, when the number of components is large, the solution of the wave equation is computationally expensive.

3.3 Formula of the gradient using the adjoint method

The adjoint method is a mathematical tool that can efficiently compute the gradient of a misfit function in the context of seismic inverse problems. It can be used for different types of the wave equation, such as the acoustic [35] or the elastic one [7] and different types of misfit function [8]. Differentiating the acoustic wave relationship (2.9) with respect to v we obtain:

$$-2v^{-3}\ddot{p}(\boldsymbol{x},t) + v^{-2}\frac{\delta\ddot{p}(\boldsymbol{x},t)}{\delta v} - \frac{\delta\Delta p(\boldsymbol{x},t)}{\delta v} = 0. \tag{3.5}$$

Multiplying (3.15) by an arbitrary test function $p^{*T}(\boldsymbol{x},t)$ and integrating in time, we obtain:

$$-2v^{-3}\int_0^T \left(\ddot{p}(\boldsymbol{x},t)\cdot p^{*T}(\boldsymbol{x},t)\right)dt + \int_0^T \left({v_p}^{-2}\frac{\delta\ddot{p}(\boldsymbol{x},t)}{\delta v} - \frac{\delta\Delta p(\boldsymbol{x},t)}{\delta v}\right)\cdot p^{*T}(\vec{x},t)dt = 0. \tag{3.6}$$

Using the theory of the adjoint operator [6], if $p^{*T}(\boldsymbol{x},T) = \dot{p}^{*T}(\boldsymbol{x},T) = 0, \forall \boldsymbol{x} \in D$, we can rewrite *(3.6)* as follows

$$-2v^{-3}\int_0^T \left(\ddot{p}(\boldsymbol{x},t)\cdot p^{*T}(\boldsymbol{x},t)\right)dt + \int_0^T \left(v^{-2}\ddot{p}^{*T}(\boldsymbol{x},t) - \Delta p^{*T}(\boldsymbol{x},t)\right)\frac{\delta p(\boldsymbol{x},t)}{\delta v}dt = 0. \tag{3.7}$$

Then, approximating the time integral with the same time sampling of (1.3), we obtain

$$-2v^{-3}dt\sum_{k=1}^{n_t}\left(\ddot{p}(\boldsymbol{x},t_k)\cdot\ p^{*T}(\boldsymbol{x},t_k)\right) + dt\sum_{k=1}^{n_t}\left(\left(v^{-2}\ddot{p}^{*T}(\boldsymbol{x},t_k) - \Delta p^{*T}(\boldsymbol{x},t_k)\right)\frac{\delta p(\boldsymbol{x},t_k)}{\delta v}\right) = 0. \tag{3.8}$$

Finally, summing (3.4) and (3.7), we obtain:

$$\left[\frac{\delta F(v)}{\delta v}\right] = -2dt\,v^{-3}\sum_{k=1}^{n_t}\left(\ddot{p}(\boldsymbol{x},t_k)\cdot p^{*T}(\boldsymbol{x},t_k)\right) + \\ dt\sum_{k=1}^{n_t}\left(\left(v^{-2}\ddot{p}^{*T}(\boldsymbol{x},t_k) - \Delta p^{*T}(\boldsymbol{x},t_k) + \sum_{r=1}^{n_r}\left(|p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^{n-1}sign\left(p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\right)\cdot\ \delta(\boldsymbol{x}-\boldsymbol{x}^r)\right)\right)\frac{\delta p(\boldsymbol{x},t_k)}{\delta v}\right). \tag{3.9}$$

It is possible to cancel the second term, and so the red term of the sum if we can determinate a field p^{*T} is determined, that satisfies the following equation, called **adjoint wave equation**:

$$\ddot{p}^{*T}(\boldsymbol{x},t) - v^2\Delta p^{*T}(\boldsymbol{x},t) = -v^2\cdot g^{*T}(\boldsymbol{x},t), \tag{3.10}$$

subjected to the conditions $p^{*T}(\vec{x},T) = \dot{p}^{*T}(\vec{x},T) = 0,\ \forall\vec{x} \in D$, and where g^{*T} is the **adjoint source**, namely

$$g^{*T}(\boldsymbol{x},t) = -\sum_{r=1}^{n_r}\left(|p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)|^{n-1}sign\left(p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\right)\cdot\ \delta(\boldsymbol{x}-\boldsymbol{x}^r)\right). \tag{3.11}$$

Note that if $\boldsymbol{x}$ is at a receiver node, the adjoint source is the derivative of the L^{n-1}-norm difference between the observed and the synthetic seismograms

$$g^{*T}(\boldsymbol{x},t_k) = \left|p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\right|^{n-1} sign\left(p(\boldsymbol{x},t_k) - u_0(\boldsymbol{x},t_k)\right) \tag{3.12}$$

Since the adjoint equation (3.10) is of the same type of the acoustic wave equation, it can be solved using the same numerical method (2.25), but backward in time due to the to the final conditions at time T:

$$\begin{cases} {q^*}_i^{k-1/2} = G_i\left({q^*}_i^{k+1/2} + dt\, {v_i}^2 \left(\hat{\Delta} {p^*}_i^k - {v_i}^2 \cdot {g^*}_i^k\right)\right) \\ {p^*}_i^{k-1} = G_i\left({p^*}_i^k + dt\, {q^*}_i^{k-1/2}\right) \end{cases} \tag{3.13}$$

Then, due to the finite difference approximation of the space domain, we obtain

$$\left[\frac{\partial F(v)}{\partial v_i}\right] = -2dt \cdot {v_i}^{-3} \sum_{k=1}^{n_t} \left(\ddot{p}(x_i,t_k)\, p^{*T}(x_i,t_k)\right), \tag{3.14}$$

and, finally, I derive the formula for the gradient of the misfit function:

$$\frac{\partial F(v)}{\partial v_i} = \begin{cases} -\frac{2dt}{v_i} \sum_{k=1}^{n_t} \left(\hat{\Delta} p_i^k \cdot {p^*}_i^k\right),\ if\ x_i \neq \boldsymbol{x}_0 \\ -\frac{2dt}{v_i} \sum_{k=1}^{n_t} \left(\left(\hat{\Delta} p_i^k + \delta(x_i - \boldsymbol{x}_0) s^k\right) \cdot {p^*}_i^k\right),\ if\ x_i = \boldsymbol{x}_0 \end{cases}. \tag{3.15}$$

Therefore, to obtain the gradient of the misfit function, it is necessary to know Δp and p^{*T} at each grid nodes x_i and each time samples t_k. This method, compared with the finite difference approximation of the gradient, requires solving only two equations: the acoustic wave equation and the adjoint acoustic wave equation. The solution of the first equation p will be call regular solution whreas the one of the adjoint equation p^{*T} will be call adjoint solution.

3.4 Computational aspects of the adjoint method

To implement the adjoint method efficiently, some practical precautions are necessary.

First, to compute the adjoint solution p^{*T} backward in time the adjoint source g^{*T} is necessary, that depends on the regular solution p, computed forward in time. To overcome this issue, first I compute the regular solution p and I evaluate the adjoint source g^{*T} and store it at each time step t_k. Then, I compute the adjoint solution p^{*T} using the scheme in (3.13).

The second problem is that to compute each k −component of the sum in (3.15) the knowledge of p and p^{*T} simultaneously, at each time step t_k, over the whole physical domain, is necessary.

Three approaches are possible to solve this last issue. In the first approach, during the computation of p^{*T}, I recover the regular solution p, previously stored at each time step t_k in all the physical domain. To store the regular solution p, $nt * nx * nz$ times the memory used for the value of p on a grid node is necessary. Unfortunately, this method is not practical in case of an acoustic FWI with a considerable number of grid nodes and many step times, because of the excessive memory requirement.

In the second approach, in absence of any absorbing boundary layers, when p^{*T} is computed backward in time, the regular solution p is re-computed reverse in time, from t_{k+1} to t_k, starting from the final states $p_i^{n_t}$ and $q_i^{n_t-1/2}$ and using the following reverse scheme (2.16):

$$\begin{cases} p_i^k = p_i^{k+1} - dt\, q_i^{k+1/2} \\ q_i^{k-1/2} = q_i^{k+1/2} - dt\, {v_i}^2\left(\Delta p_i^k + \delta(x_i - \boldsymbol{x}_0) s^k\right) \end{cases}, \tag{3.16}$$

At this point, both p and p^{*T} are simultaneously known, and it is possible to compute the gradient using (3.15). Using this procedure, I solve the wave equation only three times: the first to compute the regular solution forward in time, the second to compute the adjoint solution backward in time, and the third to re-compute the regular solution reverse in time. However, we saw in the previous chapter that it is necessary to consider absorbing boundary layers to reduce the dimension of the computational domain. In this case, it is not possible to use the scheme in (3.16) to reconstruct the regular solution p reverse in time from its final value only. In fact, the regular solution must be multiplied, at each time step t_k, by the inverse of the Gaussian tapering factor. This term is bigger than 1 for the grid nodes inside the absorbing boundary layers, causing numerical instability. Figure 3.1a shows the inverse of the Gaussian taper factor on the computational domain of Figure 2.4a.

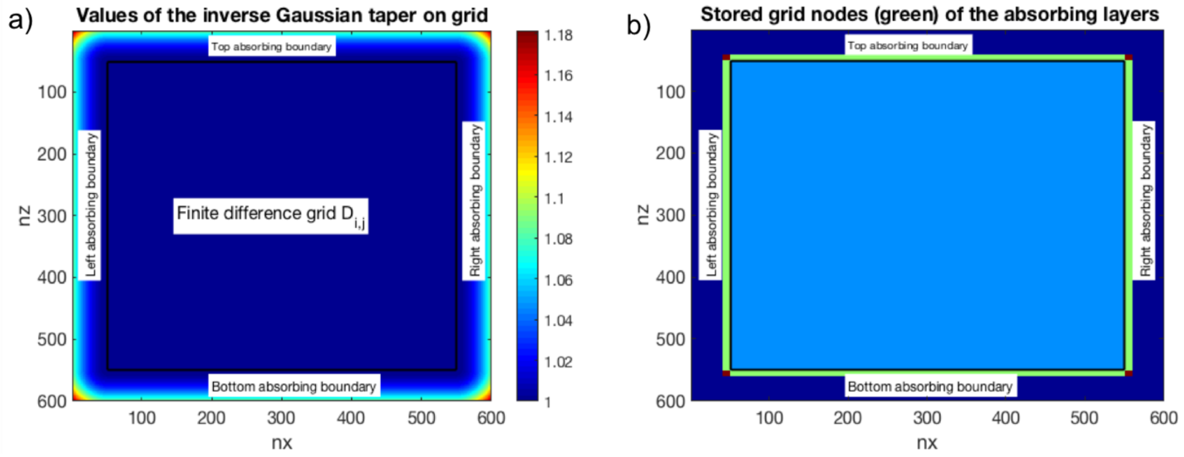


Figure 3.1: a) Values of the inverse Gaussian taper on the grid of Figure 2.4a and b) the grid nodes (coloured by green) where the solution is stored at each time step, with a global order p_{ord}=10. In this case, the absorbing layers have a width of $L = 50 \cdot dx$, with $n_{abs} = 100(n_x + n_z) + 4 \cdot 50^2$, but we must store only $20(n_x + n_z)$ nodes of the absorbing region.

The third approach combines the previous two ones, and it is the one I implemented. During the computation of (2.25), a part of the solution p inside the absorbing boundary layers is stored at each time step t_k. This part corresponds to the part of the solution inside the absorbing boundary layers necessary for the computation of the Laplacian of the grid nodes near the boundaries. If a global order of approximation of the spatial derivatives p_{ord} is used, the memory required corresponds to $n_t \cdot p_{ord} \cdot (n_x + n_z)$ times the memory used by the value of p on a grid node and this amount is much smaller than the one required by the first approach for big modelling grids. In Figure 3.1b the absorbing layers have a width of $\mathrm{L} = 50 \cdot \mathrm{dx}$, with $\mathrm{n_{abs}} = 100(\mathrm{n_x} + \mathrm{n_z}) + 4 \cdot 50^2$, but we must store only $20(\mathrm{n_x} + \mathrm{n_z})$ nodes of the absorbing region in case of $\mathrm{p_{ord}} = 10$.

Then, to avoid numerical instability, during the computation of the adjoint solution p^{*T}, the regular solution p is re-computed reverse in time using the scheme (3.16) and the previously stored solution. Note that, if a local order of approximation of the spatial derivatives is used, the memory required to store the regular solution inside the absorbing layers further decreases.

3.5 Including the processing operations in the misfit function

Usually, a certain number of processing operations are required before a comparison between synthetic and observed seismograms, among which:

- **trace normalization**, to enforce information at long offset of the seismograms;

- **trace envelope,** to reduce the cycle-skipping effect and, in general, the non-linearity of the misfit function;
- **muting**, to focus the modelling and/or the inversion on specific events of the seismograms;
- **filtering**, to have similar frequency range for the observed and the predicted seismograms;

To include the processing operations, the Equation (3.1) is modified to a more general equation:

$$F(v) = \frac{1}{n}\sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}\left|G\big(u(v,\boldsymbol{x},t_k)\big) - u_0(\boldsymbol{x},t_k)\right|^n \ \delta(\boldsymbol{x}-\boldsymbol{x}^r)\right)dt. \tag{3.17}$$

where G represents a sequence of processing operations to apply on the synthetic seismograms:

$$G\big(u(v,\boldsymbol{x},t_k)\big) = G_n\left(G_{n-1},..G_1\big(u(v,\boldsymbol{x},t_k)\big)\right). \tag{3.18}$$

For example, in case of a muting and a trace normalization operation we have:

$$G\big(u(v,\boldsymbol{x},t_k)\big) = \frac{m(\boldsymbol{x},t_k)\cdot u(v,\boldsymbol{x},t_k)}{\sqrt{\sum_{k=1}^{n_t}\big(m(\boldsymbol{x},t_k)\cdot u(v,\boldsymbol{x},t_k)\big)^2}}, \tag{3.19}$$

where $m(\boldsymbol{x},t_k)$ is the muting function. For a better readability, I set $\Psi(v)$ the quantity

$$\Psi(v) = \Psi\left(G\big(u(v,\boldsymbol{x},t_k)\big)\right) = \left|G\big(u(v,\boldsymbol{x},t_k)\big) - u_0(\boldsymbol{x},t_k)\right|^n, \tag{3.20}$$

obtaining the following expression for the misfit function

$$F(v) = \frac{1}{n}\sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}\Psi\left(G\big(u(v,\boldsymbol{x},t_k)\big)\right)\ \delta(\boldsymbol{x}-\boldsymbol{x}^r)\right)dt. \tag{3.21}$$

The gradient of (3.21) respect to the model v corresponds to:

$$\left[\frac{\delta F(v)}{\delta v}\right] = \sum_{k=1}^{n_t}\left(\sum_{r=1}^{n_r}\left(\frac{\delta\Psi\big(G(u(v,\boldsymbol{x},t_k))\big)}{\delta G}\cdot\frac{\delta G(u(v,\boldsymbol{x},t_k))}{\delta u}\cdot\frac{\delta u(v,\boldsymbol{x},t_k)}{\delta v}\right)\delta(\boldsymbol{x}-\boldsymbol{x}^r)dt\right) \tag{3.22}$$

where:

- $\frac{\delta\Psi(G(u(v,\boldsymbol{x},t_k)))}{\delta G} = \left|G\big(u(v,\boldsymbol{x},t_k)\big)-u_0(\boldsymbol{x},t_k)\right|^{n-1} sign\left(G\big(u(v,\boldsymbol{x},t_k)\big)-u_0(\boldsymbol{x},t_k)\right)$;
- $\frac{\delta G(u(v,\boldsymbol{x},t_k))}{\delta u}$ is the derivative of the processing operator with respect to the synthetic seismograms;
- $\frac{\delta u(v,\boldsymbol{x},t_k)}{\delta v}=\frac{\partial p(\boldsymbol{x},t_k)}{\partial v}$ is the derivative with respect to v of the solution of the wave equation at time t_k, and point $\boldsymbol{x}$, using v as velocity model.

Using similar mathematic passages, such as from (3.5) to (3.8), we obtain a formula for the gradient of the misfit function similar to (3.9), that is:

$$\left[\frac{\delta F(v)}{\delta v}\right] = -2dt\, v^{-3} \sum_{k=1}^{n_t} \left(\ddot{p}(\boldsymbol{x}, t_k) \cdot p^{*T}(\boldsymbol{x}, t_k)\right) + \tag{3.23}$$
$$dt \sum_{k=1}^{n_t} \left(\left(v^{-2} \ddot{p}^{*T}(\boldsymbol{x}, t_k) - \Delta p^{*T}(\boldsymbol{x}, t_k) + \sum_{r=1}^{n_r} \left(\frac{\delta \Psi\left(G(u(v, \boldsymbol{x}, t_k))\right)}{\delta G} \cdot \frac{\delta G(u(v, \boldsymbol{x}, t_k))}{\delta u}\, \delta(\boldsymbol{x} - \vec{x}^r) \right) \right) \frac{\delta p(\boldsymbol{x}, t_k)}{\delta v} \right).$$

If the second term of the sum is cancel out, that means to determinate a field p^{*T}, that satisfies the adjoint wave equation (3.10) where this time the adjoint source $g^{*T}(\boldsymbol{x}, t)$ corresponds to:

$$g^{*T}(\boldsymbol{x}, t) = -\sum_{r=1}^{n_r} \left(\frac{\delta \Psi\left(G(u(v, \boldsymbol{x}, t_k))\right)}{\delta G} \cdot \frac{\delta G(u(v, \boldsymbol{x}, t_k))}{\delta u}\, \delta(\boldsymbol{x} - \vec{x}^r) \right). \tag{3.24}$$

The computation of the red term in (3.24) requires the computation of the gradient of processing operator respect to the synthetic seismogram. In case of a processing operator such (3.18) we can use the chain rule to obtain:

$$\frac{\delta G\left(u(v,\boldsymbol{x},t_k)\right)}{\delta u} = \frac{\delta G_n\left(G_{n-1},..G_1(u)\right)}{\delta u} = \frac{\delta G_n\left(G_{n-1},..G_1(u)\right)}{\delta G_{n-1}} \cdot \ldots \cdot \frac{\delta G_1(u)}{\delta u}. \tag{3.25}$$

For example, in case of a processing sequence such as (3.19) we have:

$$\frac{m(\boldsymbol{x},t_k)}{\sqrt{\sum_{k=1}^{n_t}\left(m(\boldsymbol{x},t_k) \cdot u(v,\boldsymbol{x},t_k)\right)^2}} \cdot \left[1 - \frac{m(\boldsymbol{x},t_k) u(\boldsymbol{x},t_k) \sum_{k=1}^{n_t}\left(m(\boldsymbol{x},t_k) \cdot u(\boldsymbol{x},t_k)\right)}{\left(\sum_{k=1}^{n_t}\left(m(\boldsymbol{x},t_k) \cdot u(\boldsymbol{x},t_k)\right)^2\right)^{\frac{1}{4}}} \right]. \tag{3.26}$$

Using (3.25), many other processing operations can be included in the misfit function to compute the specific adjoint source. However, the possible changes in (3.24) do not influence the kernel of numerical implementation proposed in the previous section.

3.6 Conclusions and results

In this Chapter, I have described a detailed numerical implementation of the adjoint method to compute the gradient of a misfit function efficiently. Starting from classical theoretical developments, I illustrate the main computational aspects and the practical cautions I consider concerning stability, accuracy and memory resources. As misfit function, I consider L^n-norm difference between the synthetic and the observed seismograms, but I also include the possibility to execute a simple processing sequence on the synthetic seismograms.

Chapter 4.
The software fullwave2D

In this chapter, I describe the software developed during my Ph.D. The software is designed to simulate the propagation of seismic waves in a 2D acoustic model of subsurface and use the numerical approach described in the previous Chapter 2. The code organization, the input and output files, the software and hardware requirements, and how to compile and execute the software is described in Appendix A.

4.1 Why writing another modelling code?

Nowadays a great number of modelling codes for the solution of the wave equation (acoustic and elastic, 2D and 3D) have been developed to simulate the synthetic seismograms in the context of FWI.

However, these codes have often been optimized for specific applications and suffer from some limitations that can make difficult their use, such as:

- a rigid structure of modelling, often optimized for a specific context, with few parameters that can be set from the user;
- the difficulty to include some processing operations on the synthetic seismograms at the end of the modelling code.

In writing this code, we have paid attention to implement a software that it is efficient but flexible, which can be used for modelling and inversion problems, and that can include some processing operations on the seismograms. Among the number of features of the code, we can mention:

- sources and receivers may be placed anywhere in the model- not just at integer grid points; this option is used to treat seismograms in which the location of sources and receivers can be very irregular, such as the one of a land acquisition;
- both free surface and absorbing boundary conditions are implemented, and different length of the absorbing layers can be set. This option is used to optimize the modelling when, for example, the location of the sources is near a boundary of the grid;
- the order of approximation of the spatial derivatives can be set either by the user to a global order or can be optimized using the p_{loc}-method described in the previous Chapter 2;
- different wavelets can be chosen for the different sources. Experience reveals that this option can be useful in cases where there are different wavelets for the seismograms, such as, for example, in a land acquisition;
- if the time step chosen for the synthetic seismograms, called recording step, violates the numerical stability relation, the code uses a sub-step of the recording one to assure the numerical stability relation during the modelling. However, the synthetic seismograms are recorded using the original recording step in order to save memory. This option can be useful to test different velocity models with different velocity range in an inversion procedure;
- a distinction can be done between the recording time, that is the common length of all the synthetic seismograms, and the modelling time necessary to compute each of them. This option can save computational time, for example, in case of seismograms with different characteristics, such as different maximum offset or muting masks with maximum time length;

- the possibility to use the synthetic seismograms, obtained at the end of the modelling procedure, together with the observed ones, for the computation of the value of a misfit function that is the L^n-norm difference between the synthetic and the observed seismograms, with n an integer number (typically 1 or 2).
- the possibility to use the synthetic seismograms, obtained at the end of the modelling procedure, together with the observed ones, for the computation of the gradient a misfit function. the possibility to include some processing operations, in any desired order, on the synthetic data, such as filtering, trace envelope, muting and trace normalization. These operations can also be included in computation of the misfit function and its gradient;
- the name of files containing the receivers positions, the sources positions, the velocity model and the wavelets are input parameters that can be specified by the user. This fact can be important when a user wants to execute different runs of the code at the same time, using different velocity models or acquisition layouts.

4.2 Type of tasks

The code is designed to undertake three different possible tasks that may arise in the context of FWI:

- **Modelling task**, in which the code produces as output file the synthetic seismograms, using the information of the input file, on the basis of the implementation explained in Chapter 3;
- **Misfit computation**, in which the code uses the synthetic seismograms for the computation of a misfit function;
- **Gradient computation**, in which the code uses the synthetic seismograms for the computation of the gradient of a misfit function, by the implementation explained in Chapter 4.

The tasks must be specified during the setting of the code by the user.

4.3 Some Examples

In this section, I show how to use our code in the context of seismic exploration, to simulate a possible marine seismic acquisition. As input files, I used the ones located in the subdirectory *work_example* (See Appendix A). The acquisition layout consists of two sources, located at the begin and the end of the seismic profile, and a recording spread composed by 384 receivers, with a group interval of 24m. Both sources and receivers are at a depth of 24 m (see Figure 4.1). The source wavelet is a Ricker wavelet with a peak frequency of 6 Hz and maximum frequency of 18 Hz (see Figure 4.2). The velocity models I consider is the Marmousi model (see Figure 4.3) and a smooth version of it (see Figure 4.4).

In the numerical implementation, I used $dt = 0.002\ s$ (sampling time), $T = 6s$ (recording time), $dx = 24m$ (sampling space) and $p_{ord} = optimized$ (optimized order for the approximation of spatial derivatives, using the p_{loc} −method). The grid is made of 46848 nodes, with $n_x = 384$ and $n_z = 122$; the water layer is modelled by the first two rows of the grid. I put absorbing boundary conditions on the lateral and bottom sides of the model, with $n_r = n_l = n_b = 30$ and reflecting boundary conditions at the top side to simulate the sea-air interface. The modelling parameters are specified in the *runfile* file.

Figure 4.5 and Figure 4.6 display the two raw synthetic seismograms related to the Marmousi model, without any processing operations on them. Figure 4.7 and Figure 4.8 display the same synthetic seismograms but with the application of a processing sequence consisting of a lowpass filter up to 10Hz and trace normalization. Figure 4.9 shows the gradient of a misfit function, obtained using:

- the seismograms of Figure 4.5 and Figure 4.6 as observed data;
- the model of Figure 4.4 as velocity model;
- The L^2-norm difference between the predicted and observed seismograms as misfit function.

Figure 4.10 shows the gradient of a misfit function, obtained using:

- the seismograms of Figure 4.5 and Figure 4.6 as observed data;
- the model of Figure 4.4 as velocity model;
- the L^2-norm difference between the predicted and observed seismograms as misfit function;
- a processing sequence on the synthetic seismograms consisting of a lowpass filter up to 10Hz and trace normalization.

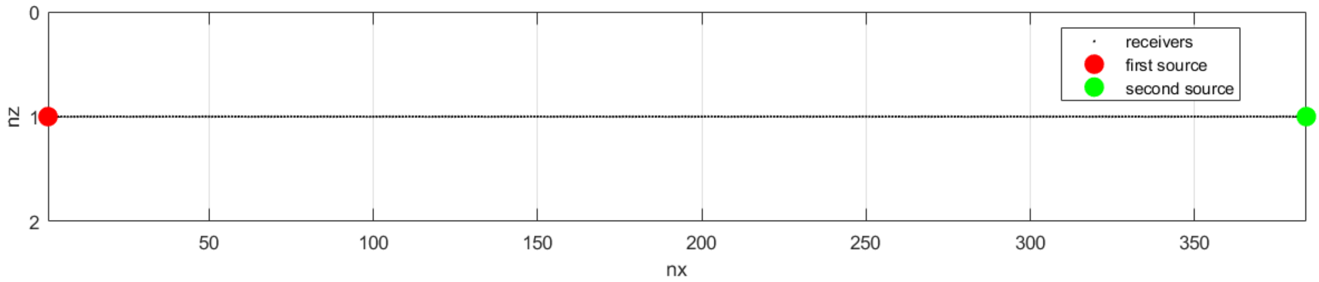


Figure 4.1: Position of the sources and the receivers on the modelling grid.

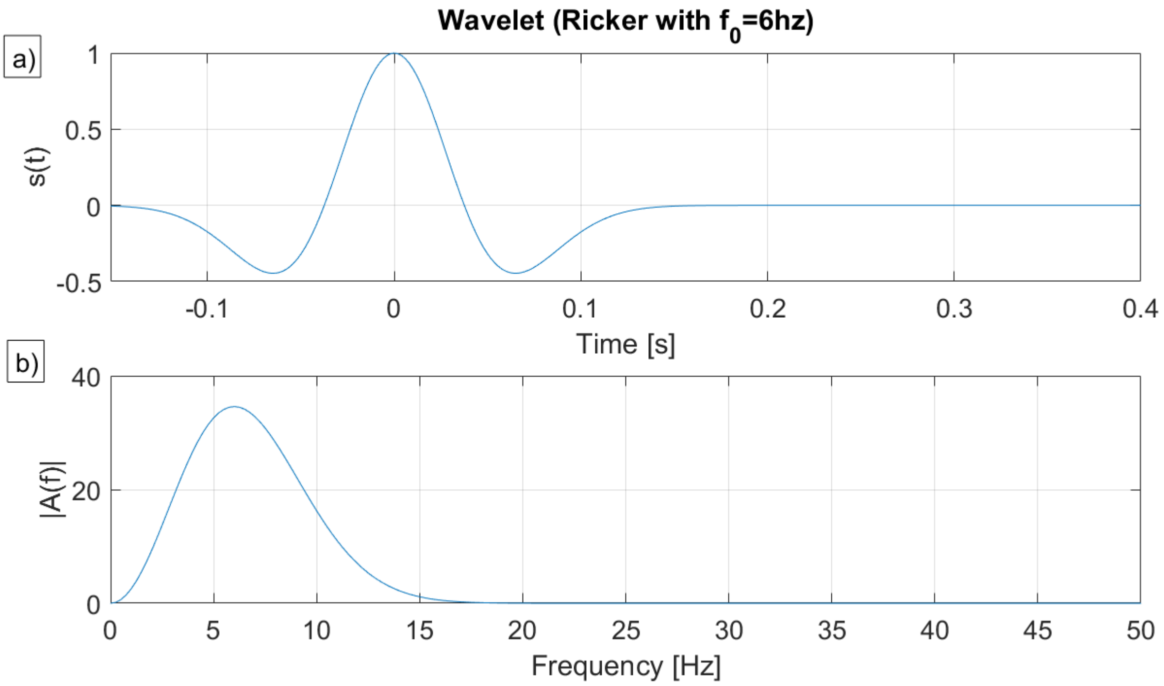


Figure 4.2: a) Ricker wavelet with a peak frequency of 6 Hz and b) its amplitude spectrum.

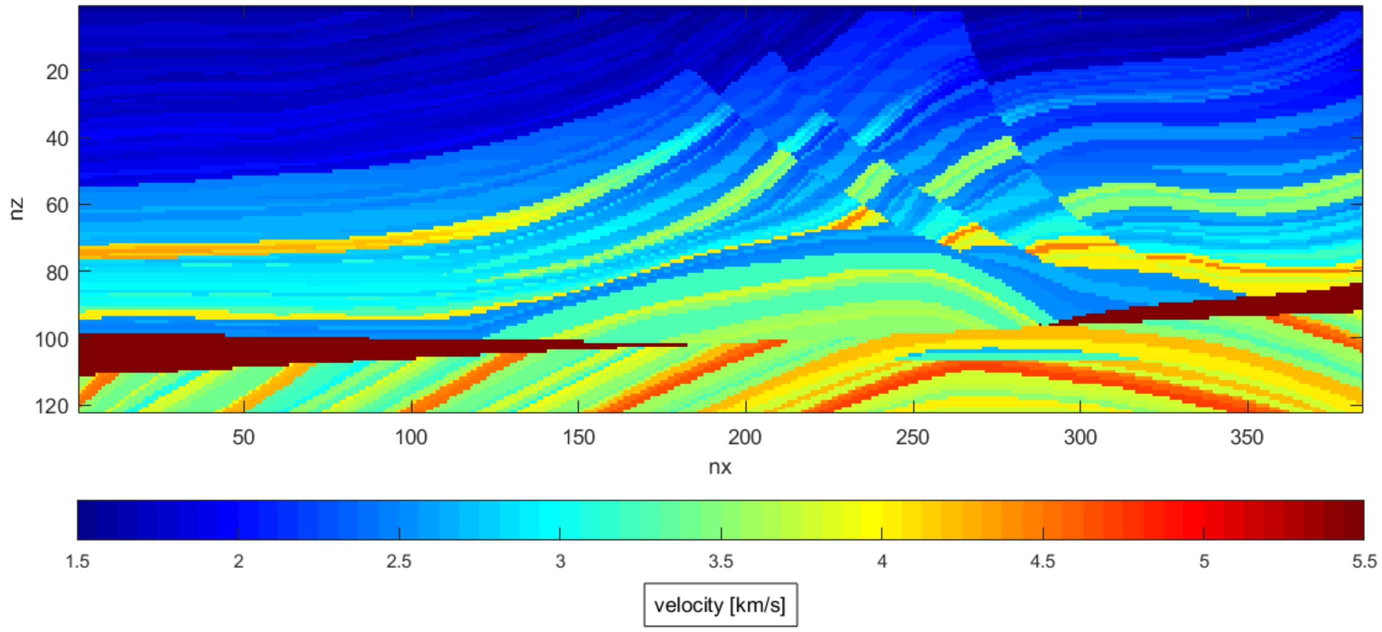


Figure 4.3: Marmousi model on the modelling grid.

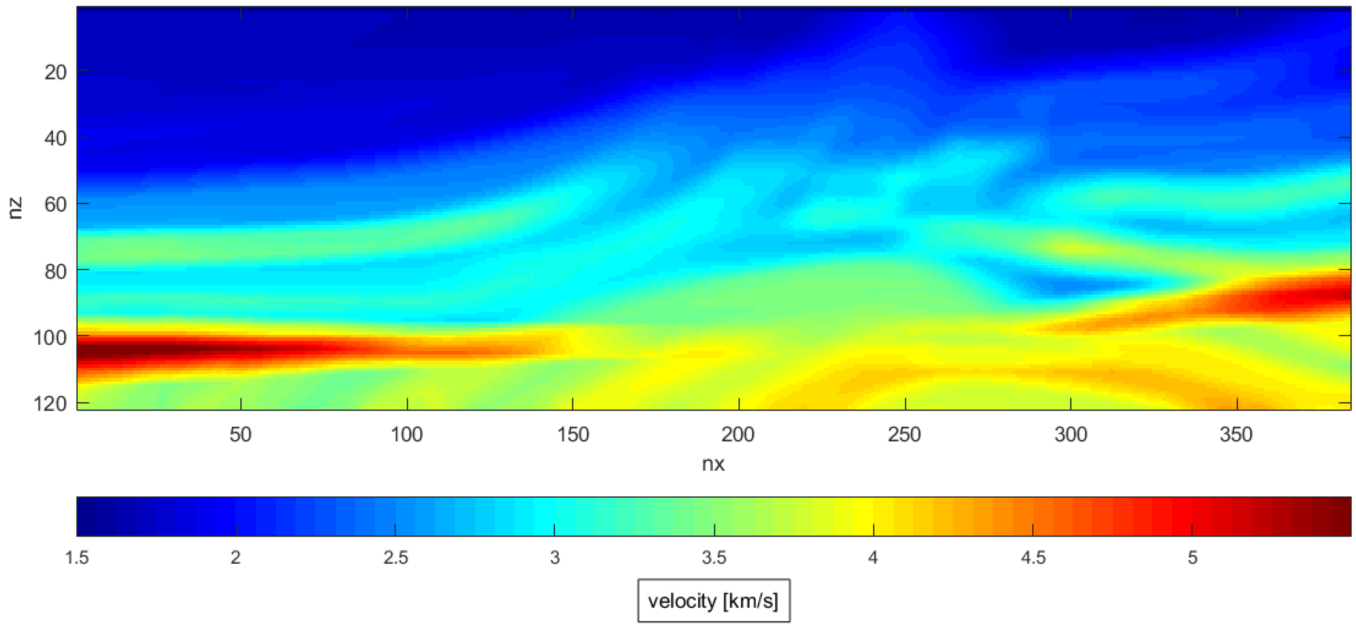


Figure 4.4: Smooth Marmousi model on the modelling grid.

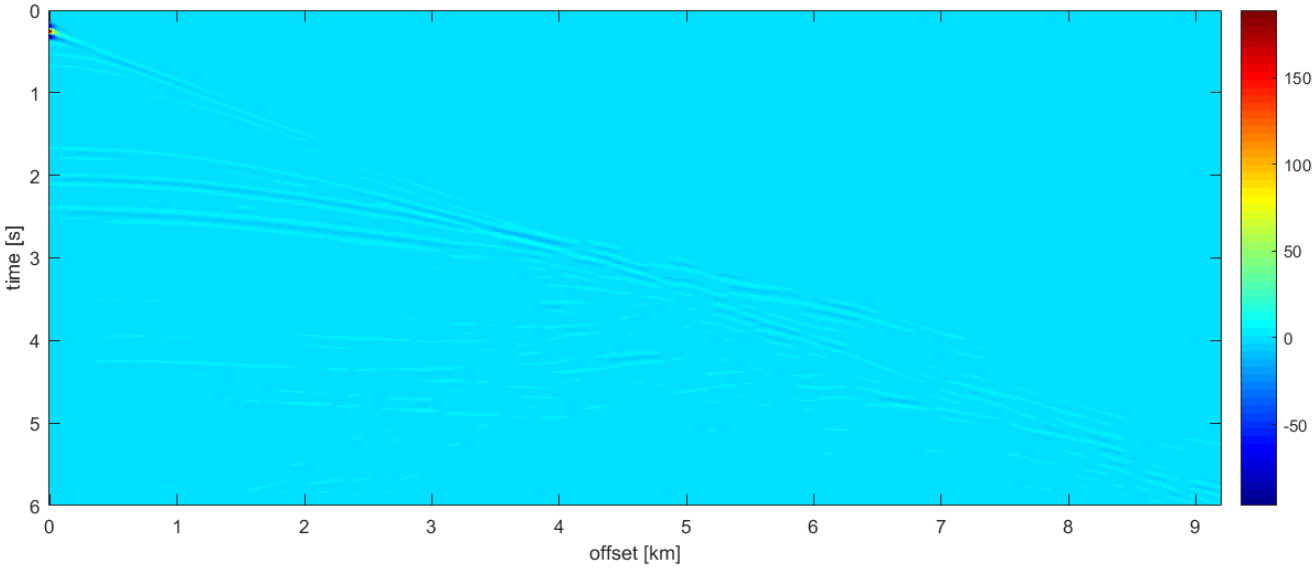


Figure 4.5: Raw synthetic seismogram related to the first source, using the Marmousi model.

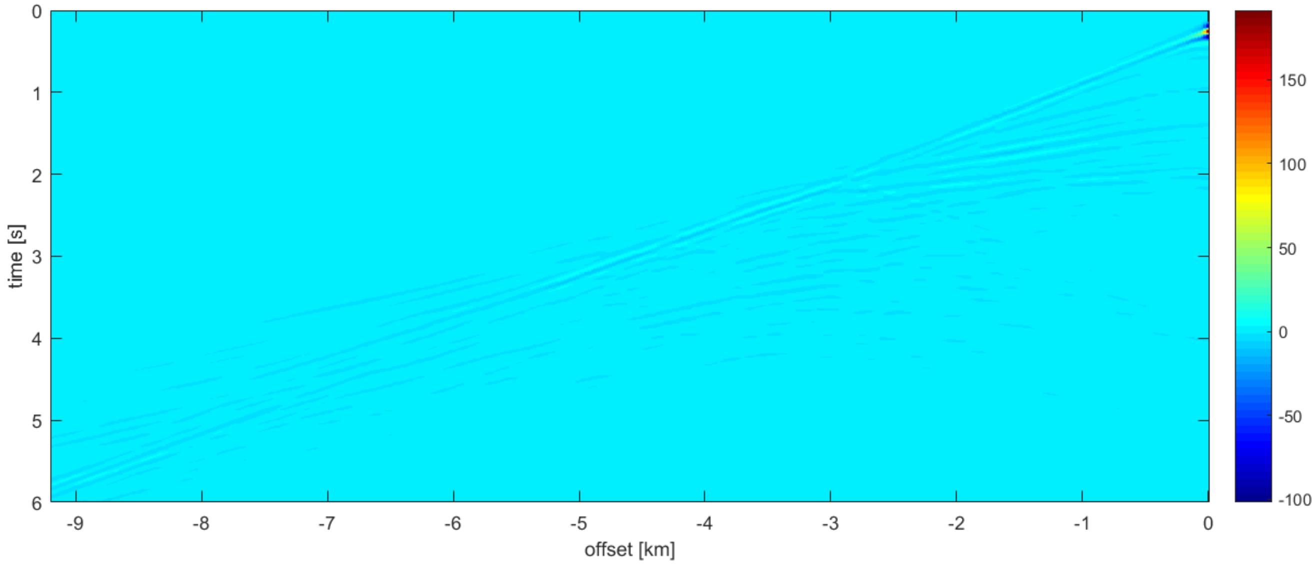


Figure 4.6: Raw synthetic seismogram related to the second source, using the Marmousi model.

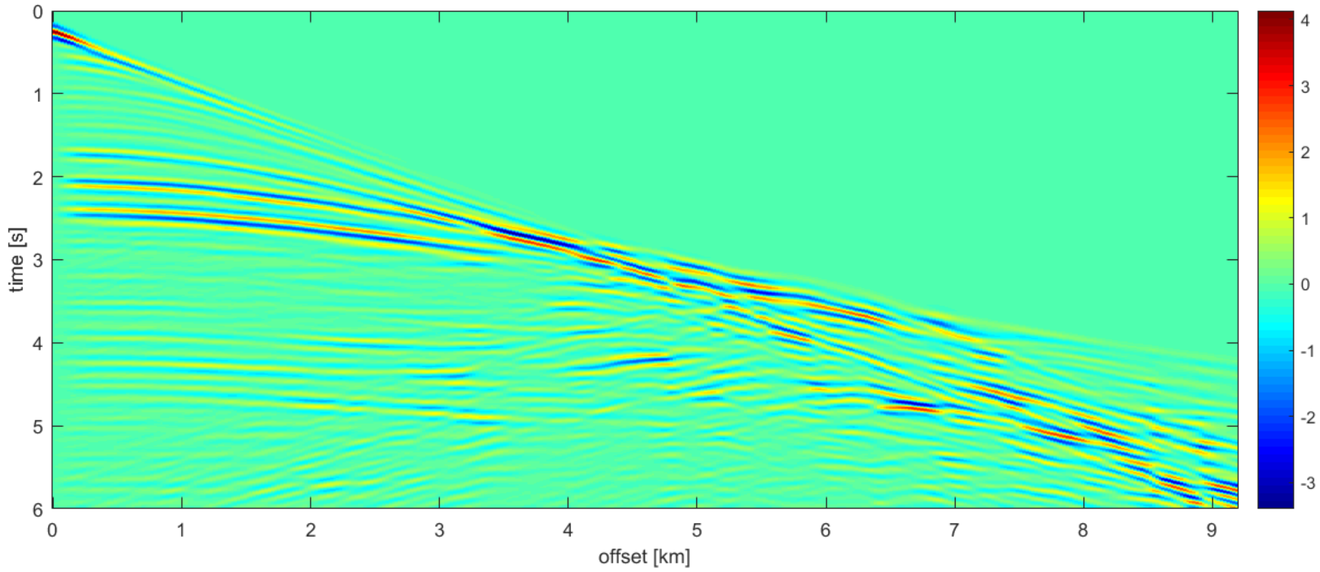


Figure 4.7: Synthetic seismogram related to the first source, using the Marmousi model, with the application of a processing sequence consisting of a lowpass filter up to 10Hz and trace normalization.

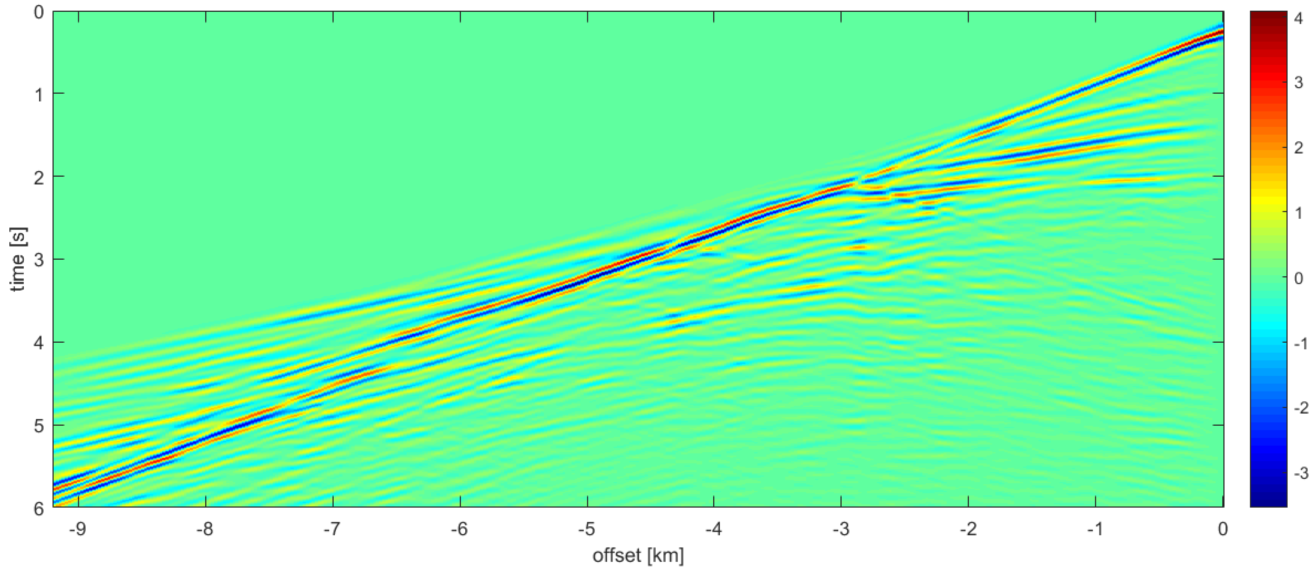


Figure 4.8: Synthetic seismogram related to the second source, using the Marmous model, with the application of a processing sequence consisting of a lowpass filter up to 10Hz and trace normalization.

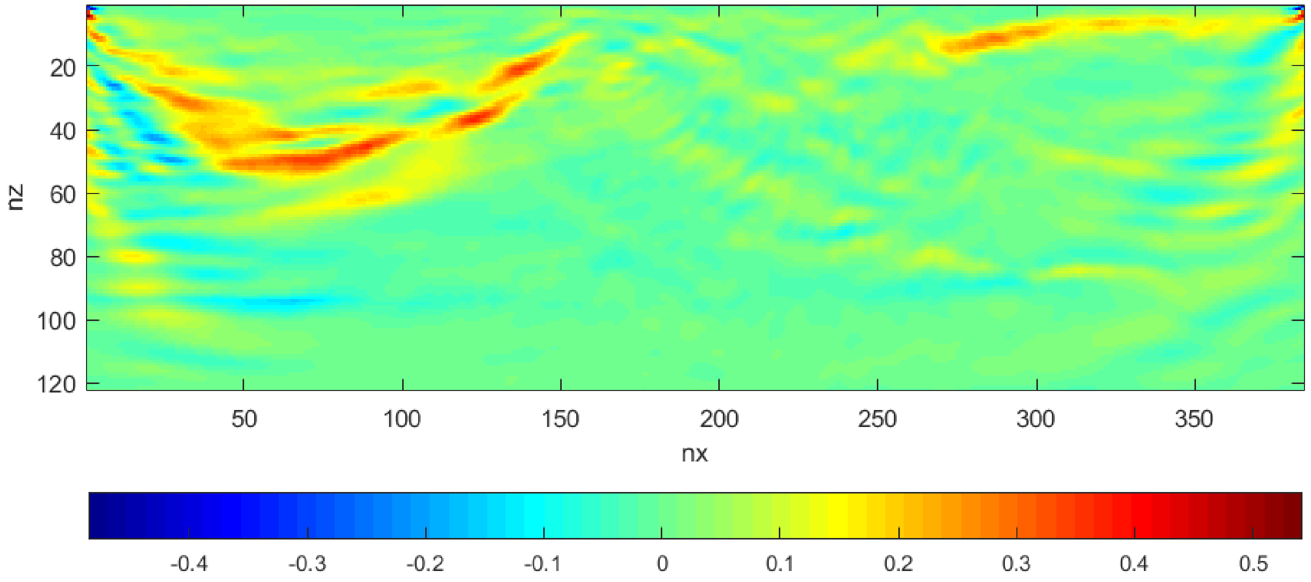


Figure 4.9: Gradient of the L^2- norm difference between the observed seismograms and predicted seismograms. As observed ones, I consider the seismograms of Figure 4.5 and Figure 4.6. As predicted ones, I consider the synthetic seismograms obtained from the smooth Marmousi model of Figure 4.4.

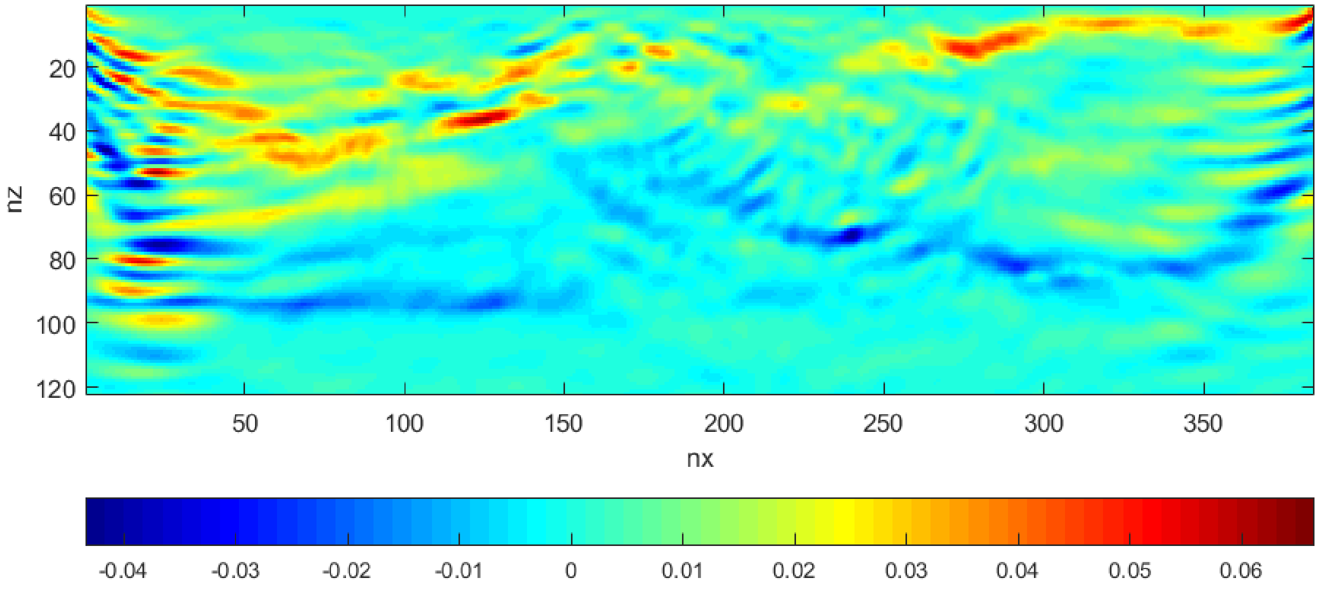


Figure 4.10: Gradient of the L^2- norm difference between the observed seismograms and predicted seismograms. As observed ones, I consider the seismograms of Figure 4.7 and Figure 4.8. As predicted ones, I consider the synthetic seismograms obtained from the smooth Marmousi model of Figure 4.4, On both seismograms a processing sequence consisting of a lowpass filter up to 10Hz and trace normalization is applied.

4.4 Conclusions and results

In this Chapter I presented the software developed during my PhD to undertake three different possible tasks in the context of FWI: the simulation of the synthetic seismograms, the evaluation of a misfit function, and the gradient computation of a misfit function. Besides it is also possible to include simple processing operations on the seismograms for a better comparison between synthetic and observed seismograms. As example, I have used my software to simulate the seismograms and the gradient of a misfit function in a marine seismic acquisition.

Chapter 5.
Heuristic algorithms for global optimization

In this chapter, I introduce the study I have done on the global optimization, focusing the attention on two global optimization algorithms to use in the context of Full Waveform Inversion.

5.1 Introduction to Global optimization

Global optimization is a branch of applied mathematics and numerical analysis with the goal of finding the best possible parameter x from a set X, according to some criterion f.
In case of a continuous optimization problem, X corresponds to a subset of R^n and the criterion can be expressed as a mathematical function, called **objective function**,

$$f: X \subseteq R^n \rightarrow R. \tag{5.1}$$

In this case, the common form to represents a global optimization problem is as a minimization problem: find $\bar{x} \in X$, such that:

$$f(\bar{x}) = \min_{x \in X} f(x). \tag{5.2}$$

Note that finding a global maximum of a function $g(x)$ is analogous to find the global minimum of $f(x) = -g(x)$. We distinguish between:

- an **analytic objective function**, in which an analytic expression between a parameter x and the value of $f(x)$ is available;
- a **black-box objective function**, where the value of the function $f(x)$ can only be obtained by giving the parameter value to a subroutine or a program that generates the value for f after some time.

FWI and, in general, optimization problems that involve the numerical solution of PDE equations, are typical examples of black-box objective functions.

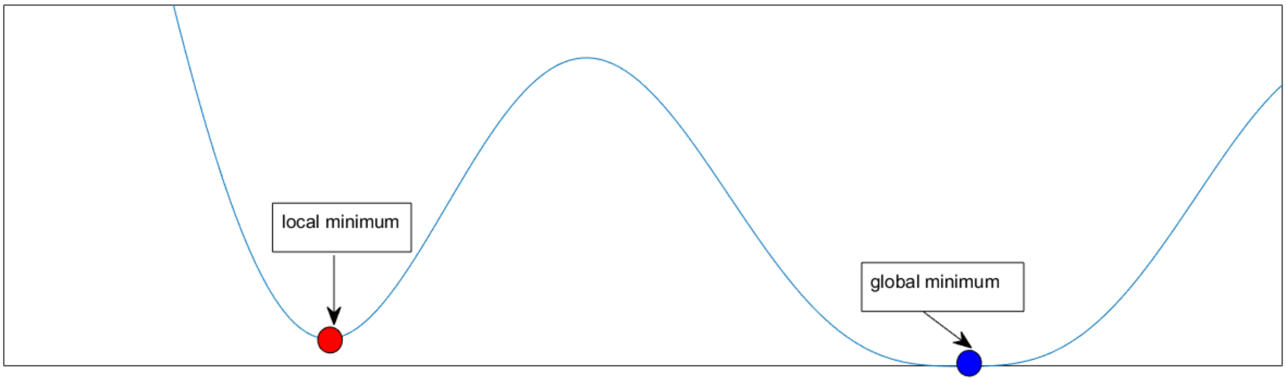


Figure 5.1: An example of objective function with a global minimum (blue) and a local minimum (red)

A global optimization problem is distinguished from a classical optimization one by its focus on finding the minimum over all the X, as opposed to finding local minima (see Figure 5.1). In particular, we distinguish between:

- a **local minimum**, that is a point $\tilde{x} \in X$, where $f(\tilde{x}) \leq f(x)$ at a subset $\tilde{X} \subseteq X$;

- a **global minimum**, that is a point $\bar{x}$ where $f(\bar{x}) \leq f(x)$ at all X.

Finding a local minimum can be relatively straightforward by using classical **local optimization methods**. These are search strategies that use the knowledge of local information, such as the gradient or the hessian of the objective function, creating an iterative sequence of points $\{x_i\}$, that converges towards a local minimum, starting from an initial model x_0. A complete review of the main local optimization strategies can be found in [11]. Classical local strategies, used in the geophysical context, are the line-search algorithms such as the steepest descend method or the conjugate gradient one. Generally, the use of local strategies to solve a global optimization problem can be done only in particular cases, such as:

- **convex optimization problem**, in which a local minimum is also global one (see Figure 5.2a);
- the **starting model** x_0 is in the basin of attraction of the global minimum $\bar{x}$ (see Figure 5.2b).

In both cases, the use of a local strategy assures the convergence of the algorithm to the global minimum.

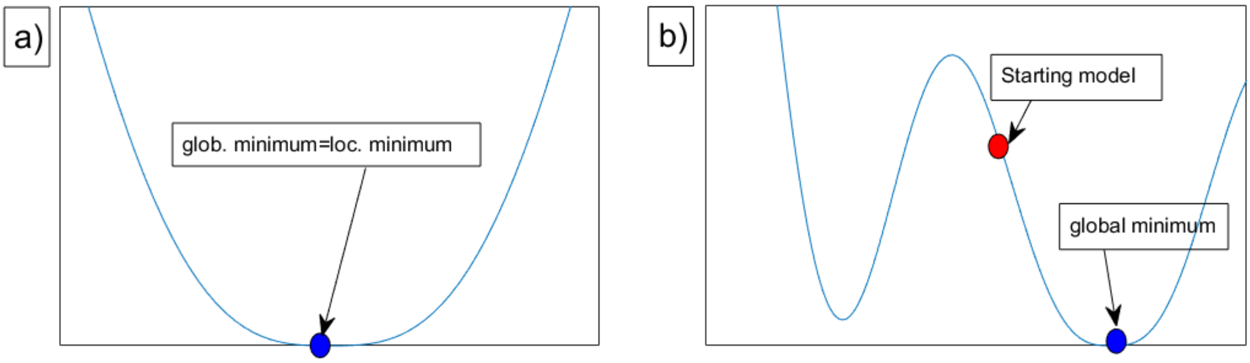


Figure 5.2: a) Example of a convex objective function and b) a non-convex objective function, where the local optimization procedure starts in the basin of attraction of the global minimum.

However, many optimization problems are characterized by a nonlinearity of the objective function and the presence of many local minima. In these cases, the use of a local strategy for finding the global minimum is not possible because of the risk to be entrapped in a local minimum, if the starting model is not in the basin of attraction of the global minimum.

In this context, global optimization algorithms represent search methods that try to estimate the global minimum of an objective function using:

- an **exploration strategy**, to find the more promising regions in the search domain, in which the global minimum can be located;
- an **escape mechanism**, to avoid being entrapped in a local minimum;
- an **exploitation strategy**, to probe a limited, but promising, region of the search domain and refine the search of the global minimum in this region.

Many algorithms have developed over the years, based on very different strategies. For a complete review of most of these strategies, the reader can see, for example, [41]. Figure 5.3 sketches a rough taxonomy of global optimization methods from the book of Weise.

Generally, optimization algorithms can be divided in two basic classes:

- **deterministic algorithms**, based on enumeration, generation of cuts and bounding in such a way that a part of the search domain is proved not to contain any optimum solution;

- **stochastic algorithms**, based on the random generation of feasible trial points and nonlinear local optimization procedures to explore the search domain.

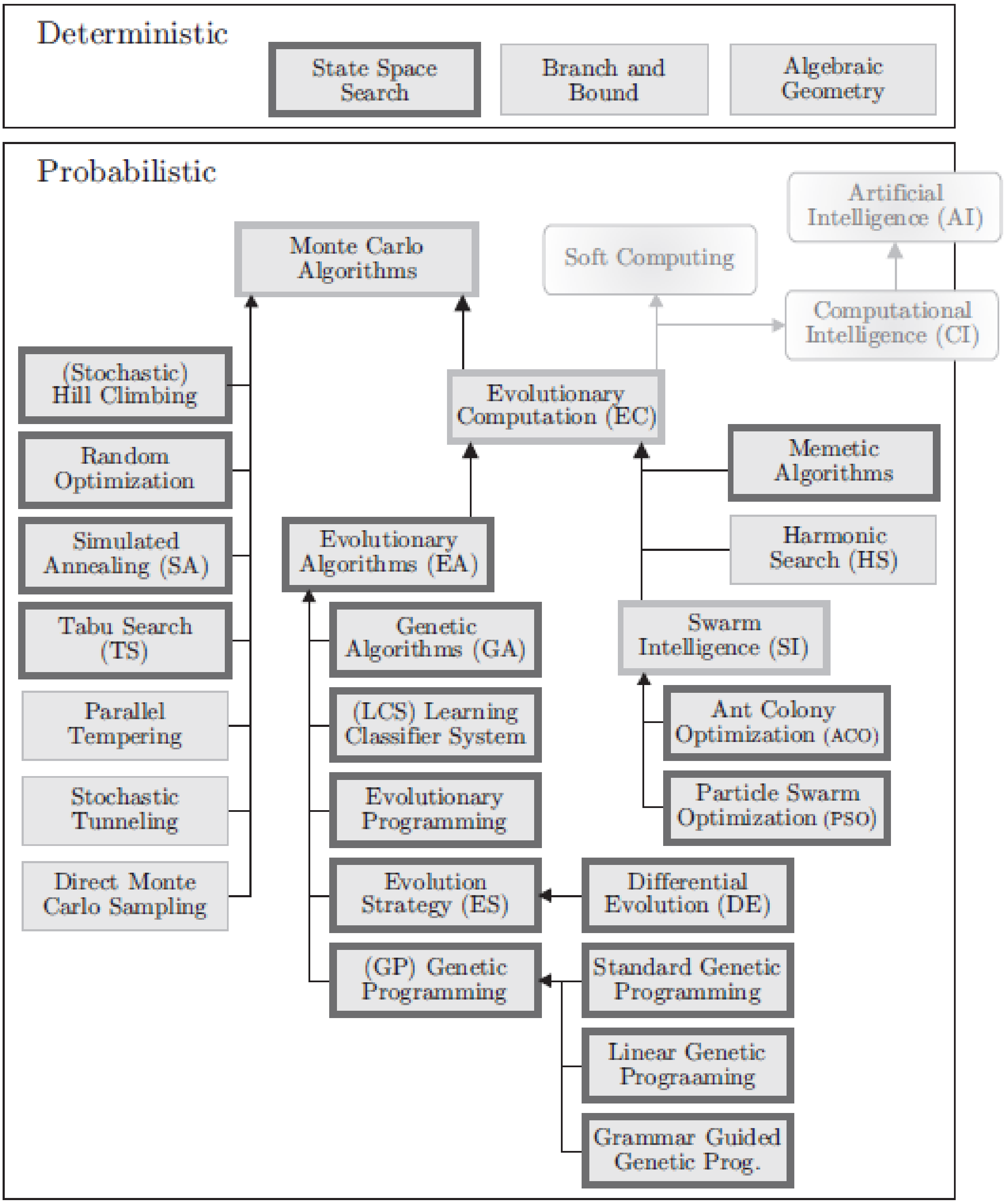

Deterministic
State Space Search
Branch and Bound
Algebraic Geometry
Probabilistic
Artificial Intelligence (AI)
Soft Computing
Monte Carlo Algorithms
Computational Intelligence (CI)
(Stochastic) Hill Climbing
Evolutionary Computation (EC)
Random Optimization
Memetic Algorithms
Simulated Annealing (SA)
Evolutionary Algorithms (EA)
Harmonic Search (HS)
Tabu Search (TS)
Genetic Algorithms (GA)
Swarm Intelligence (SI)
Parallel Tempering
(LCS) Learning Classifier System
Ant Colony Optimization (ACO)
Stochastic Tunneling
Evolutionary Programming
Particle Swarm Optimization (PSO)
Direct Monte Carlo Sampling
Evolution Strategy (ES)
Differential Evolution (DE)
(GP) Genetic Programming
Standard Genetic Programming
Linear Genetic Prograaming
Grammar Guided Genetic Prog.

Figure 5.3: The taxonomy of global optimization algorithms, taken from [41].

Algorithms belonging to the first class are most often used if the dimensionality of the search domain is low and some a priori information about the structure of the objective function is available (e.g. the concavity, the difference of convex function, the Lipschitz continuity). A typical example of a deterministic algorithm is the branch and bound method [42]. Algorithms belonging to the second class do not require any assumption about the objective function and are based often on the use of

heuristic techniques. These last ones represent the components of an optimization algorithm that uses information currently gathered to help to decide which candidate solution should be tested next or how the next individual can be produced. They combine objective function values and reasoning rules in a somewhat abstract and hopefully effective way, in many cases inspired by natural phenomena, such as biological evolution or the social behavior of animals, or physical process. In my work, I have focused the study and the research on two of these heuristic methods: the **Simulated Annealing** and the **Genetic algorithms**.

5.2 Simulated annealing

The **simulated annealing** (SA) method is a stochastic optimisation method developed by [43]. Originally proposed for objective functions with a discrete search domain, in [44] some modifications are proposed to apply to optimization of a function defined in a continuous domain.

The idea of this algorithm was to develop a minimization strategy inspired by the mechanism of cooling of the metals. In metallurgy, to remove possible defects of the crystalline structure, a metal is annealed to bring it back to the molten state. After the annealing, a slowly cooling process follows to obtain a new possible crystalline structure without defects. To remove all the defects, the sequence of annealing and cooling can be performed several times.

In analogy with this process, a minimization on a function $f(x)$ represents a cooling process in which:

- the global minimum $\bar{x}$ corresponds to a metal without any defects in the crystalline structure;
- the local minima correspond to a metal with some defects in the crystalline structure;
- the algorithmic procedure to find the global minimum, avoiding to be entrapped in a local one, corresponds to the process of cooling and annealing.

Following this philosophy, the simulate annealing creates an iterative sequence of models $\{x_k\}_{k\geq 0}$, where k is the index of the iteration of the algorithm to reach the global minimum.

For each iteration k, a new candidate model y is generated in a neighbourhood of the current model using the generation formula:

$$y = x_k + \Delta x_k(T_g, p), \tag{5.3}$$

where Δx_k is the step size, which depends on a parameter T_g, called the **generation temperature**, and p is a random number uniformly distributed over $[0,1]$. The step size is proportional to T_g.

Once the candidate model y is created, the algorithm chooses whether the model is accepted or not. If the candidate is rejected, the algorithm proceeds from the current model $x_{k+1} = x_k$, otherwise the candidate model becomes the current model $x_{k+1} = y$. The new model is always accepted if the new value of the objective function is lower, whereas if this value is higher, the model is accepted with a probability value dependent on a parameter called the **acceptatance temperature** T_a.

The number of accepted models until the iteration k, is indicated with k_a. This procedure is expressed with the following formula, called **Metropolis choice** [45]:

$$x_{k+1} = \begin{cases} y, & p \leq \min(1, e^{-\frac{f(y)-f(x_k)}{T_a}}), \\ x_k, & \text{otherwise} \end{cases} \tag{5.4}$$

where p is a random number uniformly distributed over $[0,1]$. The possibility of randomly accepting models with higher objective function values permits to escape from local minima, simulating the annealing process of the metals. Figure 5.4 shows a possible sketch of an iterative minimization, using SA, whereas Figure 5.5 gives an outline of how a generic SA algorithm works.

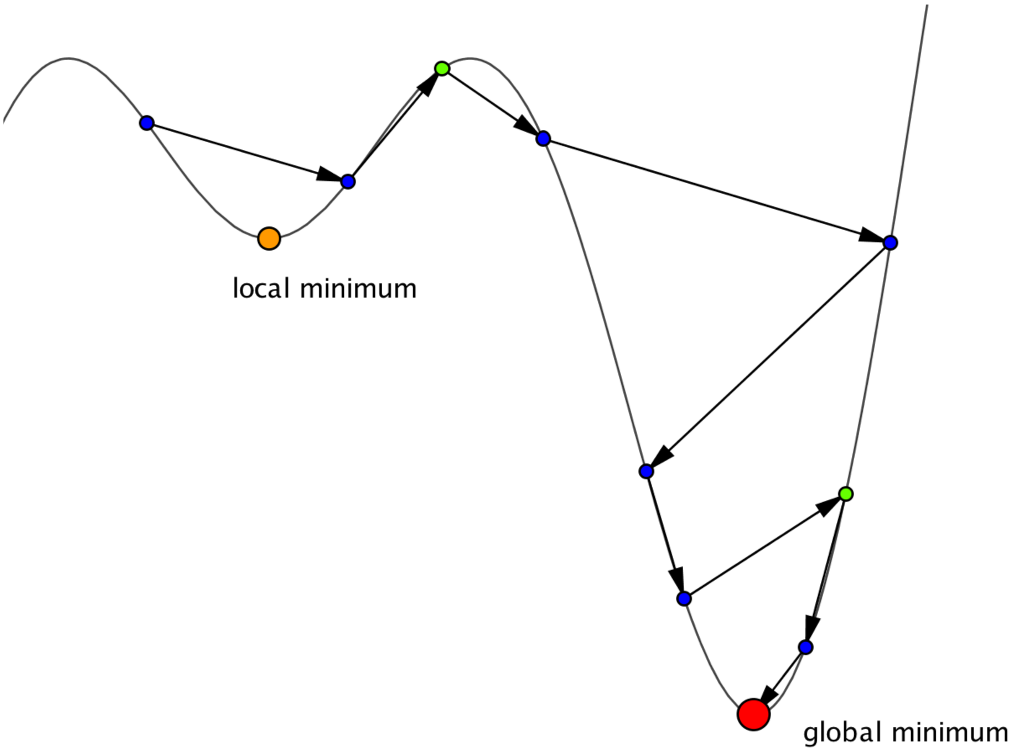


Figure 5.4: Sketch of an optimization procedure using SA.

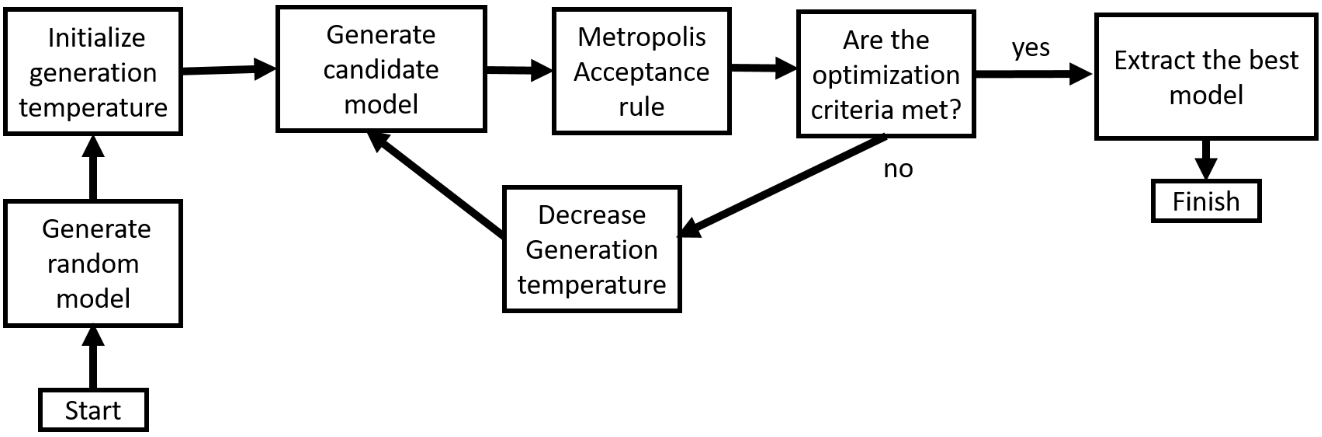

Initialize generation temperature
Generate candidate model
Metropolis Acceptance rule
Are the optimization criteria met?
yes
Extract the best model
no
Decrease Generation temperature
Generate random model
Start
Finish

Figure 5.5: outline of how a general SA algorithm works.

At the early stages of the optimisation, the initial generation and acceptatance temperatures ($T_{a,0}, T_{g,0}$) must be set to high values to allow a wide exploration of the model space. Subsequently, during the optimisation, their values are progressively reduced to the final values ($T_{a,f}, T_{g,f}$) to focus the search on the most promising zones of the model space. The rate at which the temperatures decreases is called **cooling rate**. As suggested in [46], it is a good practice to increase the initial generation temperature and to reduce the cooling rate as the dimension of the model space increases

Different implementations of the SA can be found in the literature, which differ for the types of step size Δx_k, generation temperature T_g and acceptance temperature T_a used. A complete description of the many variants of SA can be found, for example, in [47].

In my work I consider a variant of SA called **Adaptive Simulate Annealing (ASA)** developed by [48]. The main characteristics of this implementation are:

- an acceptance temperature characterized by an exponential decrease with respect to the number of accepted model k_a:

  $$T_{a,k+1} = T_{a,0}e^{-c_a \sqrt[n]{k_a}}, \tag{5.5}$$

  where c_a is a scalability factor, and $T_{a,0}$ is the initial temperature. The acceptance temperature T_a reaches his final value $T_{a,f}$ after a given number of accepted model n_{acp}. Afterwards, it remains constant.
- A generation temperature different for each i-th direction of the model space, characterized by an exponential decrease with respect to the number of iteration k, respectively:

  $$T^i_{g,k+1} = T^i_{g,0}e^{-c \sqrt[n]{k}}, \tag{5.6}$$

  where n is the dimension of the model space, c_a is a scalability factor and $T^i_{g,0}$ are the initial generation temperatures for the different components i. The generation temperatures T^i_g reach thier final values $T^i_{g,f}$ after a given number of iteration n_{iter}. Afterwards, it remains constant.

- Different generation formula for each i-th direction of the model space:

  $$y^i = x^i_{k+1} + \Delta x^i_k (B_i - A_i), \tag{5.7}$$

  where $i = 1, \dots, n$, $x_i \in [A_i, B_i]$ and Δy^i_k is obtained by the following formula:

  $$\Delta x^i_k = sgn\left(p^i - \tfrac{1}{2}\right)\left(\left[1 + \frac{1}{T^i_{g,k}}\right]^{|2p^i - 1|} - 1\right)T^i_{g,k}, \tag{5.8}$$

  with $p^i \in [0,1]$ uniform at random.
- The possibility to increase again the acceptance and the generation temperatures (reannealing) after a certain number of iteration as a function of k and k_a respectively. The increase of the generation temperature of a single component $T^i_{g,k}$ is inversely proportional to the absolute value of partial derivatives $\left|\frac{\partial f(x_k)}{\partial x^i}\right|$, where x_k is the current model of the iteration process.

5.3 Genetic algorithm

Genetic algorithm (GA) is a search algorithm developed by [49] that belongs to the larger class of evolutionary algorithms. The idea of Holland was to study the phenomenon of the adaptation in the natural contest to develop possible strategies inspired by the mechanisms of genetic evolution. Indeed, biological evolution is a research method within a large number of possible solutions to create individuals with a strong ability to survive and reproduce. For example, the immune system of the mammals is an extraordinary solution which has evolved to defend them by the bacteria. In this regard, evolutionary mechanisms such as selection, mutation or recombination can suggest efficient research strategies to solve the optimization problem.

Figure 5.6 gives an outline of how the algorithm works. The first step to solve a minimization problem using a genetic algorithm consists of the creation of a set, called **population**, of N_{ind} randomly individuals $\{x\}$, each one representing a candidate solution for the global minimization problem. The value of the objective function $f(x)$ is computed for all the individuals. If the optimization criterion is met for some of this individual, the algorithm is stopped. Otherwise, another population is formed combining the best individuals of the previous generation, called **parents**, obtained by **ranking** and **selection** mechanisms, and new individuals, called **offsprings**, obtained using mechanisms of **recombination** and **mutation** on the parents. This iterative process is repeated until the stopping criterion is met.

The **ranking operation** consists of ordering all the individuals of a population based on their value of the objective function: the lower the value of the objective function for an individual, the greater is its position in the ranking. The position of an individual in the ranking is indicated with Pos. Thus, the individual with the lower objective function value has $Pos = 1$, the greater has $Pos = N_{ind}$.

Then a **fitness function,** $Fitness(Pos)$, which depends on position Pos is used to measure the probability of everyone to be selected as a parent for the new population.

The **selection operation** consists of choosing a certain number of individuals from the population based on their value of the fitness function. An individual with a high fitness value has more probability to be selected as parent compared with respect to one with a lower fitness value. The total number of individuals to be selected as parents is controlled by a **selection-rate** parameter $sel_{rate} \in [0,1]$. For example, $sel_{rate} = 0.6$ means that the number of parents in a population of 100 individuals is 40.

The **recombination operation** produces new offsprings using the information contained in two or more *parents*, combining the variable values of the parents.

The **mutation operation** randomly alters the offsprings obtained by recombination operation. These variations are mostly small and are applied to the variables x_i of the individuals with a low probability, called **mutation-rate** parameters, that is inversely proportional to the number of variables n. For example, it is possible to choose $mut_{rate} = \frac{1}{n}$ to produce good results for a wide variety of test functions [50]. In general, a good mutation operator should often produce small step-sizes with a high probability and large step-sizes with a low probability.

Different implementations of the genetic algorithm can be found in the literature, which differs in function for mechanisms used for ranking, selection, recombination and mutation operators. A complete description of many variants of GA can be found, for example, in [51] and [52].

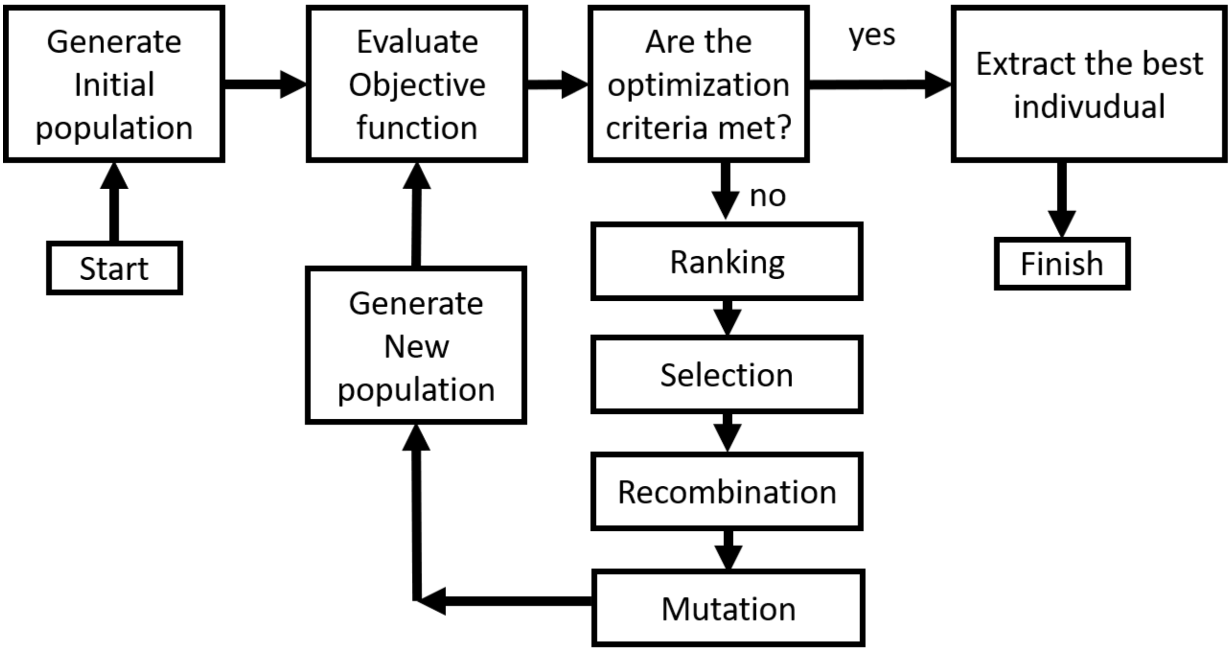

Generate Initial population
Evaluate Objective function
Are the optimization criteria met?
yes
Extract the best indivudual
Start
no
Ranking
Finish
Generate New population
Selection
Recombination
Mutation

Figure 5.6: Outline of a generic algorithm GA.

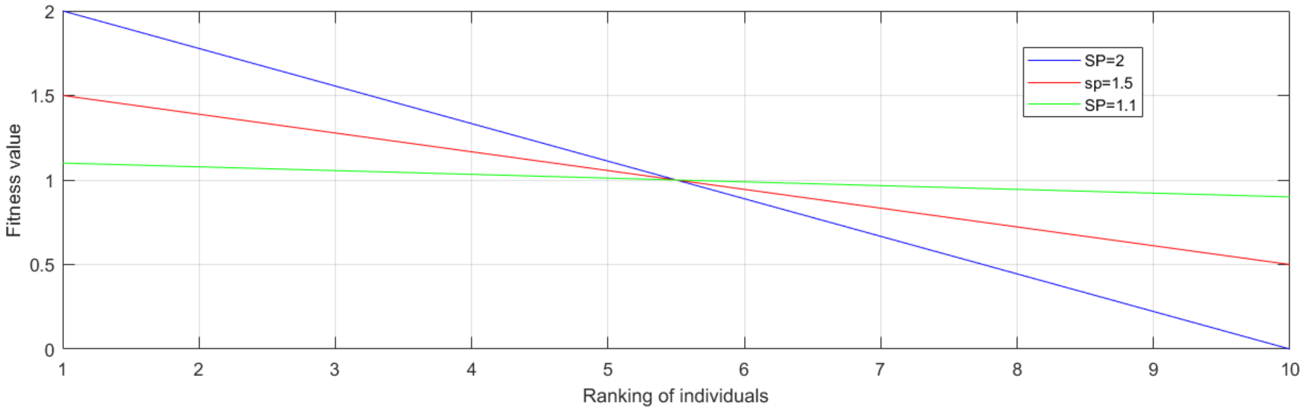


Figure 5.7: Example of fitness assignment using a linear-ranking for a population of 10 individuals and different choice of selection pressure.

I consider a real-coded GA version developed in [53]. The main characteristics of this implementation are:

- a fitness function based on a **linear ranking** of the individuals [54], using the following formula

$$Fitness(Pos) = 2 - SP + 2 \cdot (SP - 1) \cdot \frac{(Pos-Nind)}{(1-Nind)}, \qquad (5.9)$$

  where SP is the selection pressure, that can vary between 1 and 2. Figure 5.7 shows an example of linear ranking for a population of 10 individuals and different selection pressure;

- a **stochastic universal sampling method** for the selection operation [55], in which all the individuals of the population are mapped to contiguous segments of a line. The line length is equal to 1 and the length of each individual's segment is proportional to its fitness value. Here equally $Npointer$ spaced pointers are placed over the segment, where $Npointer$ is the number of the individuals, $\frac{1}{Npointer}$ is the distance between two consecutive pointers, and the position of the first pointer is given by a randomly generated number in the range $\left[0, \frac{1}{Npointer}\right]$.

Figure 5.8 shows an example of the application of stochastic universal sampling method, in which the number of individuals is 10, the selection pressure is 2 and $Npointer$ is 8.

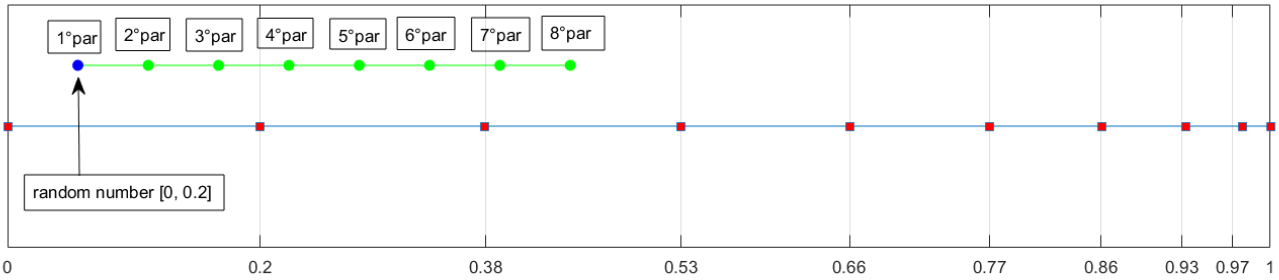


Figure 5.8: Example of universal sampling: the individuals are 10, the selection pressure is 2 and Npointer is 8

- an **intermediate recombination method** [50] for the recombination operation, in which the components x_i of a new individual x are chosen somewhere around and between the variable values of two parents, randomly chosen randomly with equal probability in the set of all the parents. A new offspring is produced according to the following formula:

$$Var_i^0 = a_i \cdot Var_i^{p1} + (1 - a_i) \cdot Var_i^{p2}, \tag{5.10}$$

with $i = 1, \ldots, n$, $a_i \in [-0.25, 1.25]$ uniform at random for each i. Figure 5.9 shows the area of the variable range of the offspring defined by the variables of the parents;

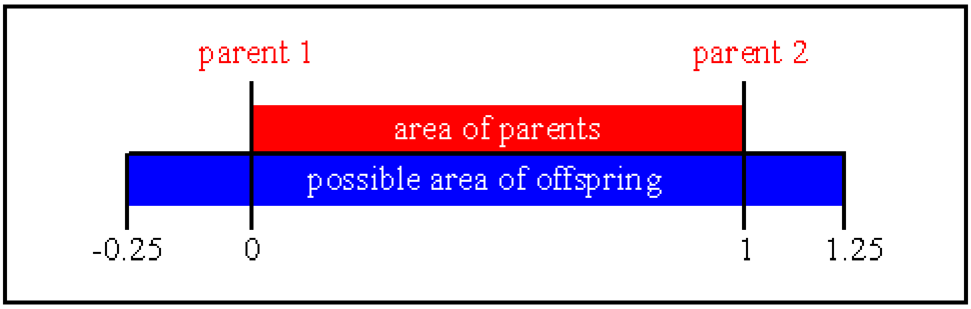


Figure 5.9: Area for variable value of offspring compared to parents in intermediate recombination.

- a **real variables mutation operator** [50] based on the creation of randomly values that are added to the variables x_i of a new offspring x, according to the following formula:

$$Var_i^{Mut} = Var_i + s_i * r * Range_i * 2^{-u*k}, \tag{5.11}$$

with $s_i \in \{-1, 1\}$ uniform at random, r mutation range, $u \in [0,1]$ uniform at random and k **mutation precision**. Thus, the mutation steps are created inside the area $[r * 2^{-k}, r]$. Typical parameters of the mutation operator are $k \in \{4, 5, \ldots, 20\}$ and $r \in [10^{-6}, 0.1]$. Figure 5.10 shows a sketch of how an individual in two dimensions can be changed by the mutation operations.

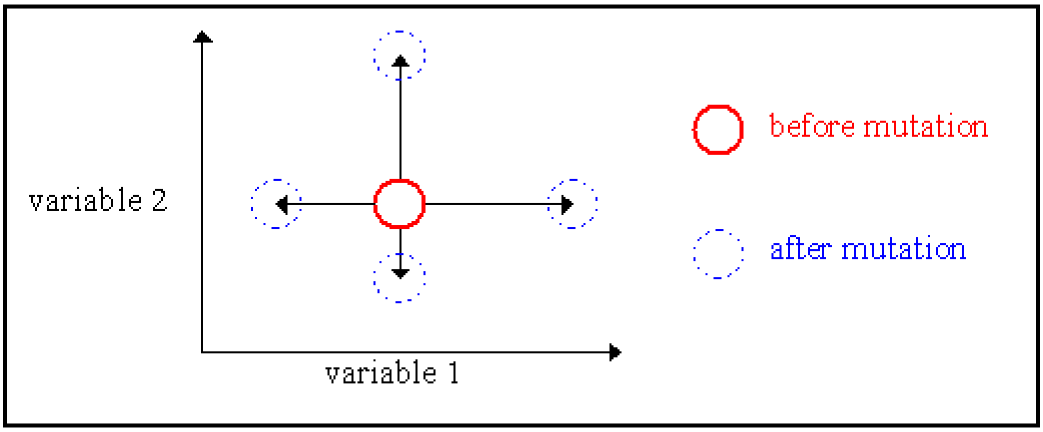


Figure 5.10: Possible mutations for an individual in two dimensions.

5.4 Conclusions

In this chapter, I studied two global optimization algorithms to use in the context of FWI: the Adaptive Simulated Annealing and the Genetic algorithms. These use very different heuristic techniques to guide the optimization procedure in the search domain: the first uses a minimization strategy inspired by the cooling mechanism of the metals, to escape from possible local minima of the objective function; the second uses strategies inspired by the mechanisms of genetic evolution such as selection, mutation or recombination.

Chapter 6.
Global optimization algorithms on analytic functions

In this chapter, I illustrate the results obtained by the application of ASA and GA to find the global minimum of some analytic test functions. The obtained results are published in a paper in the journal *Geophysical Prospecting*. See [56] for more details.

6.1 Introduction

I have evaluated the performance of ASA and GA using four analytic test functions with very different characteristics: the sphere function, which is symmetric and convex; the Rosenbrock function, which is convex but with the minimum in an elongated flat valley; the Rastrigin function, which has many regularly distributed local minima and the Schwefel function, which has the global minimum at the border of the search space and several irregularly distributed local minima. Each of these functions is defined for an integer dimension *n* of the model space, except the Rosenbrock function, which is defined only for even dimensions.

Table 6.1 summarizes the features of each function: separability, multimodality, and symmetry.

Function	Separable?	Multimodal?	Symmetric?
Sphere	yes	no	yes
Rosenbrock	no	no	no
Rastrigin	yes	yes	yes
Schwefel	yes	yes	no

Table 6.1: summary of the characteristics of the analytic test functions.

Many other analytic test functions can be found, for example, in [57].

6.2 Setting the experimental frameworks

A specific framework is set up to compare the performances of the algorithms on the different analytic functions. Since each objective function has a typical search domain different from another test function, I have rescaled the search domain of each objective function such that we can consistently use the same domain for the model parameters, that is the hypercube $[-5,5]^n$. The same convergence criterion is the following: an algorithm converges during an optimization test when it finds a model $\boldsymbol{x} \epsilon [-5,5]^n$ that satisfies the following accuracy criterion:

$$\sqrt{\frac{\sum_{i=1}^{n}\left(x_i - x_i^{glob}\right)^2}{n}} < \epsilon \qquad (6.1)$$

where x^{glob} represents the global minimum, and the accuracy ϵ is 0.05.

If the convergence criterion is not satisfied after a certain number of evaluated models, the algorithm could remain entrapped in a local minimum unable to escape from it. In this case a test is considered as failed. Anyway, a test is considered as failed even when it does not reach the global minimum within the selected accuracy after 10^7 evaluated models.

A total number of 100 tests are performed for each algorithm, for each test function, and for each dimension of the model space, to produce statistically significant results. From the subset of tests where the algorithm satisfies the criterium test, I derive the mean value and the standard deviation of the number of evaluated models, to estimate the convergence rate and its variability as a function of the dimension of the model space. The number of successful tests is used to compute the percentage of convergence of the algorithms as a function of the dimension of the model space and the type of test function.

6.3 Setting the parameters of the algorithms

Some of the main parameters of the algorithms are set according to values that can be found in literature in particular, according to the dimension of the model space and the complexity of the objective functions.

Table 6.2 shows the ASA parameters: according to [46] the value of T_{g0}, T_{gf} and n_{iter} increase with the dimension of the model space n. Multiple cooling/heating cycles (reannealing) are used for multi-minima functions, every 1000 iterations or 100 accepted models.

Table 6.3 shows the GA parameters: according to [50] the size of the GA population is increased linearly with n, and the mutation rate decreases linearly with n^{-1}. The selection rate is 0.8.

ASA parameters	Sphere	Rosenbrock	Rastrigin	Schwefel
T_{g0}	10n	10n	10n	10n
T_{gf}	$10^{-19}n$	$10^{-19}n$	$10^{-19}n$	$10^{-19}n$
n_{iter}	100n	100n	100n	100n
Reannealing	no	no	yes	yes

Table 6.2: Control parameters for the ASA method used in the tests on the analytic objective functions; n indicates the number of dimensions.

GA parameters	Sphere	Rosenbrock	Rastrigin	Schwefel
Nind	10n	10n	10n	*100n*
Sel-rate	0.8	0.8	0.8	0.8
Mut-rate	n^{-1}	n^{-1}	n^{-1}	n^{-1}

Table 6.3: Control parameters for the GA method used in the tests on the analytic objective functions; n indicates the number of dimensions.

6.4 Test on the Sphere function

The first test compares the ASA and GA performances on the **Sphere function**, or De Jong's function n°1 [58]:

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{6.2}$$

This function is symmetric, convex and unimodal, with a unique minimum in [0,…,0]. Even if it is a simple function, it is possible to evaluate the ability of the different algorithms in a favourable scenario in which there is a single minimum of the objective function. Figure 6.1a and Figure 6.1b show this function in two dimensions as a surface in the three-dimensional space and as a projection onto a 2D map, respectively. Table 6.4 displays the percentage of success computed on the set of 100 tests for the two algorithms in a range of model-space dimensions from 2 to 60. The two algorithms successfully converge for all the values of *n*.

	2	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
	22	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
	42	**44**	**46**	**48**	**50**	**52**	**54**	**56**	**58**	**60**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table 6.4: percentage of success of ASA and GA on the sphere function, in a range of dimension from 2 to 60.

Figure 6.2 displays the rate of convergence of the two algorithms as a function of the dimension *n*. The ASA algorithm exhibits an exponential trend, that is a linear trend when using a semi-logarithmic plot, whereas GA is characterized by a polynomial trend, that is a logarithmic trend when using a semi-logarithmic plot. For medium-low dimensions (2<*n*<40), ASA is the best-performing algorithm, while for dimensions higher than 44, GA outperforms ASA.

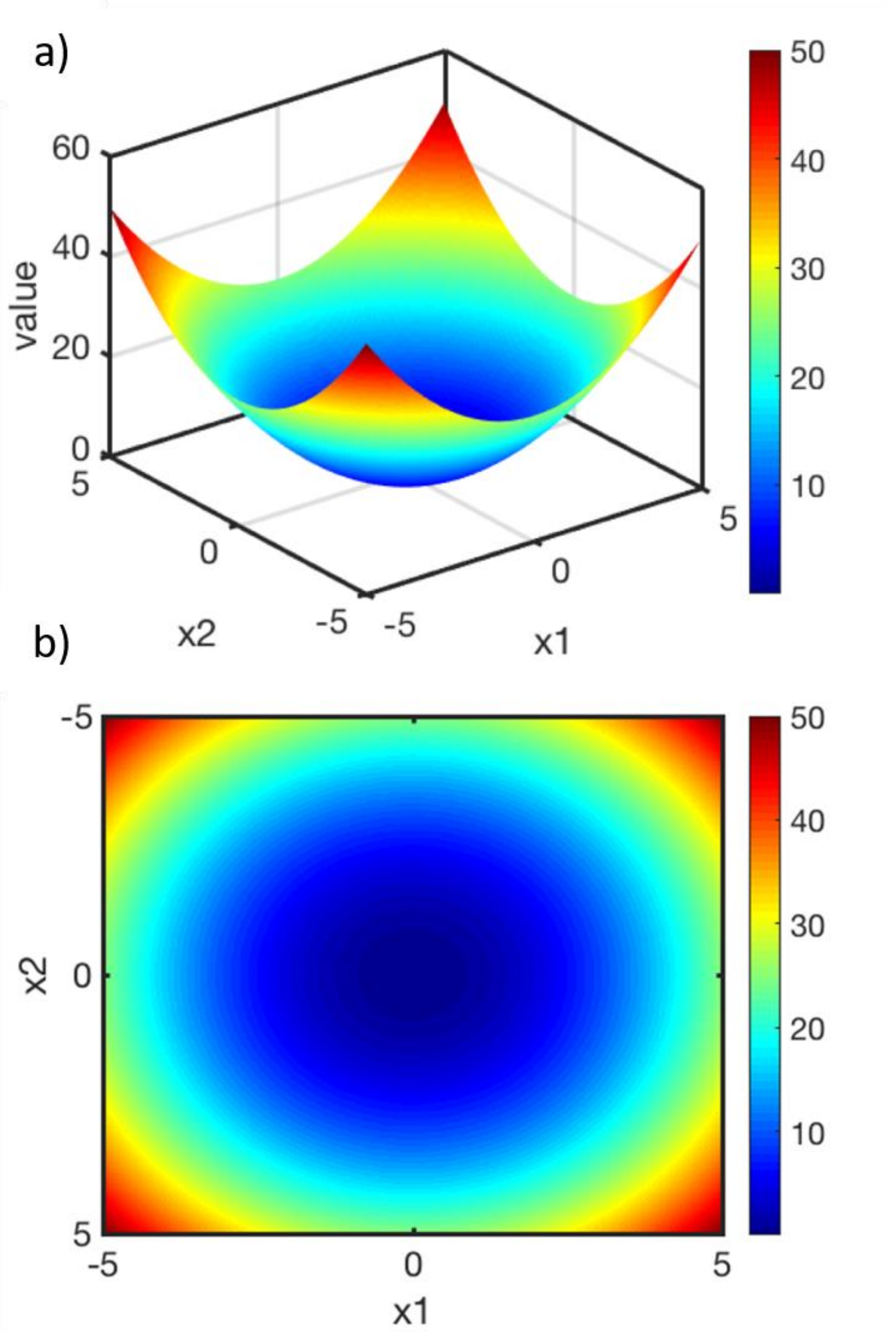


Figure 6.1: The sphere function with n=2, represented a) as a surface in 3D space and b) as a 2D projection.

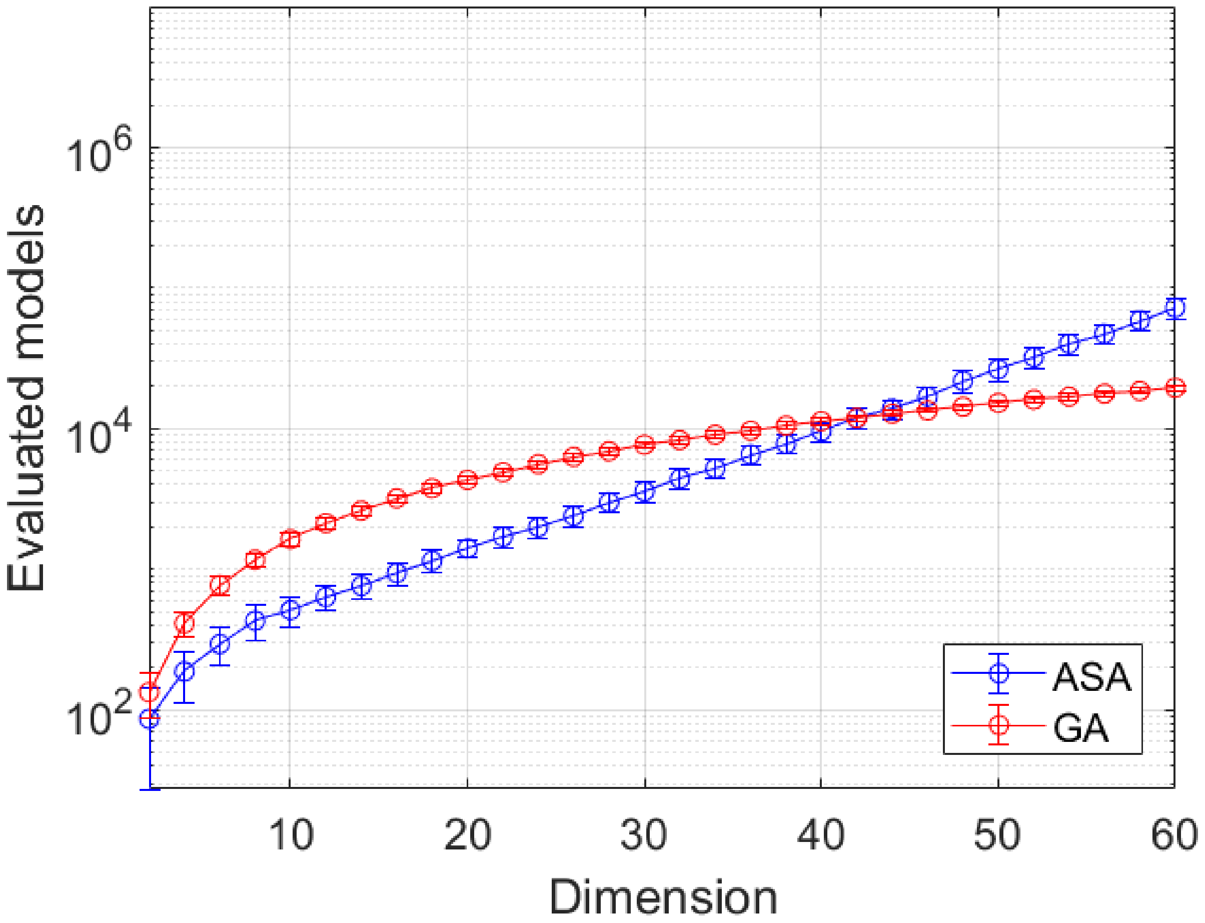


Figure 6.2: Curves of convergence for the sphere function computed on the ensemble of 100 tests performed for each method and for each dimension of the model space.

6.5 Test on the Rosenbrock function

The second test compares the ASA and GA performances on the **Rosenbrock function** [59]:

$$f(x) = \sum_{i=1}^{\frac{n}{2}} (\mathrm{A}(x_{2i-1} - x_{2i}^2)^2 + (1 - x_{2i-1})^2) \tag{6.3}$$

with $A = 100$ and n assuming only even values (for odd dimensions, the nature of the function changes, becoming a multi-minima function). Figure 6.3a Figure 6.3b show the Rosenbrock function with $n = 2$ as a surface in the three-dimensional space and as a projection onto a 2D map, respectively.

This function is convex with a single minimum located in the flat valley at $[2, \dots, 2]$. Finding the valley is trivial, but, because of the flatness of the valley, it is difficult to converge to the minimum.

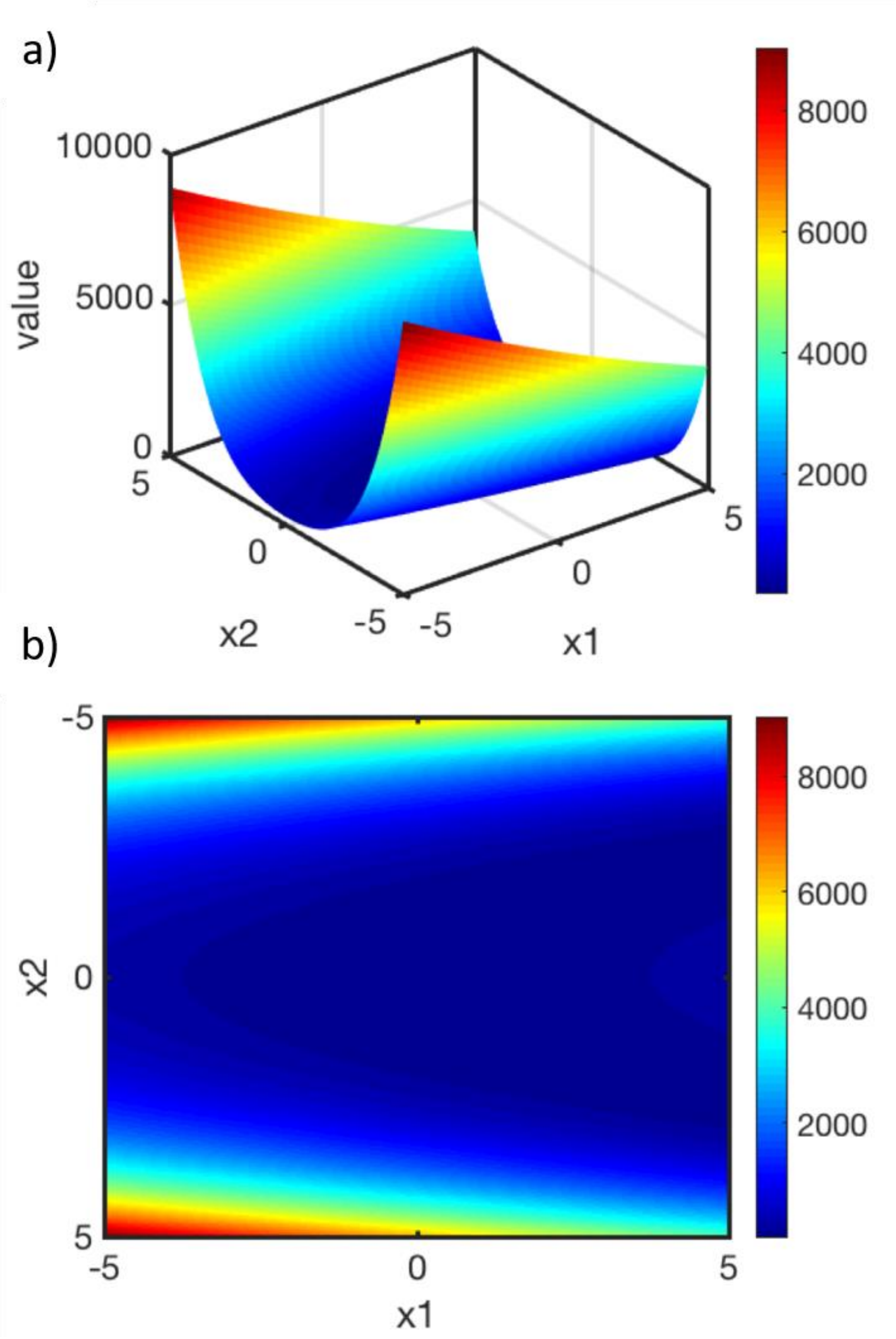


Figure 6.3: The Rosenbrock function with n=2, represented a) as a surface in 3D space and b) as a 2D projection.

For this reason, the number of evaluated models required to converge can be very high. Table 6.5 represents the percentage of success for dimensions varying from 2 to 40 for the two algorithms. Both

GA has complete success for all the dimensions. The ASA algorithm is completely successful for $n < 40$; then it becomes to fail because of the high number of evaluated models necessary to converge, higher than 10^7.

	2	4	6	8	10	12	14	16	18	20
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
	22	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	40%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table 6.5: Percentage of the success of ASA and GA on the Rosenbrock function, in a range of dimension from 2 to 40.

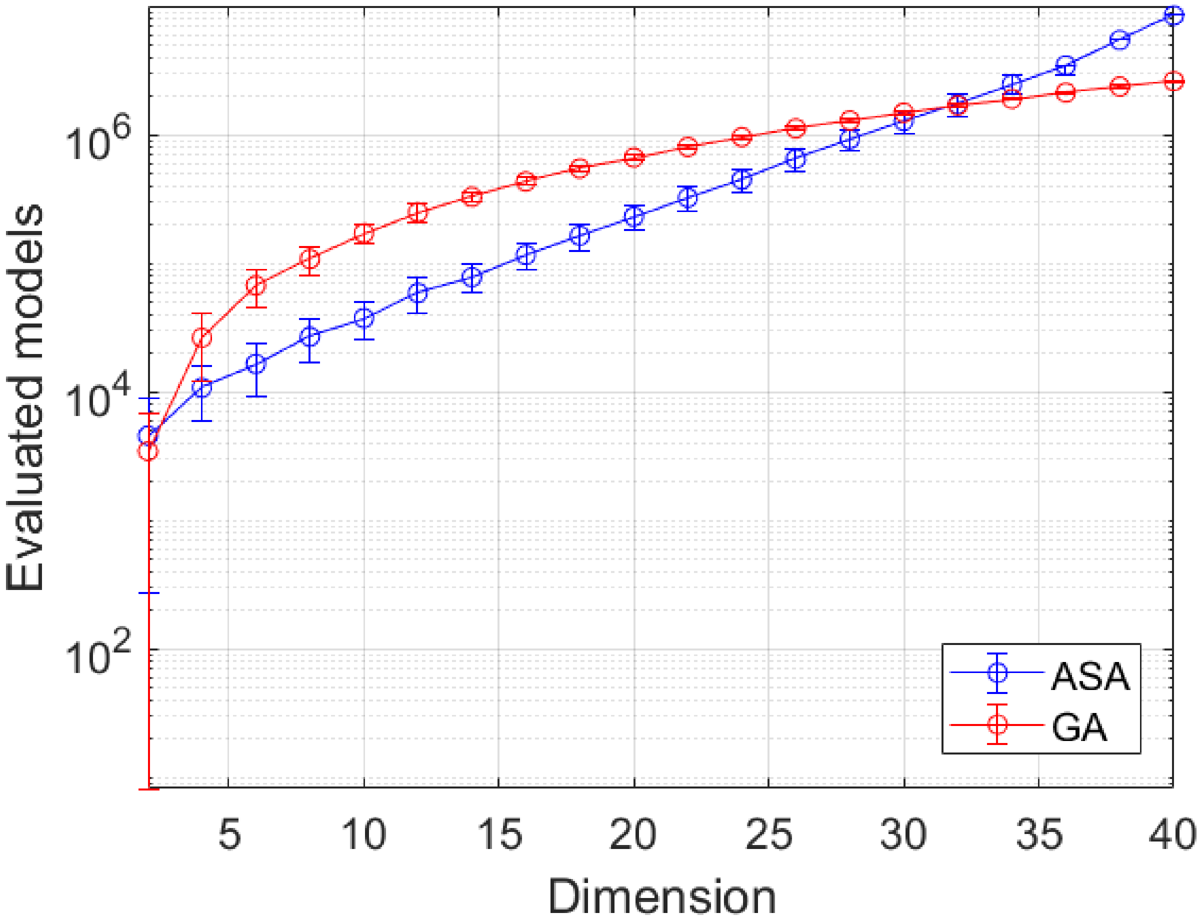


Figure 6.4: Curves of convergence for the Rosenbrock function, computed on the ensemble of 100 tests performed for each method and dimension of the model space.

Figure 6.4 shows the curves of convergence for the two algorithms up to $n = 40$. We note that their behaviour is similar to the sphere case. The ASA algorithm has an exponential trend, while GA shows polynomial trends. For medium-low dimensions (2<n<30), ASA is the best-performing algorithm, while for dimensions higher than 32, GA outperforms ASA. In particular, the ASA algorithm for dimensions greater than 40 exceeds the maximum number of evaluated models. Therefore, they are considered as failing to converge.

6.6 Test on the Rastrigin function

The third test compares the performances of the two algorithms on the **Rastrigin function** [60]:

$$f(x) = A\,n + \sum_{i=1}^{n} [x_i^2 - A\cos(2\pi x_i)] \tag{6.4}$$

where $A = 10$. Figure 6.5a and Figure 6.5b show the Rastrigin function as a surface in the three-dimensional space and as a projection onto a 2D map, respectively. This function is an example of a non-convex function, with a global minimum in $[0, \dots, 0]$ and a high number of local minima, which increases exponentially with the dimension of the model space. Precisely, this function has 11^n local minima for this range of the model space, i.e., $[-5,5]^n$.

Table 6.6 represents the percentage of success for dimensions varying from 2 to 40 for the two algorithms. The GA method successfully converges for all the dimensions, whereas the ASA algorithm fails after $n = 38$ because all the tests exceed the maximum number of model evaluations allowed.

	2	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
	22	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	40%	0%
GA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table 6.6: Percentage of success of ASA and GA on the Rastrigin function, in a range of dimension from 2 to 60.

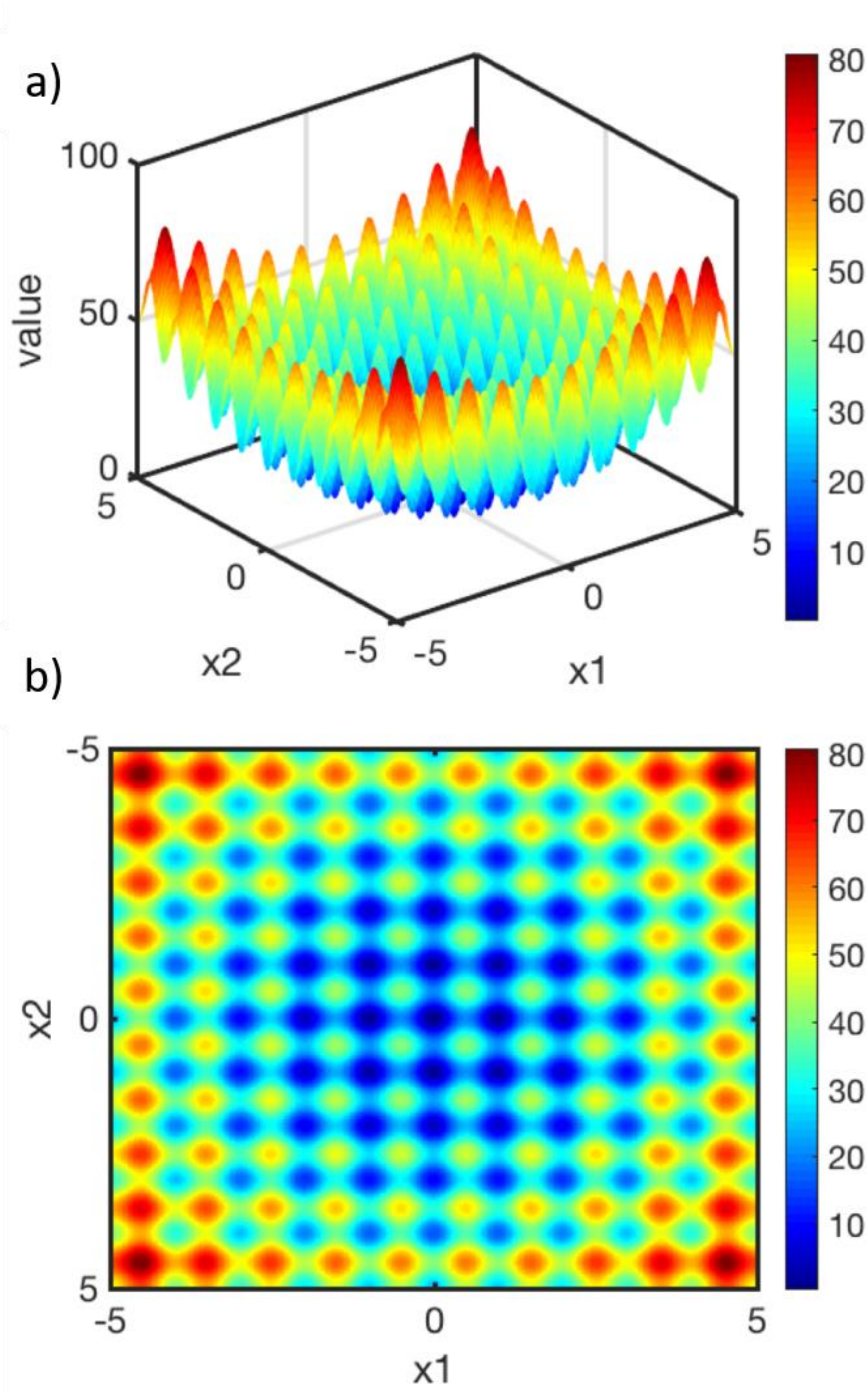


Figure 6.5: The Rastrigin function with n=2, represented a) as a surface in 3D space and b) as a 2D projection.

Figure 6.6 shows the curves of convergence for the two algorithms for n varying from 2 to 40. By comparing these curves with the ones of the sphere function we note an overall increment of the mean number of evaluated models for the two algorithms for all n, caused by the more complex nature of the Rastrigin function with respect to the sphere one. Nevertheless, the convergence trends for ASA and GA appear to be the same as before, i.e., an exponential trend for ASA and a polynomial trend for GA. The ASA algorithm results the best method for medium-low dimensions ($4 < n < 20$), but it shows an exponential convergence trend; on the contrary, GA, being characterized by a polynomial convergence trend, after a crossing point (at $n = 20$), eventually overcomes ASA in terms of performance for higher dimensions.

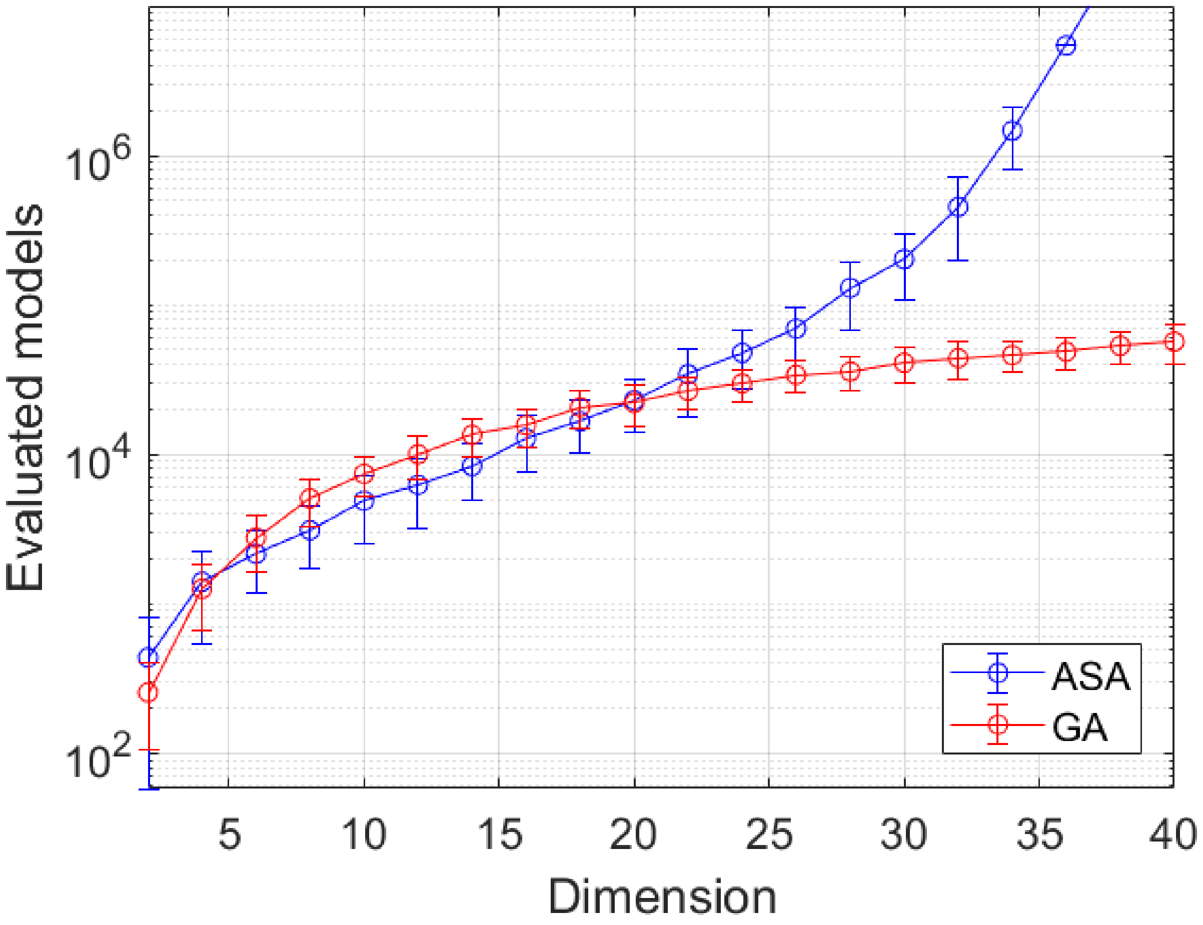


Figure 6.6: Curves of convergence for the Rastrigin function, computed on the ensemble of 100 tests performed for each method and for each dimension of the model space.

6.7 Test on the Schwefel function

The **Schwefel function** [61] rescaled in the $[-5,5]^n$ range is:

$$f(x) = A\,n - 100 \sum_{i=1}^{n} [x_i \cdot \sin(10\,\sqrt{|x_i|})] \tag{6.5}$$

where A=418.9829. This function has a large number of local minima equal to 7^n. However, differently from the previous function, in which the local minima surround the central global minimum, in the Schwefel function the local minima are more irregularly distributed, and important local minima are distant from the non-centred global minimum, which lies at $[4.209687, \ldots, 4.209687]$.

Figure 6.7a and Figure 6.7b show the 2D Schwefel function as a surface in the three-dimensional space and as a projection onto a 2D map, respectively.

Table 6.7 displays the percentage of success of the two methods with n varying from 2 to 40.

	2	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	100%	98%	90%	82%	69%	60%	53%	39%	42%	27%
	22	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**
ASA	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
GA	29%	18%	17%	14%	11%	0%	0%	0%	0%	0%

Table 6.7: Percentage of success of ASA and GA on the Schwefel function, in a range of dimension from 2 to 40.

ASA successfully converges for all tests, displaying 100% success for all dimensions. Regarding the GA method, it is completely successful only for dimensions up to 2. For higher dimensions, only a fraction of the set of 100 tests successfully converges to the global minimum. The percentage of successful tests in the ensemble of 100 tests quickly drops to zero in a range of n from 2 to 30 for GA. The increase of failures for GA is because they remain entrapped in local minima that are far from the global minimum.

Figure 6.8 shows the curves of convergence of the two algorithms. The ASA algorithm has an exponential trend and is clearly the best-performing algorithm. For the GA the small number of successful tests used to derive the convergence curve is insufficient to infer reliable convergence trend.

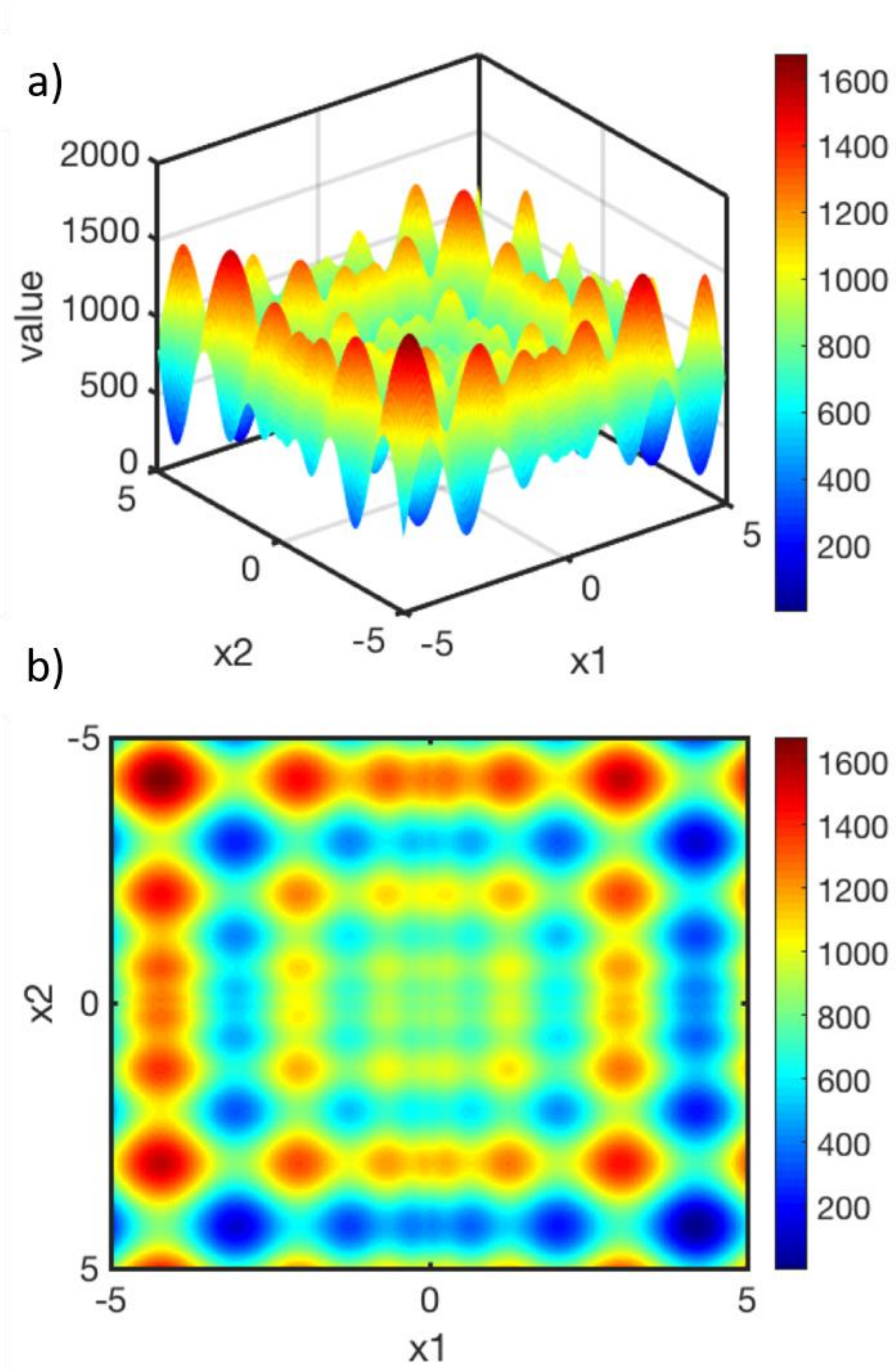


Figure 6.7: The Schwefel function with n=2, represented a) as a surface in 3D space and b) as a 2D projection.

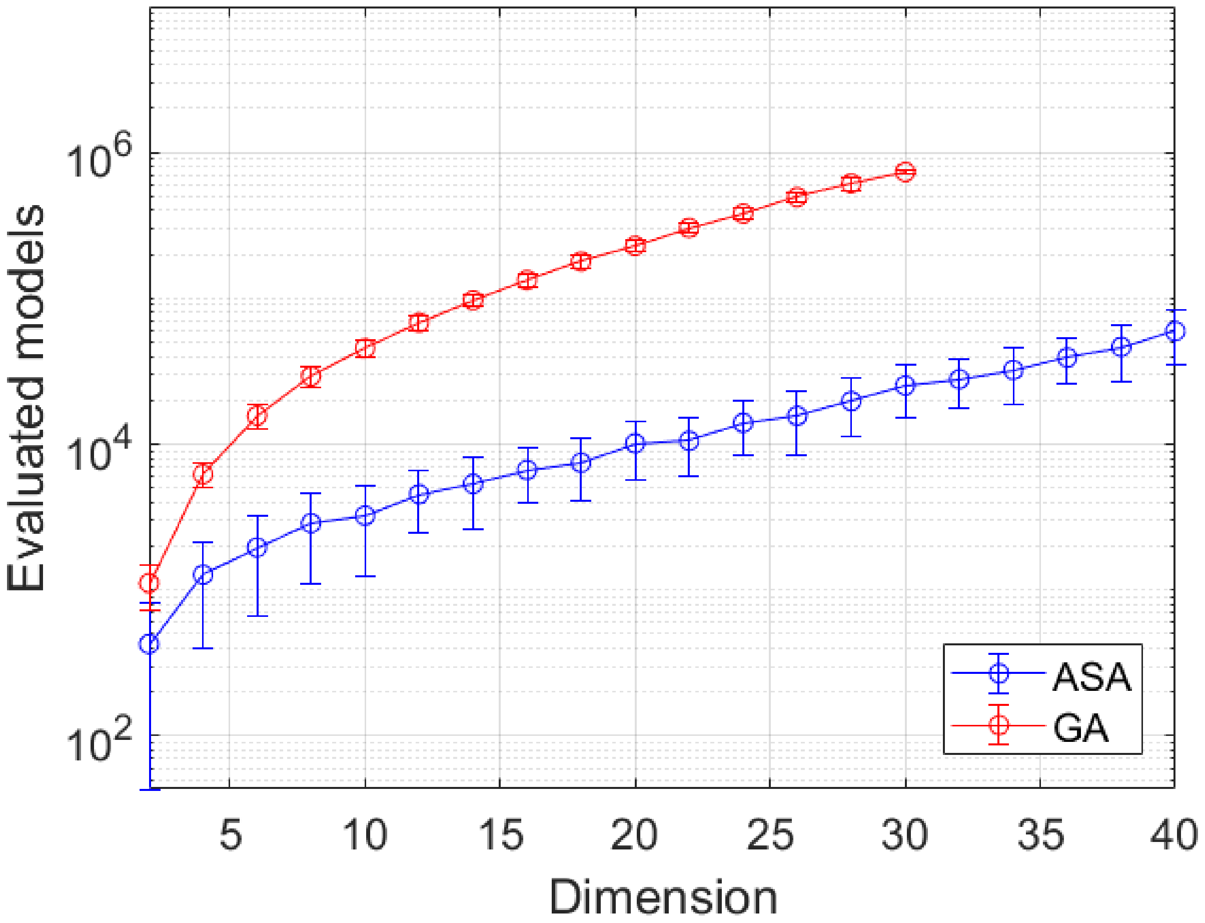


Figure 6.8: Curves of convergence for the Schwefel function, computed on the ensemble of 100 tests performed for each method and for each dimension of the model space.

6.8 Conclusions and results

Comparing the results obtained on the set of the four analytic functions I have observed that:

- in an ideal case such as asymmetric, convex, separable function (Sphere function) the two methods converge to the global minimum and the larger is the dimension of the model space, the larger is the number of evaluated model necessary to converge. In particular, I observe an exponential trend for the curve of convergence of ASA and a polynomial trend for GA. For lower dimensions of the model space, ASA appears to be the best algorithm. However, because of the better scalability with the number of unknowns, GA outperform ASA after a certain dimension of the model space;
- in case of a convex, but very asymmetric function characterized by a flattening of the function near the global minimum (Rosenbrock function), the two algorithms worsen their performances with respect to the sphere test, increasing the number of evaluated models necessary to converge. However, the trend of the curves of convergence appears very similar to the previous case. ASA is the best algorithm for low dimensions, whereas GA outperforms ASA after a certain dimension;

- in case of symmetric and non-convex function characterized by a large number of local minima (Rastrigin function), the two algorithms worsen their performances with respect to the sphere test, increasing the number of evaluated models. The trend of the curves of converges for ASA and GA is again exponential and polynomial, respectively. ASA is the best algorithm for low dimensions, whereas GA outperforms ASA for greater dimension. The number of evaluated models for the convergence is greater for the Sphere test but lower for the Rosenbrock test. In particular, the ASA algorithm can become very expensive regarding the number of evaluated models;
- in case of non-symmetric and non-convex function characterized by a large number of local minima (Schwefel function), the two algorithms worsen their performances with respect to the Sphere test increasing the number of evaluated models (ASA) or remaining entrapped in a local minimum (GA). The ASA algorithm has the same similar exponential trend of the previous tests. Its number of evaluated models for the convergence is greater than the Sphere function, but lower than the Rastrigin function, probably because of the lower number of local minima. The GA algorithm has no definite trend of convergence because of the high number of unsuccessful tests and appears to suffer from the irregular distribution of the local minima, which causes the entrapment of the algorithm in local minima far from the global one;

Starting from these results, I conclude that both algorithms have strengths and weaknesses:

1. for a low dimension of the model space the ASA algorithm appears to be the best algorithm in term of the number of evaluated models necessary to converge, whereas the GA algorithm is better for greater dimensions;
2. for both algorithms the presence of flatness zone in the model space, that means low sensibility of some parameters, worsen their performance;
3. for both algorithms the presence of a large number of local minima worsen their performances, but less than the presence of flatness zones;
4. the GA algorithm suffers the irregular distribution of local minima in the model space risking to be entrapped in one of them, whereas this phenomenon does not influence the ASA algorithm.

Chapter 7.
Experience of FWI on synthetic data

In this chapter, I illustrate the results obtained by the application of ASA and GA algorithms for a classical acoustic 2D FWI problem, which consists of the inversion of synthetic seismograms of the Marmousi model. See [62] for more details.

7.1 A classical inversion test: seismic acquisition on the Marmousi model

A common way to test the feasibility of a seismic inversion procedure is to test it in the inversion of synthetic seismograms, using some complex geological model. This procedure allows us to compare both the seismograms obtained at the end of an inversion procedure and the final best model with the true one. A standard test used in case of 2D acoustic FWI is the inversion of one or more synthetic seismograms simulated on a complex 2D acoustic structural model, called **Marmousi model**.

The Marmousi model (Figure 7.1) was formulated at the Institute Francais du Petrole (IFP). It was used for the workshop on practical aspects of seismic data inversion at the 1990 EAEG meeting in Copenhagen, where different groups (contractors, universities, and oil companies) applied their imaging tools on this data set. The possible seismograms relative to this model are generated usually using a 2D acoustic finite difference modelling program. See [63] and [64] for details.

The model is built to resemble a sedimentary basin originated during an overall continental drift with numerous large normal faults. The geometry is based somewhat on a profile through the North Quenguela in the Cuanza basin, situated in Angola, as described in [65]. It consists of 160 velocity layers and contains many reflectors, steep dips, and high velocity variations in both the lateral and the vertical direction. The velocity distribution varies from a minimum of $1500m/s$ to a maximum of $5500m/s$ with a realistic horizontal and vertical velocity gradient. The horizontal dimension is 9.200 meters, whereas the vertical one is 3000 meters.

The sources and the receivers are usually located near the top of the model in the water layer, to simulate a marine seismic acquisition. The recording time T, the time sampling dt and the type of wavelet $s(t)$ depend on the target we want to simulate.

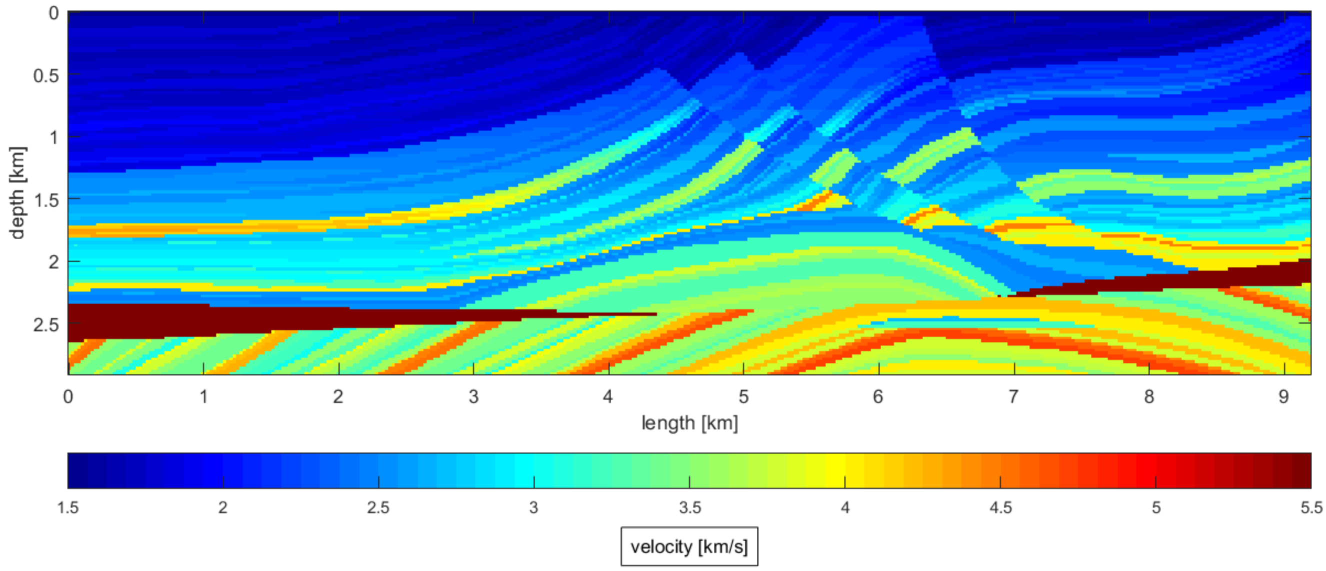


Figure 7.1: Marmousi model: a complex 2D acoustic model of a sedimentary basin with faults, reflectors, and strong velocity variations in both lateral and vertical directions.

7.2 Simulation of the observed seismograms on a portion of the Marmousi model

To simulate the synthetic seismograms, I have considered as a velocity model the one of Figure 7.2, that represents the central part of the Marmousi model, with a total length of $4608m$ and a depth of 1152m. A regular modelling grid, with $dx = 24m, n_x = 192$ and $n_z = 48$ (Figure 7.3), is used. The water layer is modelled by the first two rows of the grid. As acquisition layout, I have considered:

- 16 sources located at depth of $24m$, that corresponds to the first row of the modelling grid, and with a uniform distance of about $\Delta_s = 305m$ (the red points in Figure 7.4);
- 192 receivers for each source located at depth of $24m$, with a uniform distance $\Delta_r = 24m$ (the blue points in Figure 7.4);
- A time step of $dt = 0.002$s and a recording length of $T = 4s$;
- A Ricker wavelet with a peak frequency of $f_0 = 6\ Hz$ and a frequency range between 0 and $24\ Hz$. Figure 7.5a and Figure 7.5b display the wavelet as a function of time and temporal frequency, respectively.

Absorbing boundary conditions are set on the lateral and bottom sides of the model, with $n_r = n_l = n_b = 30$, and reflecting boundary conditions on the top side to simulate the sea-air interface. Note that the same receiver positions are used for all the sources and that the numerical stability and dispersion relations are satisfied with these modelling parameters.

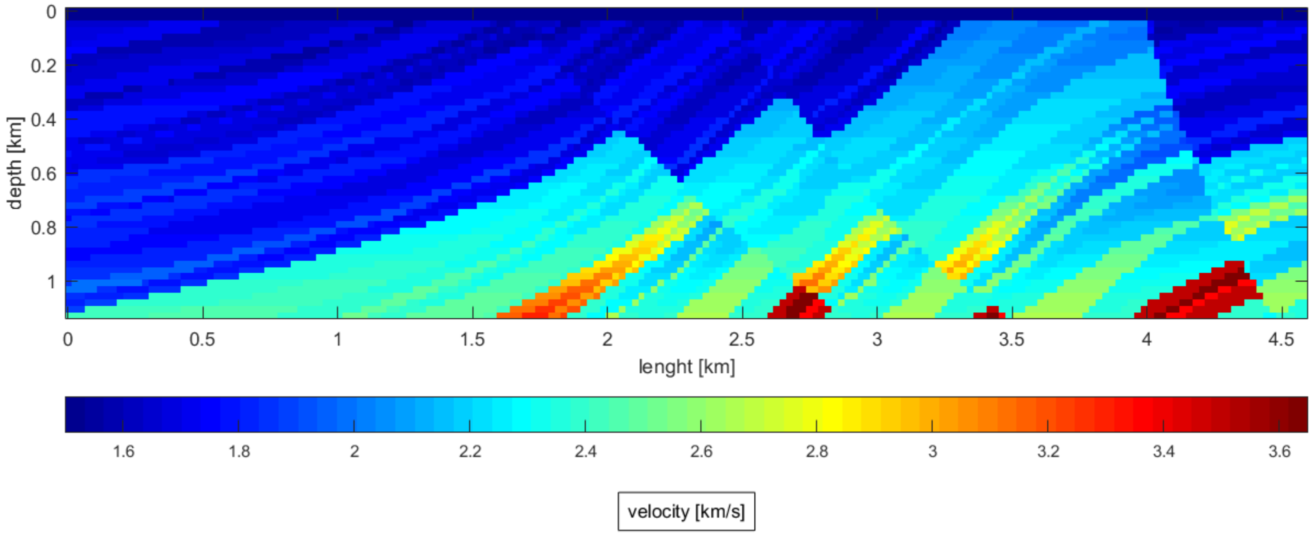


Figure 7.2: Portion of Marmousi model, that corresponds to the upper central part of it.

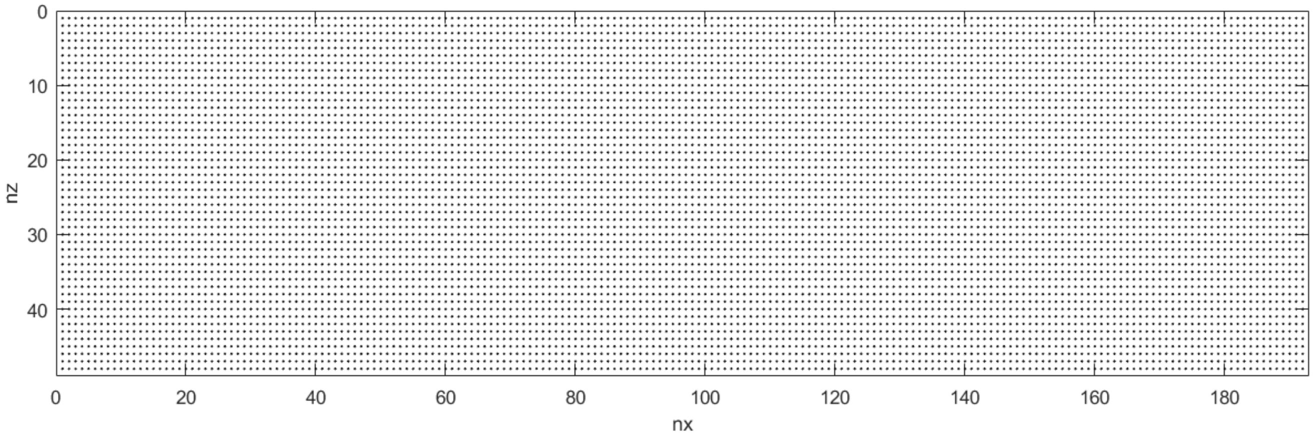


Figure 7.3: Modelling grid for the model of Figure 7.2.

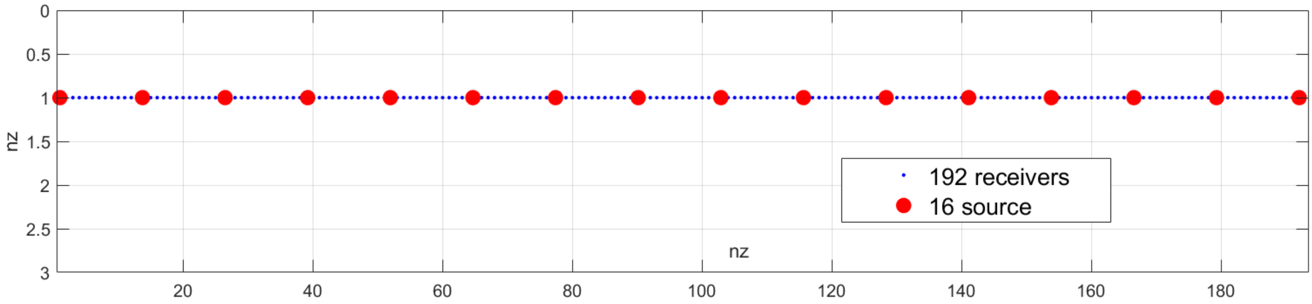


Figure 7.4: Source-receiver's layout of the seismic acquisition on the modelling grid of Figure 7.3.

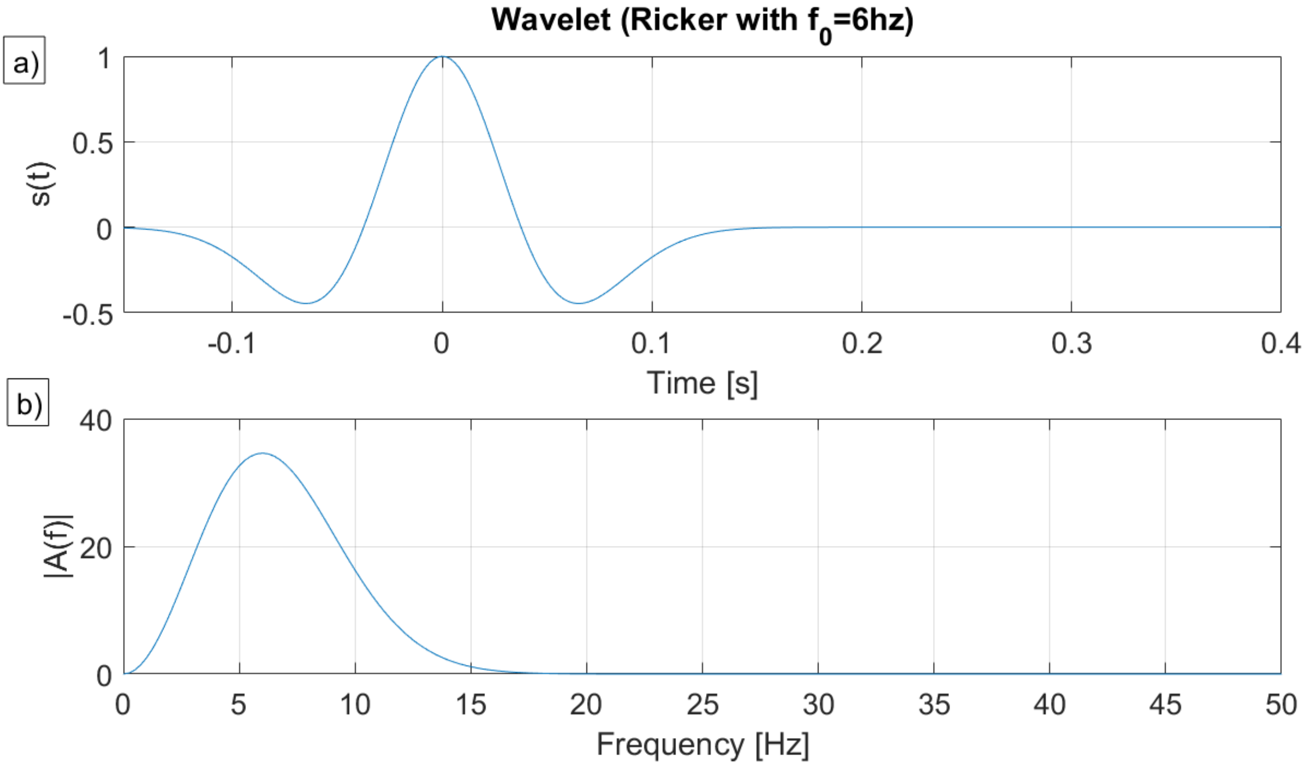


Figure 7.5: Ricker wavelet as a function of a) time and b) temporal frequency.

The computation of the seismograms is done with the software *fullwave2D*. To simulate a possible processing sequence on the data, I have applied:

1. a **low-pass filter** between 0 and 10 Hz to enforce the information of the low frequencies;
2. a **trace by trace normalization** to enforce the information of the seismograms at long offset.

Figure 7.6 shows the seismogram with the applied processing sequence related to the leftmost source. These seismograms are the set of the observed data we want to invert using a FWI procedure.

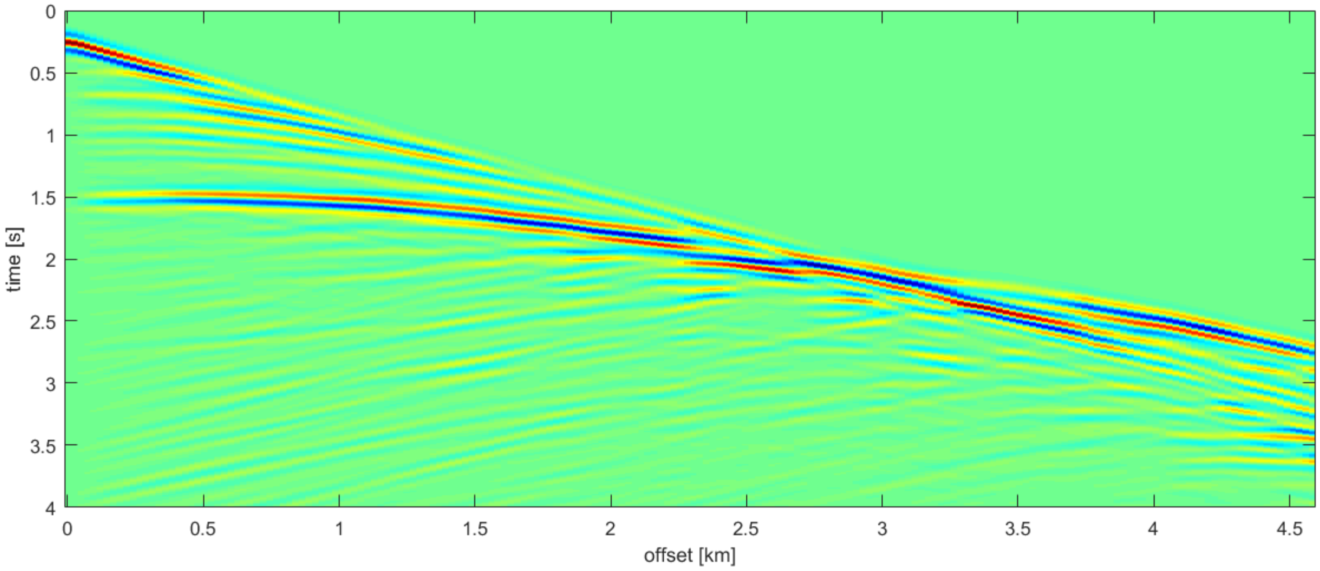


Figure 7.6: Observed seismogram related to the first source.

7.3 Defining the misfit function and the inversion grid

To apply a FWI procedure for inverting the observed seismograms, I had to set up:

- the **modelling algorithm** to simulate the synthetic seismograms;
- the **misfit function** to measure the difference between the predicted and the observed seismograms;
- the **inversion grid,** which is the set of the positions of the unknows;
- the **inversion procedure** used to estimate the models;

As modelling algorithm, I have used the software *fullwave2D* again with the same modelling parameters used to generate the observed seismograms. The computational time for each simulation corresponds to about 10s, using a Desktop PC with Intel Core i7-3770 Processor (4x 3.4 GHz) and 32 GB DDR4 RAM. As misfit function, the sum of the all the L^2-norm difference between the observed and the synthetic seismograms is used

$$F(v) = \sum_{s=1}^{16} \left(\frac{1}{2} \sum_{r=1}^{n_r} \left(\sum_{k=1}^{n_t} \left| G\big(u(v, t_k, \boldsymbol{x}^r, \boldsymbol{x}^s)\big) - G\big(u_0(\boldsymbol{x}, t_k, \boldsymbol{x}^r, \boldsymbol{x}^s)\big) \right|^2 \right) dt \right). \tag{7.1}$$

where:

- $\boldsymbol{x}^r$ are the positions of the receivers, the same for all the seismograms;
- $\boldsymbol{x}^s$ are the positions of the sources;
- t_k are the temporal moments of the seismograms;
- $u_0(\boldsymbol{x}, t_k, \boldsymbol{x}^r, \boldsymbol{x}^s)$ is the set of observed seismograms;
- $u(v, t_k, \boldsymbol{x}^r, \boldsymbol{x}^s)$ is the synthetic data;
- G is the processing operator, applied on the synthetic and observed seismograms.

Note that, using the software *fullwave2D,* I can evaluate the misfit function immediately after the computation of synthetic seismograms, without storing them.

As inversion grid, I have defined two different inversion grids:

- a **coarse inversion grid**, formed by 9 rows and 9 columns for a total number of 81 unknowns, if a global optimization algorithm is applied. In Figure 7.7 the set of the black points represents the coarse grid. To obtain the velocity model on the modelling grid from the coarse one, a bilinear interpolation function is used. The use of a coarse grid to have a feasible number of unknows for the application of the metaheuristic algorithms of Chapter 5 and Chapter 6;
- a **fine inversion grid**, equal to the modelling grid of Figure 7.3, without the first two rows. In Figure 7.7 the set of red points represents the fine inversion grid. This grid is used when a local algorithm is applied, for a total number of 8832 unknows.

Finally, as inversion procedure, I apply a two-step approach:

- in the first step, I use the two algorithms ASA and GA, as global optimization procedures to estimate two smooth models on the coarse grid;
- in the second step, I use the best models obtained by ASA and GA, as starting models for two local optimization procedures on the fine inversion grid.

Note that the water layer velocity and the water depth are supposed to be known and fix during both the global and local inversion procedures.

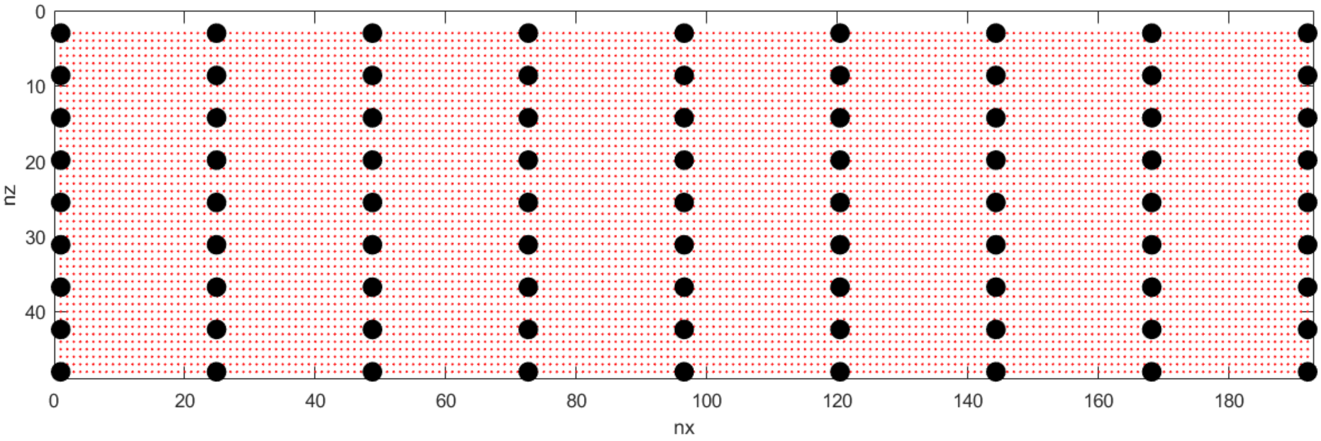


Figure 7.7: Fine inversion grid (red points) and coarse inversion grid (black points).

7.4 Global FWI using ASA and GA

In this section, I show the results obtained by the application of ASA and GA to this synthetic example. The ASA and GA parameters are set using the value of *Table 6.2*, *Table 6.3*, with $n = 81$.
The search ranges (Figure 7.8) of the 81 unknows varies from 1500 m/s to 2250 m/s for the unknowns at the first row of the coarse grid and from 1800 m/s to 3700 m/s for the unknowns at last row. We perform:

- 100000 iterations of the ASA algorithm, that correspond to generate a sequence $\{v_k\}_{k\geq 0}$ of 100000 models, a new model for each iteration k. At the end of the procedure, we consider as **best ASA model**, the model with the lowest misfit value found by the algorithm. Note that the best model could be different from the last model obtained by the algorithm.
- 150 generations of the GA algorithm, that correspond to generate an initial population of 810 individuals and 97200 offsprings, 648 for each generation. At the end of the procedure we consider as **best GA model**, the model with the lowest misfit value;

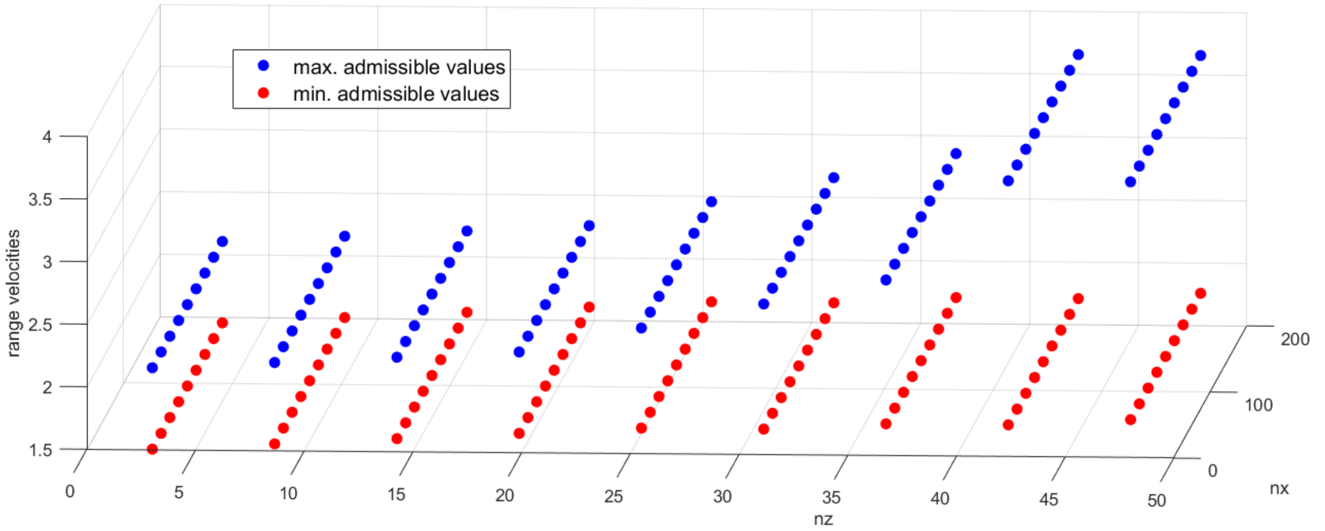


Figure 7.8: Search domain consisting of the search range, that varies from 1500 m/s to 2250 m/s for the unknowns of first row of the coarse grid and from 1800 m/s to 3700 m/s for the unknowns at last row.

To compare the performances of the two optimization algorithms, I consider the evolution of their data misfit during the optimization process. In particular, I consider:

- the **ASA data misfit**, a curve that represents the value of misfit function as a function of the total number of evaluated models k;
- the **GA data misfit**, a curve that represents the minimum value of the misfit, as a function of the total number of evaluated models, that is $810 + 648 * gen$, at each generation gen.

Figure 7.9 displays the evolution of the data misfit, obtained by the two algorithms. The ASA algorithm passes from an initial value of 2716 to a final value of 1265, but the minimum value is reached at $k = 93930$ and is 1238. The GA algorithm passes from an initial value of 2301 to a final value of 1392. Therefore, the lowest misfit value is the one obtained from ASA.

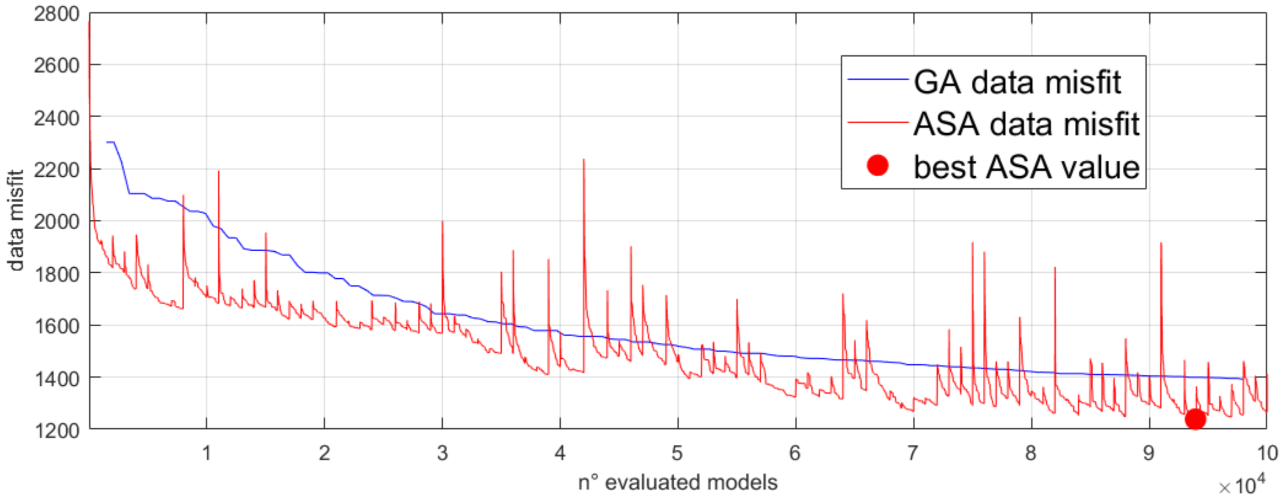


Figure 7.9: Evolution of the data misfit for ASA (red curve) and GA (blue curve).

Figure 7.10 displays the best ASA model (Figure 7.10a), the best GA model (Figure 7.10b) and their difference (Figure 7.10c) on the coarse grid. The ASA and GA best models are more similar to the top and the center of the coarse grid, whereas they present some significant differences especially on the bottom part. Figure 7.11 displays the same models on the modelling grid. Note the smoothness of the models caused by the interpolation operation between the two grids.

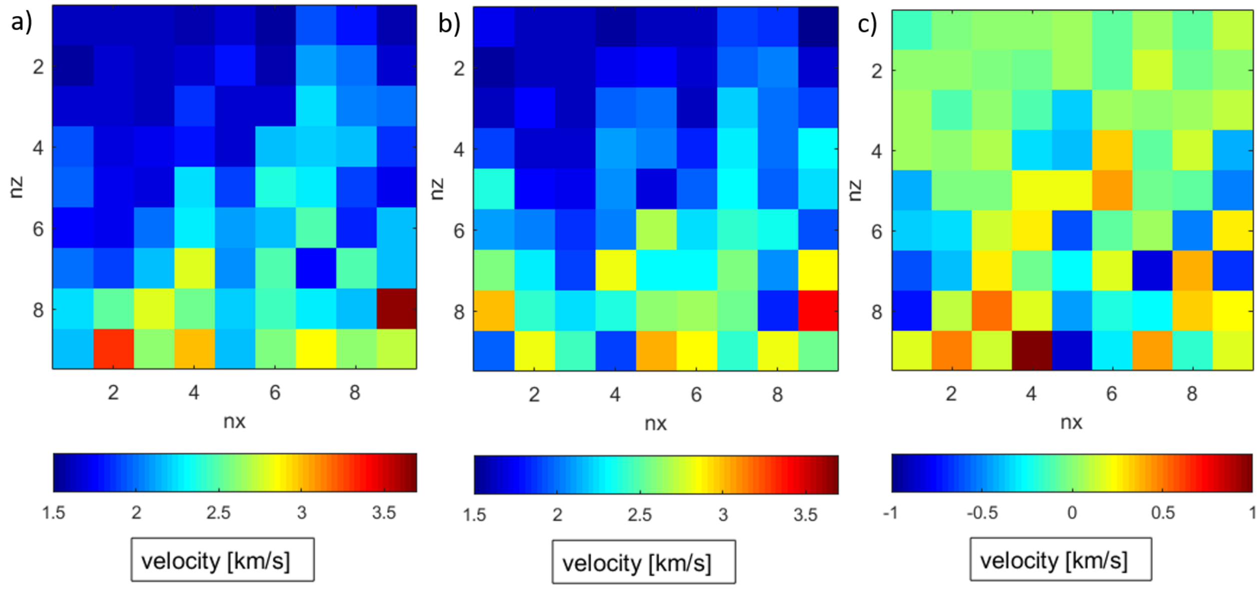


Figure 7.10: a) Best ASA model, b) best GA model, and c) their differences on the coarse grid.

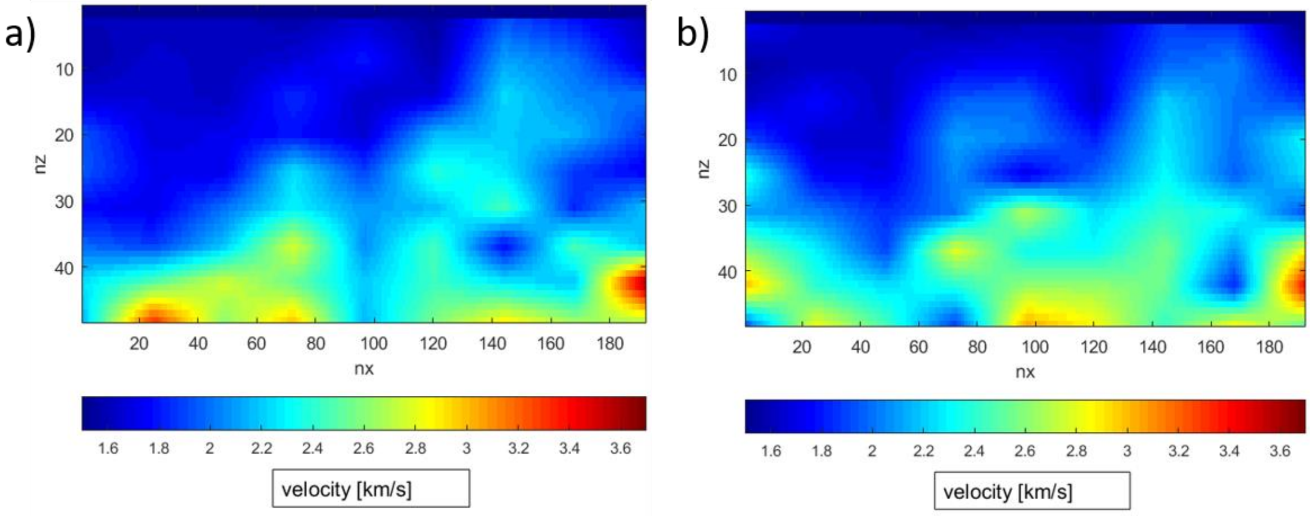


Figure 7.11: a) Best ASA model and b) best GA model on the modelling grid.

Finally, Figure 7.12 and Figure 7.13 show the differences between the predicted and the observed seismogram relative to the leftmost source in the acquisition layout. In particular, Figure 7.12 shows the differences between the observed and predicted seismograms, using the model of the first iteration of ASA (Figure 7.12a), and the best model (Figure 7.12b), respectively. Figure 7.13 displays the difference between the observed and predicted seismograms, using the best model of the first generation of GA (Figure 7.13a) and of the last generation (Figure 7.13b), respectively. Note the partial match of some of the main events on the seismograms, such as the main reflections and refractions, for both the synthetic seismograms obtained by the two algorithms.

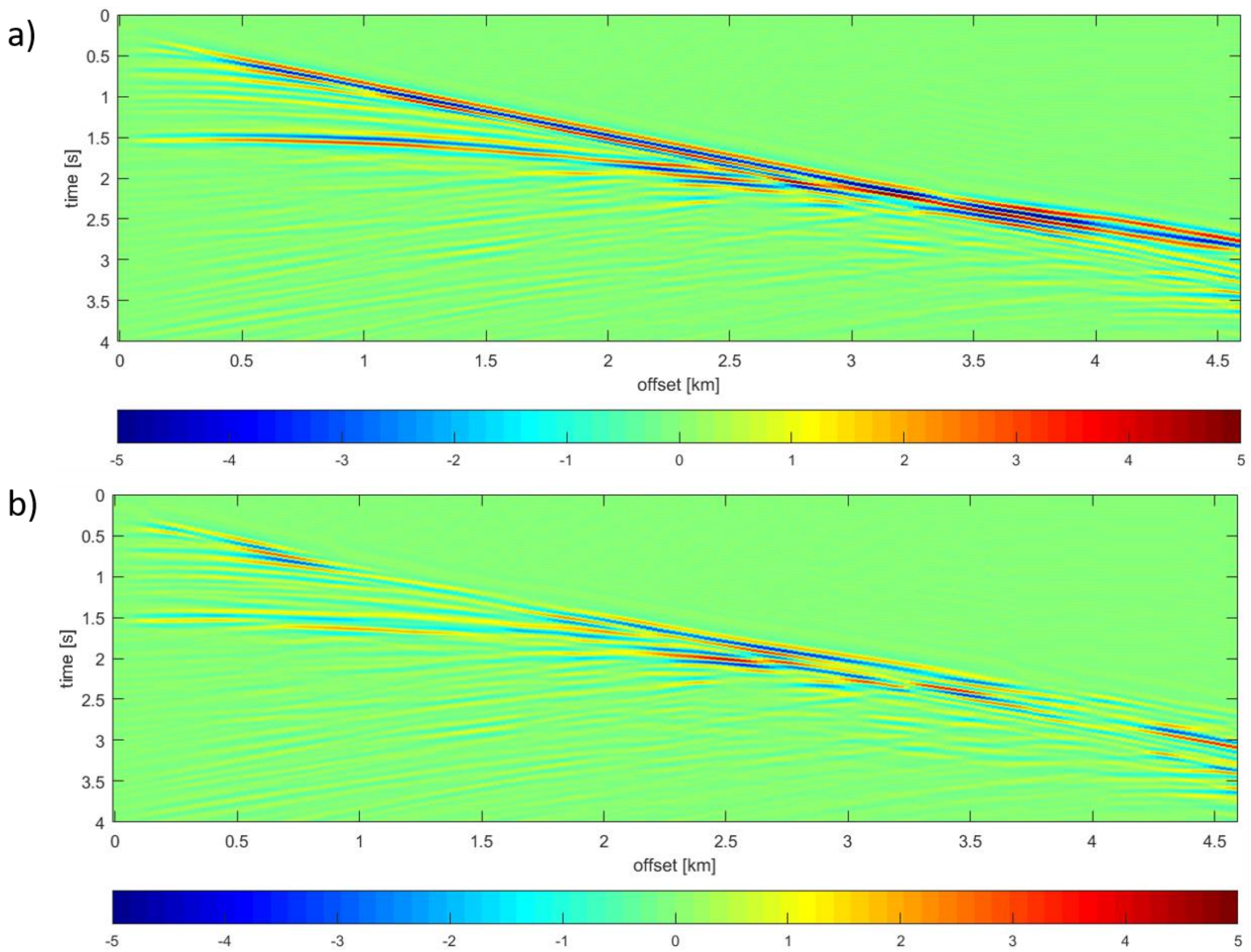


Figure 7.12: Differences between the predicted and the observed seismogram relative to the leftmost source in the acquisition layout, using a) the model of the first iteration of ASA, and b) the best model obtained by ASA.

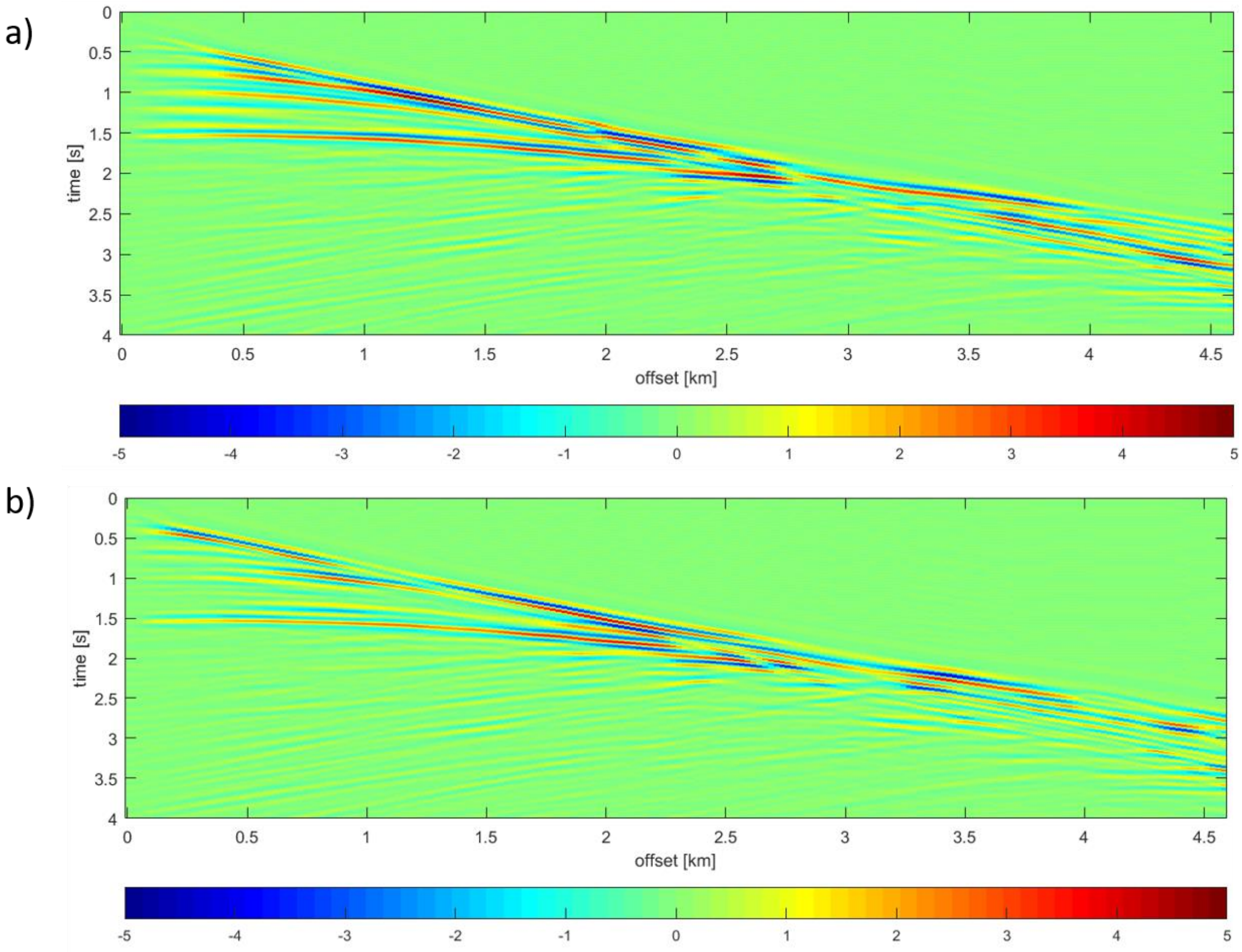


Figure 7.13: Differences between the predicted and the observed seismogram relative to the leftmost source in the acquisition layout, using a) the model of the first generation of GA, and b) the best model obtained by GA.

7.5 Local FWI starting from the best global model

We used the two best ASA and GA models, as described in the previous section, as starting model on the fine inversion grid for two different local optimization procedures. As local algorithm, I make use of the conjugate gradient method [66], where the descend direction $\mathrm{h}_{\mathrm{k+1}}$ of (1.4) is given by

$$h_{k+1} = -\nabla_{v_p} F\left(v_p{}^k\right) + \beta_k h_k, \tag{7.2}$$

and the parameter $\beta_k \geq 0$ is the Hestenes-Stiefel parameter [67], given by:

$$\beta_k = \frac{\left(\nabla_{v_p} F\left(v_p{}^k\right) - \nabla_{v_p} F\left(v_p{}^{k-1}\right)\right)^t \cdot \nabla_{v_p} F\left(v_p{}^k\right)}{\left(\nabla_{v_p} F\left(v_p{}^k\right) - \nabla_{v_p} F\left(v_p{}^{k-1}\right)\right)^t \cdot h_k}. \tag{7.3}$$

For the step length γ_k of Equation (1.4), we estimate a value $\widehat{\gamma_k}$ along h_k, that satisfies the Wolfe conditions of Equation (1.6), using as trial **initial step length**

$$\alpha = \hat{\gamma}_{k-1} \frac{\left(\nabla_{v_p} F\left(v_p{}^{k-1}\right)^t \cdot h_{k-1}\right)}{\nabla_{v_p} F\left(v_p{}^k\right)^t \cdot h_k}. \tag{7.4}$$

and we reduce the step length by half until the Wolfe conditions are satisfied, with $c_1 = 0.0001$ and $c_2 = 0.1$, as suggested in [11]. To compute the gradient $\nabla_{v_p} F\left({v_p}^k\right)$ of the misfit function in (7.2), the implementation of the adjoint method described in Chapter 3 is used.

The search domain is displayed in Figure 7.14 and consists of the search ranges of the 8832 unknows, that varies from 1400 m/s to 2350 m/s for the unknowns of the first row of the fine grid and from 1700 m/s to 3800 m/s for the unknowns at last row.

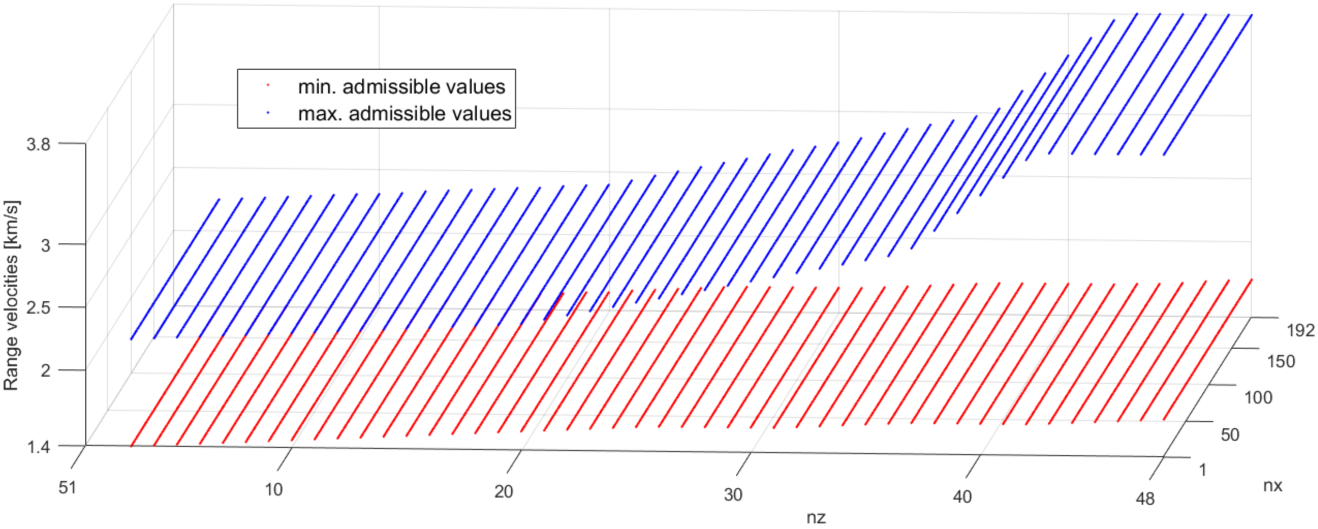


Figure 7.14: Search domain consisting of the search range, that varies from 1400 m/s to 2350 m/s for the unknowns of first row of the fine inversion grid and from 1700 m/s to 3800 m/s, for the unknowns at last row.

200 iterations of both the minimization procedures are performed. Figure 7.15 displays the two curves of the evolution of the data misfit: the blue curve represents the minimization started from the best GA model, whereas the red one represents the minimization started from the best ASA model. Note a decrease of about 98 % for the GA data misfit at the last iteration, passing from 1392 to 27.4, and a decrease of about 98.1 % for the ASA data misfit at the last iteration, passing from 1238 to 24.5.

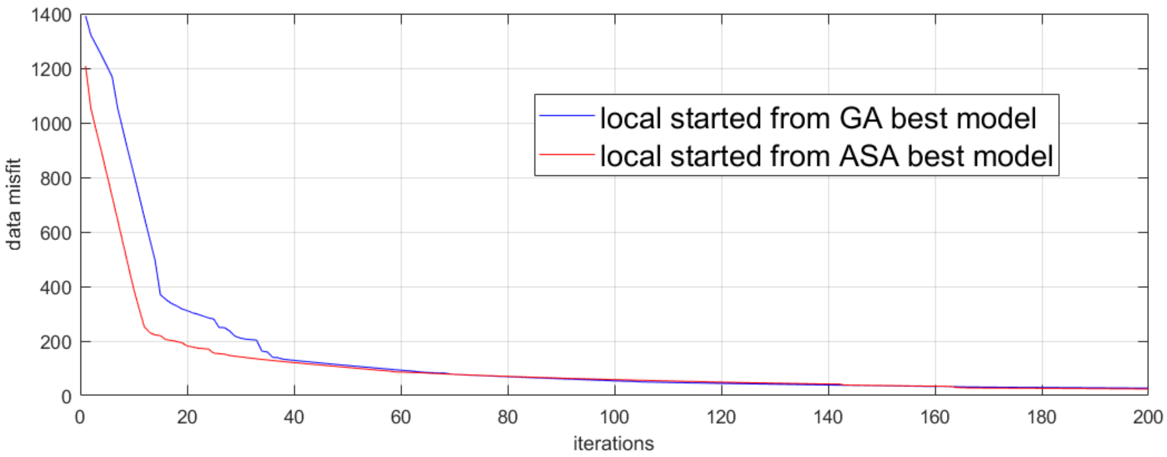


Figure 7.15: Evolution of data misfit for the two local minimization procedures on the fine inversion grid: the red curve starts from the best ASA model; the blue curve starts from the best GA model.

Starting from the GA best model, Figure 7.16a and Figure 7.16b shows the final model and the difference between the true and the final model obtained at the end of the minimization procedure, respectively. Starting from the ASA best model, Figure 7.17a and Figure 7.17b shows the final model and the difference between the true and the final model obtained at the end of the minimization procedure, respectively

The final models obtained by the two minimizations are very similar to the true one, except in some areas near the lateral and the bottom boundaries, where the seismic illumination is poor. This fact means that the starting models estimated by the two algorithms can be considered quite near the basin of attraction of the global minimum of the misfit function, that corresponds to the true model.

As a further validation of the effectiveness of the two-step procedure described, in Figure 7.18 and Figure 7.19 I show the difference between the predicted and the observed seismograms relative to the leftmost source in the acquisition layout at the end of the local FWI, using the best GA and ASA model as starting model, respectively. Note the significant reduction of the difference between the observed data and the final predicted one.

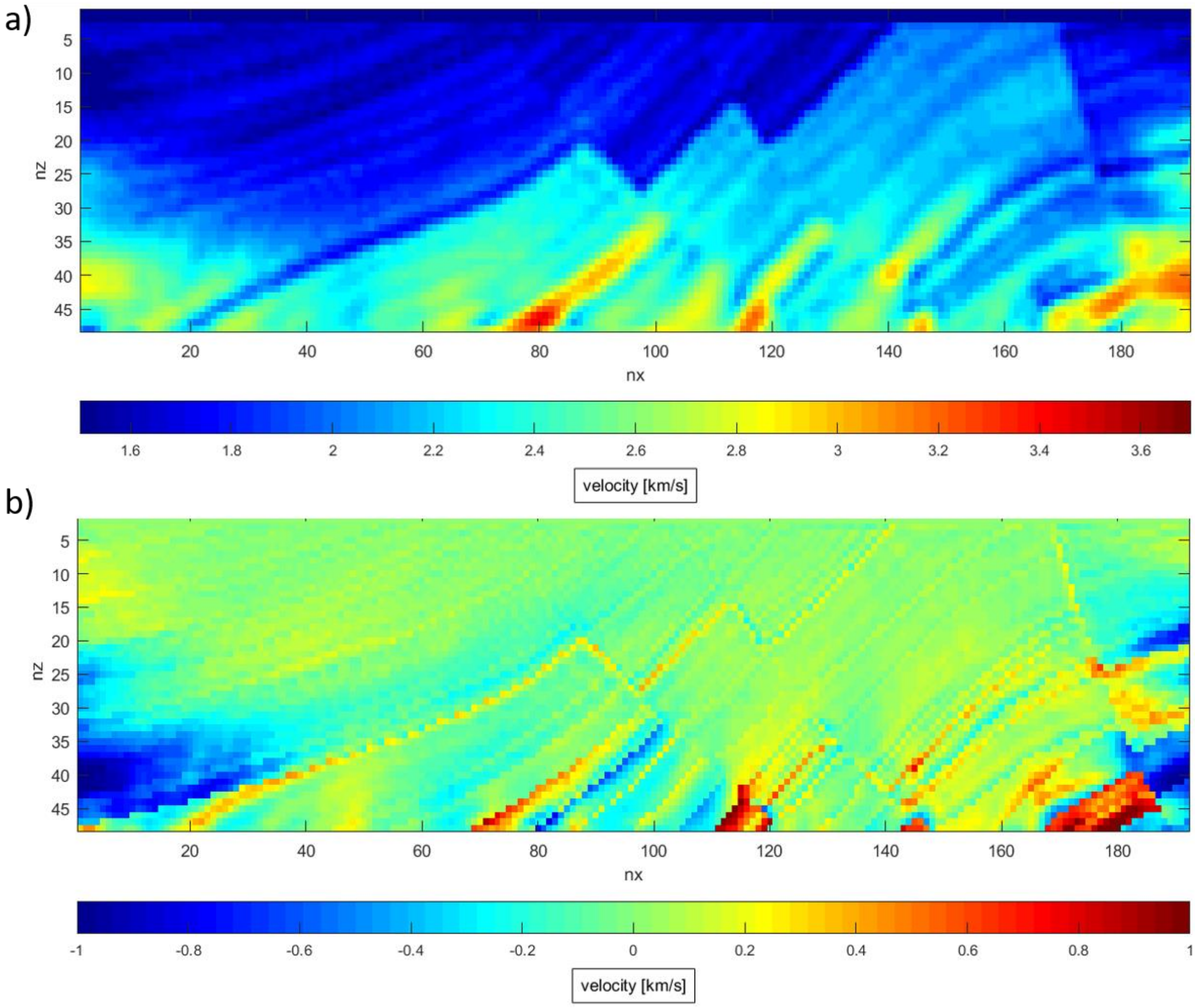


Figure 7.16: a) The final model obtained by the minimization procedure, using the GA model as starting model, and b) the difference between the true and the final model.

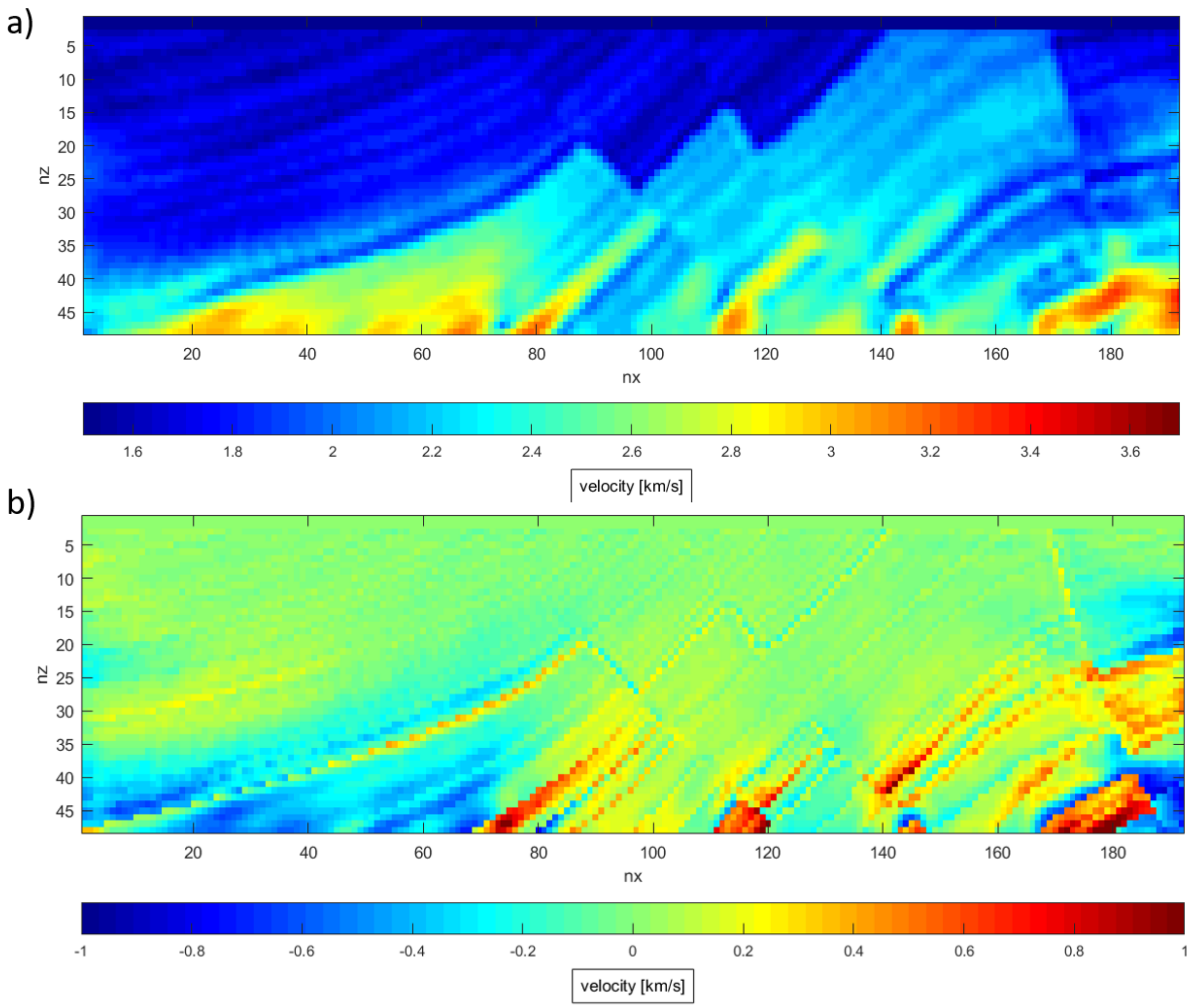


Figure 7.17: a) The final model obtained by the minimization procedure, using the ASA model as starting model, and b) the difference between the true and the final model.

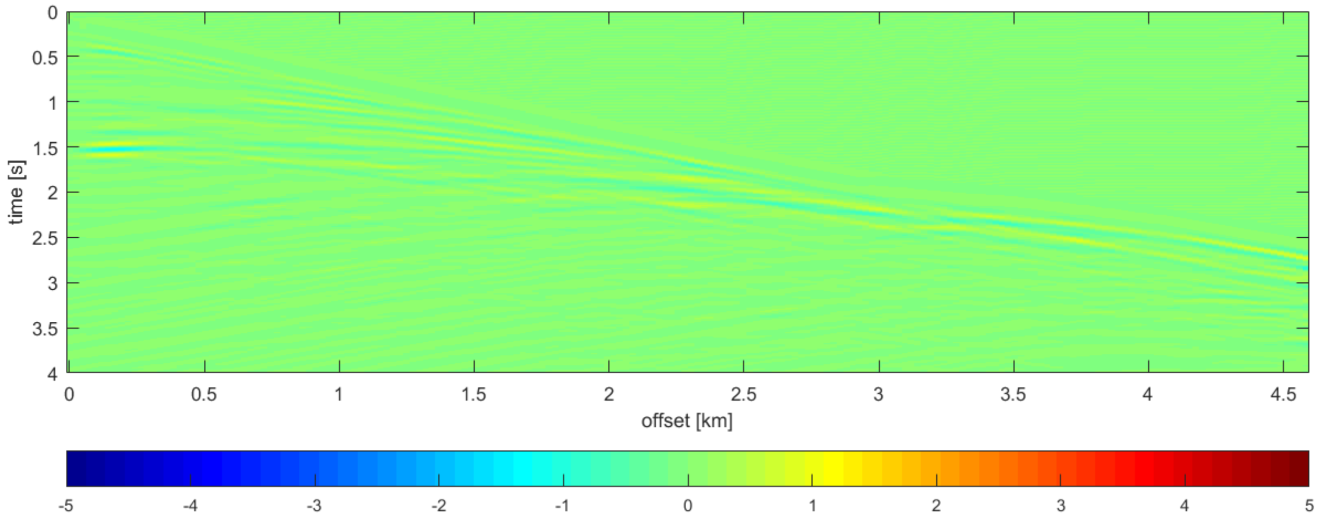


Figure 7.18: Difference between the predicted and the observed seismogram relative to the leftmost source in the acquisition layout, using the best model obtained at the end of the local FWI, starting from the best GA model.

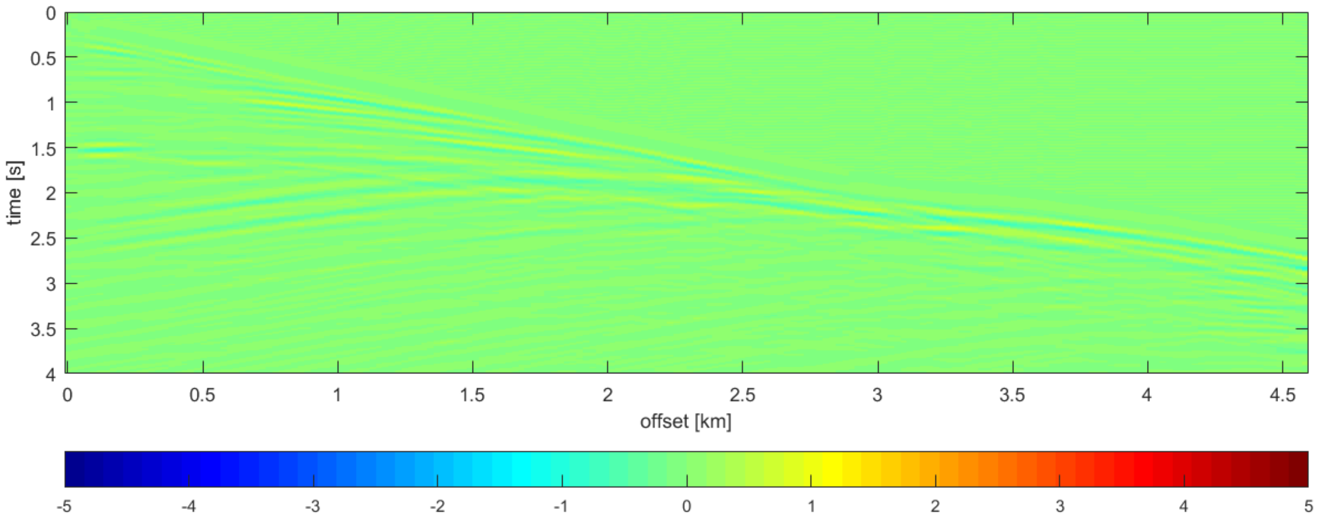


Figure 7.19: Difference between the predicted and the observed seismogram relative to the leftmost source in the acquisition layout, using the best model obtained at the end of the local FWI, starting from the best ASA model.

7.6 Conclusions and results

Using a global optimization algorithm such as ASA or GA on a coarse inversion grid, it is possible to estimate a proper starting velocity model for a local FWI, which allows us to solve a complex synthetic seismic inversion problem. The good correspondence between the true model and the final model, and between the predicted and observed seismograms, makes the application of the global optimization algorithms of particular interest, especially in seismic inversion problems characterized by highly non-linearity and multiple local minima.

Chapter 8.
Experience of FWI on real land data

In this chapter, I illustrate the results of an experience of acoustic FWI on a 2D seismic line acquired onshore, in South Tuscany, aimed at estimating a low-wavenumber P-wave velocity model of the subsurface. To obtain this, I use a similar procedure of the previous Chapter. First, I use a global optimization procedure on a coarse inversion grid to obtain a smooth velocity model of the area of interest. Then I use this model as a starting model on a fine inversion grid for a subsequent local optimization.

The synthetic seismograms are computed using the *fullwave2D* code and require a high computational time. Because of this, I make use of only of the GA algorithm as global optimization procedure, which allows us to parallelize the computation of the individuals of a generation, reducing in this way the computational time. See [68] and [69] for more details.

8.1 Location and geological framework

The seismic data are acquired in Southern Tuscany, that is characterised by:

- an average altitude of 270 meters, with positive Bouguer anomalies [70];
- high heat flow: $120\ mW/m^2$ on the average, with local peaks up to $1000\ mW/m^2$;
- the crust-mantle passage at a depth of around $22-25km$ [71];
- a lithosphere-asthenosphere boundary believed to be at about 30 km [72];
- two important geothermal systems that are the Larderello (north) and the Monte Amiata (south) geothermal fields.

The structural and stratigraphic setting of this area is the consequence of two different deformational processes:

- the first is related to a process of **convergence tectonic** [73], in which the European margin and the Adria microplate, during the period of Cretaceous-Early Miocene, collided and produced the stacking of the Northern Apennines nappes;
- the second is related to a process of **post-collisional extensional tectonic** [74], which have affected the inner zone of the Northern Apennines, starting from the Early-Middle Miocene; this process caused the opening of the northern Tyrrhenian Basin, with the development of the northern and southern tectonic sectors in Tuscany, which are separated by a SW-NE trending lineament of lithospheric significance.

In particular, in the southern sector, the extensional tectonic have caused the emplacement of Neogene-Quaternary magmatic bodies, mainly derived from mixing of crust and mantle magmas. For more details and references about this area, the reader can see [75].

8.2 Data

In this section, I present the seismic data used to test the acoustic FWI. These data are taken from a 2D land seismic profile, acquired in 1995, during the CROP project.

8.2.1 The CROP project

The CROP project (CROsta Profonda) was an Italian deep crust exploration project, that is a result of a collaboration between the National Research Council (CNR), ENI-AGIP and ENEL. It represents a multidisciplinary research program, with the following targets:

- understanding the main geodynamic processes responsible for the current geological arrangement of Italy;
- define and prevent geological hazard;
- search for energy resources, especially hydrocarbons and geothermal fields;
- identification of stable zones for the industrial installation.

The CROP Project started in the middle of '80 as a feasibility study. Then from 1986 to 1999, about $10000\ km$ of land and sea seismic reflection profiles were acquired, processed and interpreted. Figure 8.1 shows the location map of all the seismic profiles.

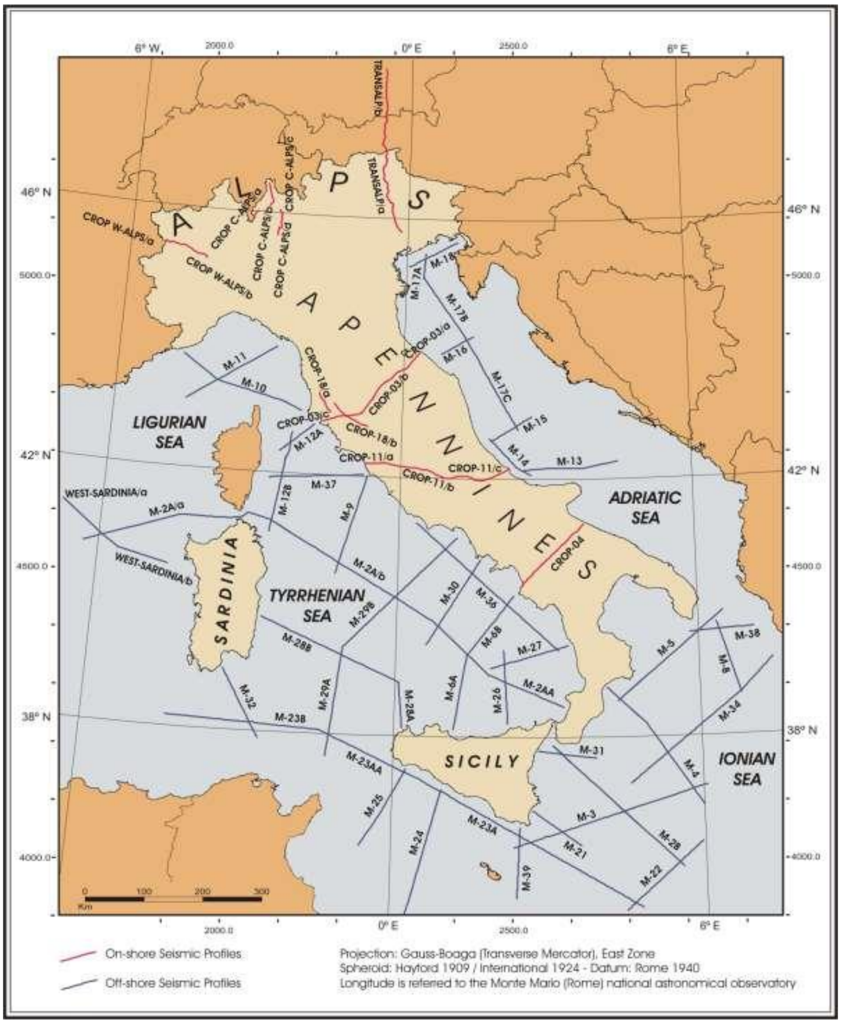


Figure 8.1: Map of all the seismic profiles of the CROP project.

The marine profiles have a total length of about $8740\ km$ and were acquired in shallow and deep waters around the Italian peninsula, in both compressional and extensional structural domains, and in the presence of sea floors of variable morphology and acoustic characteristics. During the first phase (CROP Mare 1, 1991) about $3400\ km$ of seismic profiles were recorded in the Ligurian, Tyrrhenian, and Ionian

Seas. Then, in a second phase (CROP Mare 2, 1993-1994), over 5000 km of seismic profiles were acquired in the southern Tyrrhenian Sea, in the Sardinia Channel and the Ionian and Adriatic Seas. In both phases, the data were acquired by the oceanographic vessel OGS-Explora, using air-gun sources. Data processing were performed by OGS, AGIP and ISMES.

The land profiles consist of about 1250 km and were acquired in very different areas such as the Alps (CROP01/TRANSALP), the Northern Apennines (CROP03, CROP18), the central Apennines (CROP11) and the Southern Apennines (CROP04). The seismic source was the vibroseis (CROP01, CROP04) or the dynamite (CROP03, CROP11, CROP18). Table 8.1 and Table 8.2 illustrate the main characteristics and acquisition parameters of the land and marine profiles, respectively [76].

	CROP 01	CROP 03	CROP 04	CROP 11	CROP 18
Year	1999-2000	1992-1993	1989-1990	1995,1999	1995
N° of profiles	1	1	1	1	2
Area	Eastern Alps	Northern Appenines	Southern Appenines	Central Appenines	Southern Appenines
Total length	300km	230km	172km	265km	116km
Recording time	18-20s	25s	18-20s	25s	25s
Sampling interval	2ms	2ms	2ms	2ms	2ms
Station interval	50m	60m	80m	40-60m	60m
Source interval	4500m	180m	2400m	160-280m	180m
N° of channels	360	192	240	192	192
Type of source	vibroseis	dynamite	vibroseis	dynamite	dynamite

Table 8.1: Main characteristics of the land profiles of the CROP project.

	CROP-ECORS	CROP MARE 1	CROP MARE 2
Year	1988	1991	1993-1995
Area	Western Sardinia	Ligurian, Tyrrhenian and Ionian Sea	Tyrrhenian, Ionian and Adriatic Sea and channel of Sicily
Total length	205km	3410km	5225km
Recording time	16s	17-20s	17-30s
Sampling interval	2ms	2ms	2ms
Hydrophone interval	25m	25m	25m
Streamer length	3000m	4500m	4500m
N° of channels	120	180	180
Type of source	Air-gun	Air-gin	Air-gun

Table 8.2: Main characteristics of the marine profiles of the CROP project.

The digital and analog data have been stored in a Data Center created at the Bologna Section of Institute of Marine Sciences ISMAR-CNR, to create an official CROP Data Base, becoming operative during 2004.

Even if the first phase of elaboration of seismic data was already done, nowadays it is possible to improve the results using more modern and sophisticated techniques, among which FWI. In the last decade, a significant effort has been made to study the geological interpretation, the seismic signals, the correlations with other geophysical data and the re-processing of seismic data. See, e.g., [77], [78], and the references therein.

In this thesis, I used one of the seismic profile, called CROP18, to perform a 2D land FWI.

8.2.2 The CROP18A seismic line

The reflection crustal seismic line CROP18 was acquired in 1995 to investigate the relationship between the geological structures and geothermal resources of southern Tuscany. Figure 8.2 shows a simplified map of Southern Tuscany with the location of the profile.

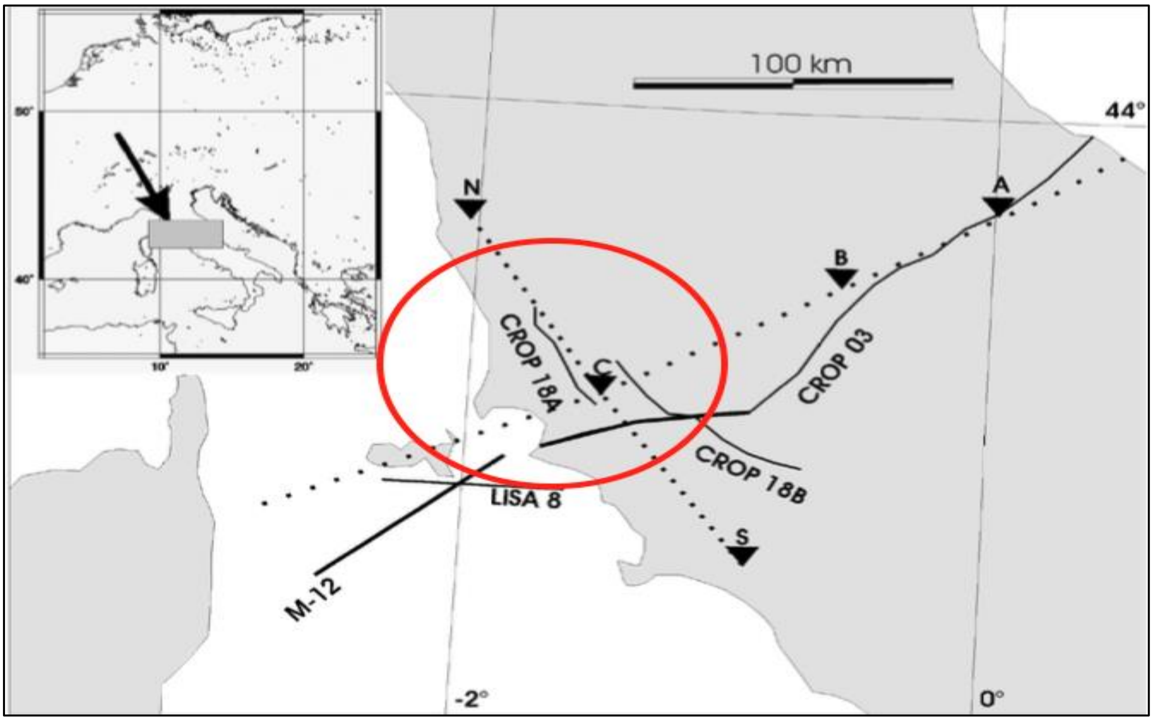


Figure 8.2: Location of the CROP 18 profile, crossing the CROP03 profile.

The line is 115km long, and NNW-SSE oriented from S. Giovanni delle Contee to Guardistallo and it consists of two parts: the first is called CROP18a and is in the northern part of the Larderello geothermal field, the second is called CROP18b and is in the southern part, over the Amiata area. The seismic line crosses another seismic line, the CROP03, that was previously acquired in 1993, and extends across the whole Central Italy, from Punta Ala (on the Tyrrhenian coast) to Gabicce (near Pesaro, on the Adriatic coast) (Figure 8.2).

The total length of the CROP18a segment is about 48km and consists of 195 shots. The shot interval is 180m. Figure 8.3a shows the position of shots along the profile as a function of latitude, longitude, and elevation, whereas Figure 8.3b shows the positions of the receivers related to the first shot. Note that the maximum topography variation is about 450m.

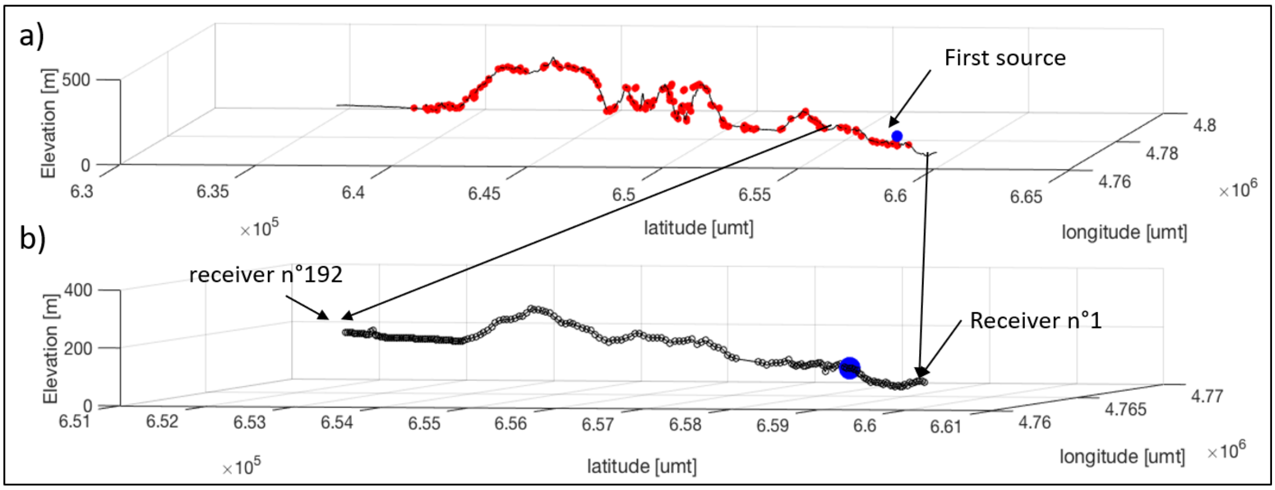


Figure 8.3: a) Positions of the shots (red points) along the profile and b) Positions of the receivers (black points) relative to the first source.

The data were acquired using an explosive source of about $30kg$. The depth of the borehole is between $15m$ and $30m$. The receiver system consists of an array of 24 geophones with a natural frequency of $10Hz$. The spread is asymmetric and formed by 192 active channels, with a station interval of $60m$. The minimum offset is between $150m$ and $300m$, while the maximum offset is between 6.6 and 8.4 km. The total number of seismic traces is 37824. The time sampling is $2ms$, while the recording time is $25s$.

8.2.3 Preliminary comments about the seismic line

Before any processing, inversion or interpretation operation on the seismic profile, it is necessary to display the raw data to have an idea of the possible characteristics or difficulties we must face, such as the signal-to-noise ratio (S/N), the temporal frequency range or seismic velocity of first arrivals. Figure 8.4 shows four different shots as a function of the positions of the sources and the receivers along the profile.

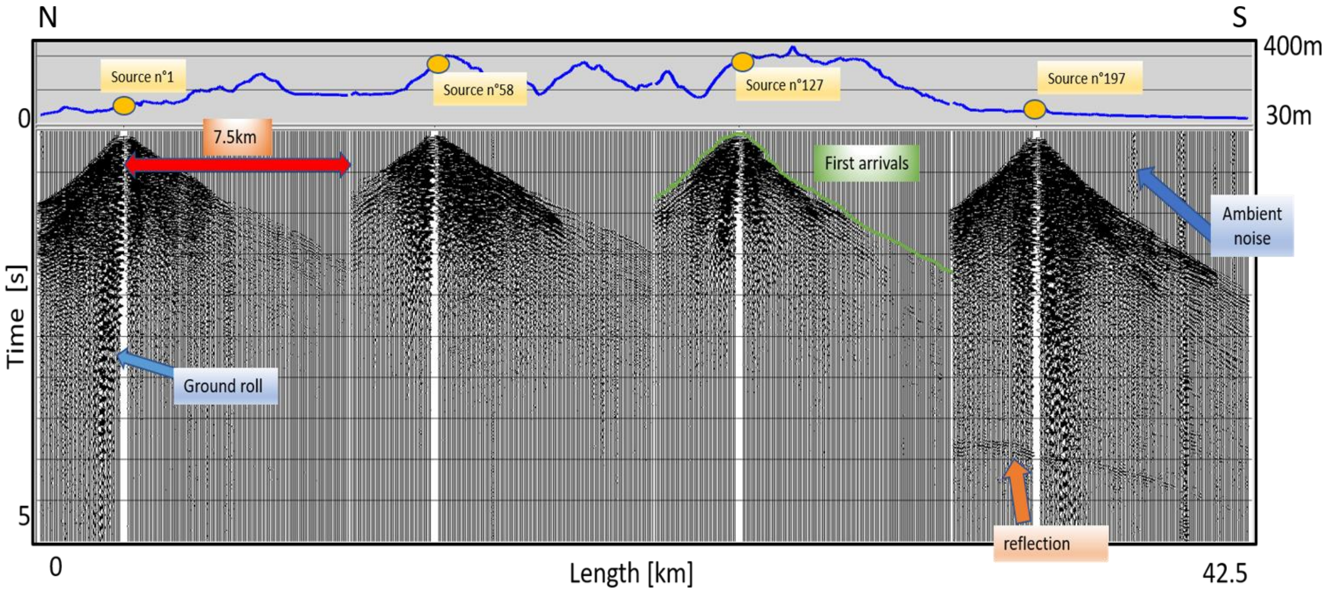


Figure 8.4: Four different shots as a function of the sources and the positions of the receivers along the profile.

At first sight, it is possible to note a low S/N ratio, caused by the presence of the surface wave (ground roll) at near offset and by the random noise at far offset. The apparent velocity of the first arrivals appears to be very high, between 3km/s to 7km/s, and with a general decrease moving from the first to the last shot. Despite the good visibility of the first arrivals, in general the CROP18a profile is characterized by a low presence of reflections. The main strong reflection is the so-called K-horizon, that is possible to be noted along all this profile for a registration time of about 4s, especially for the shot at the end of the profile (in the last shot of Figure 8.4 the K-horizon is clearly visible). This seismic event is also present on other seismic lines, such as the CROP03, and is nowadays subject to research about the geological interpretation. The seismic data are in SEG-Y file format, and the header of the file contains many information about the acquisition parameters of the entire profile.

8.2.4 Expanding spread experiments

Besides the shots, two wide-angle experiments, called **expanding-spread experiments** (ES) were acquired to integrate the shot information, with the aim to reach the deep targets with illumination angles wider than those used in the shot.

The idea is to perform a composite shot, where the charge size increases with the source-spread distance to preserve an appropriate signal-to-noise-ratio at far offset. The position of the first ES experiment is at the begin of the seismic profile, while the second one is at the end.

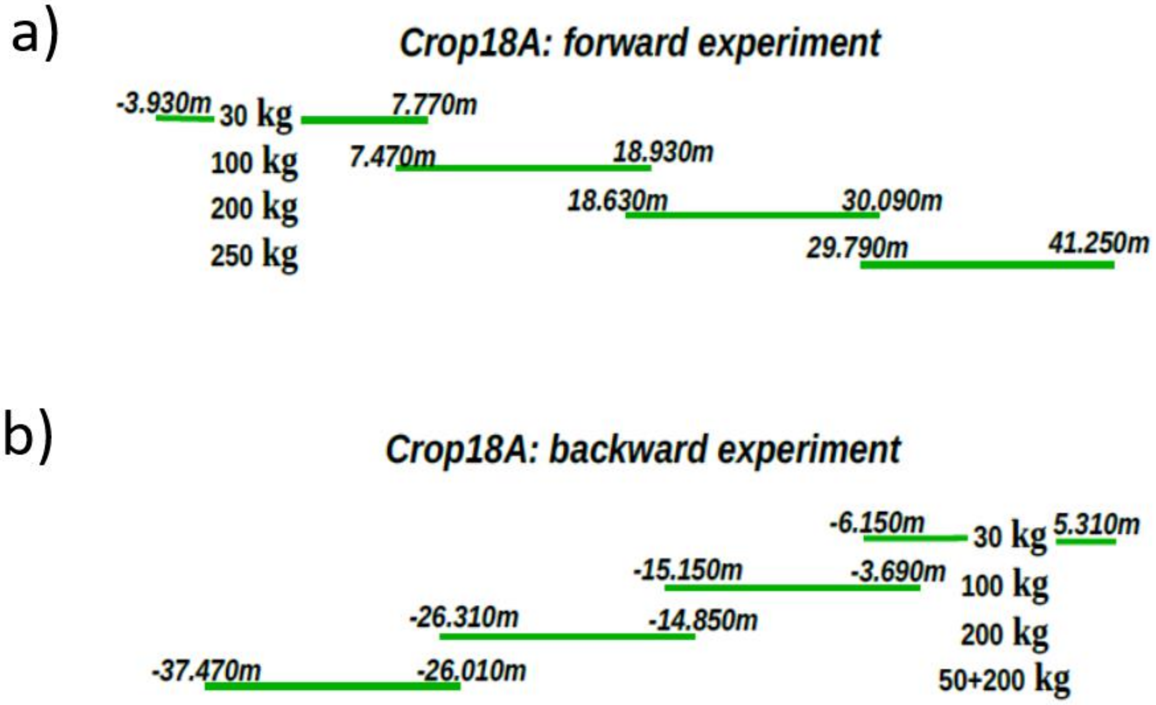


Figure 8.5: Design of the expanding spread experiments: a) forward experiment, b) backward experiment. The Figures are taken from [79].

Figure 8.5 shows the design of each experiment and the explosive charge used for each spread position. See [79] for more details. The first ES (Figure 8.5a), called **forward experiment**, is the result of an assembly of four different data acquisitions. Each of them is formed by 192 active channels and the position of the last eleven channels of the first, the second and the third segment overlaps to the first eleven channel of the second, the third and the fourth segment respectively. By this way, the forward experiment has a maximum offset of about 45km, and the total number of different active channels is 750. The seismogram of the forward ES is displayed in Figure 8.6a. The second ES (Figure 8.5b), called **backward experiment**, is the result of an assembly of four different segments. The position of the last eleven channels of the first and the second segment overlaps to the first eleven channels of the second and the third segment, while the position of the last 42 channels of the third segment overlaps the first 42 channels of the fourth segment. By this way, the backward experiment has a maximum offset of about 42km and the total number of active channels is 714. Each of the four segments is formed by 192 active channels. The seismogram of the backward ES is displayed in Figure 8.6b.

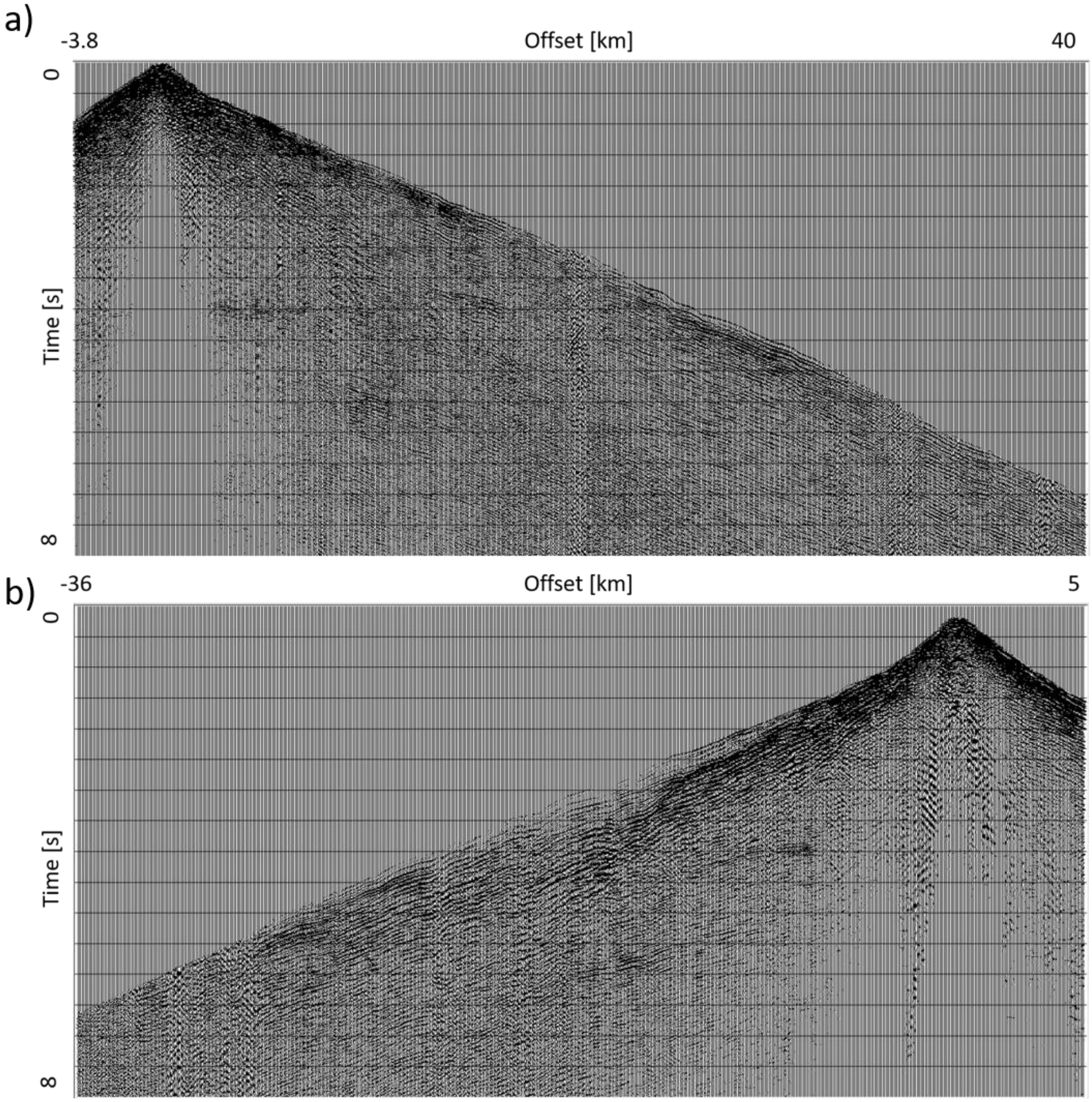


Figure 8.6: Seismograms of a) forward and b) backward expanding spread experiments.

8.2.5 The time section

A time stack section of the CROP18a profile was elaborated in 1996 and is displayed in Figure 8.7 until $T = 6s$. At first sight, it is possible to note the presence of strong low-frequency events in the upper part of the section (between $0s$ and $1s$), that could represent the shallow geological discontinuities of the subsurface. A strong reflection, so-called K-horizon, is located at about $4s$, between CDP 1180 and 1460 (the end of the seismic profile). Finally, a very dip reflection is visible between CDP 950 and 1150, deepening towards North.

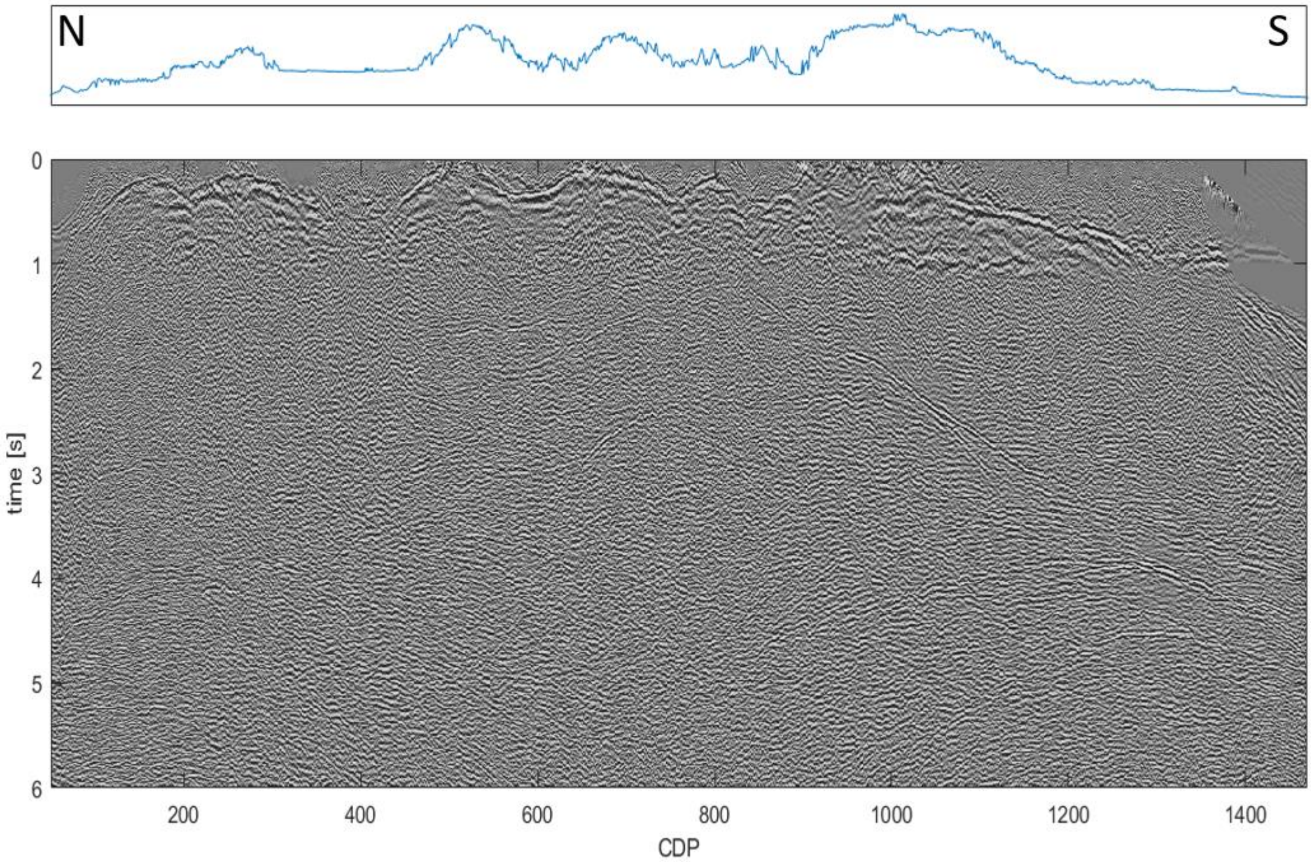


Figure 8.7: Time section for the CROP18a profile, taken from [77].

This section can be used for post-stack depth migration, if we have a reasonable acoustic velocity model of the subsurface, to map the events in the correct positions in depth. The estimation of a reasonable velocity model represents a challenge for the FWI process, mainly because of the low S/N ratio of the seismic data.

8.3 Setting of seismic data for 2D acoustic FWI

To reduce the computational cost of the FWI, I selected only a certain number of shots representative of the entire profile. I have chosen both the ES and 18 shots along the profile, for a total number of 20 shots. Figure 8.8 shows the positions of the sources of the selected shots.

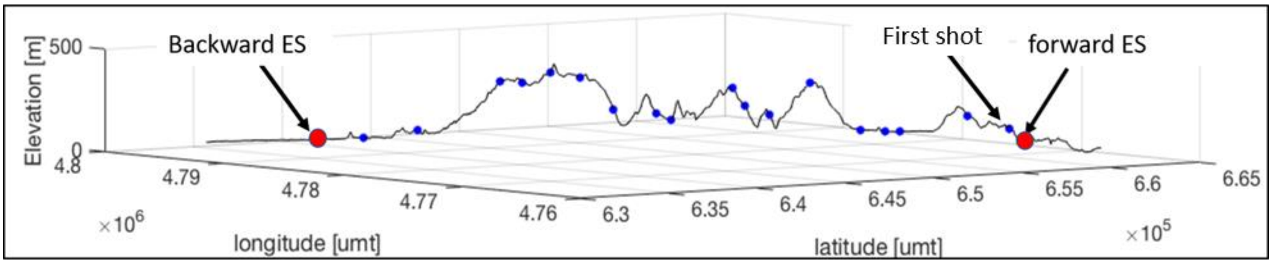


Figure 8.8: The 20 shots of the CROP 18a profile selected for the FWI. The blue points represent the positions of the shot, the red ones the positions of the ES.

The use of a 2D acoustic FWI requires a certain number of operations on the raw seismograms. I have divided the operations of the raw seismograms into two categories:

- **noise removal operations** that are the operations necessary to remove any noise or surface waves, performed only on the raw observed seismograms;
- **processing operations**, that are the set of all the possible operations applied to the observed and synthetic seismograms.

8.3.1 Noise removal operations

The set of operations applied to the raw seismograms consists of:

- **trace killing**, to remove the noisy traces;
- **top mute**, to remove all the noise before the first arrivals;
- **adjacent shot sum**, to reduce the random noise and enforce the coherent signals. In particular, each shot was summed with the two-adjacent shots on the left and the two ones on the right;
- **f-k filter**, to attenuate the ground roll effect at near offset. A polygonal filter was applied on the $F - K$ domain to attenuate all the seismic events with low frequencies between 1 and 14 Hz and low velocity between $0.5km/s$ and $1.7km/s$ for negative and positive wavenumbers;
- **predictive deconvolution**, to further reduce to the ground roll effect at near offset. It was applied using a predictive distance of $50ms$, an operator length of $250ms$ and an offset between $-650m$ and $650m$;
- $\boldsymbol{f - x}$ **deconvolution**, to remove the random noise at far offset. A temporal window of 300ms and a frequency range between 2 and 60 Hz were used;
- **dip-scan stack**, to strengthen the coherent signal of the seismic events over a specific velocity that is $3km/s$.

Figure 8.9 and Figure 8.10 show one of the shot and the backward ES experiment before and after the noise removal operations. As we can note, the ground roll effect was mitigated, and the coherent signals of the first arrivals and the reflections were enforced.

These processing operations were carried out using the Promax software of Landmark Graphics Corporation. See [80] for more details about these steps of data pre-processing.

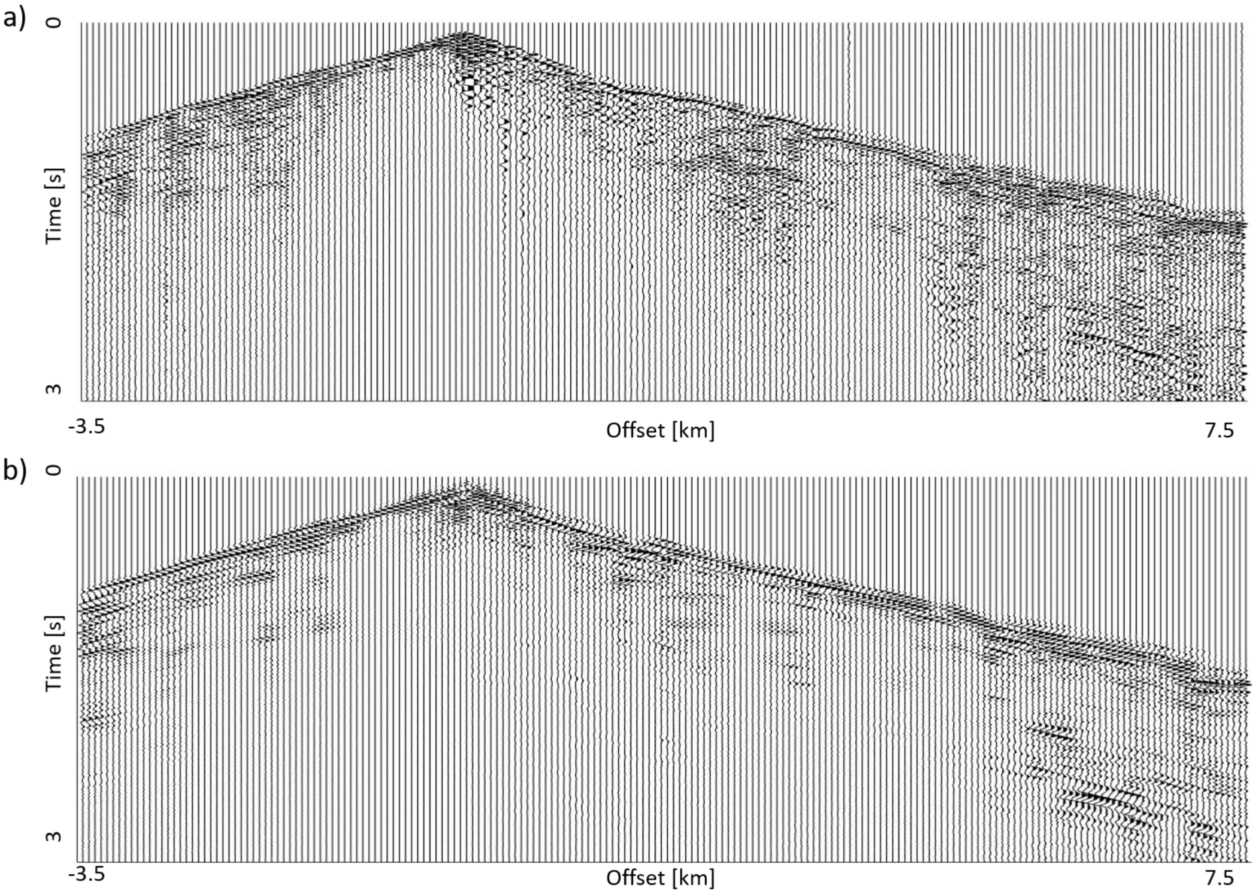


Figure 8.9: First shot selected for the FWI, a) before and b) after the noise removal operations. The seismogram is displayed until T=3s. To improve the visualization, a trace by trace normalization is applied.

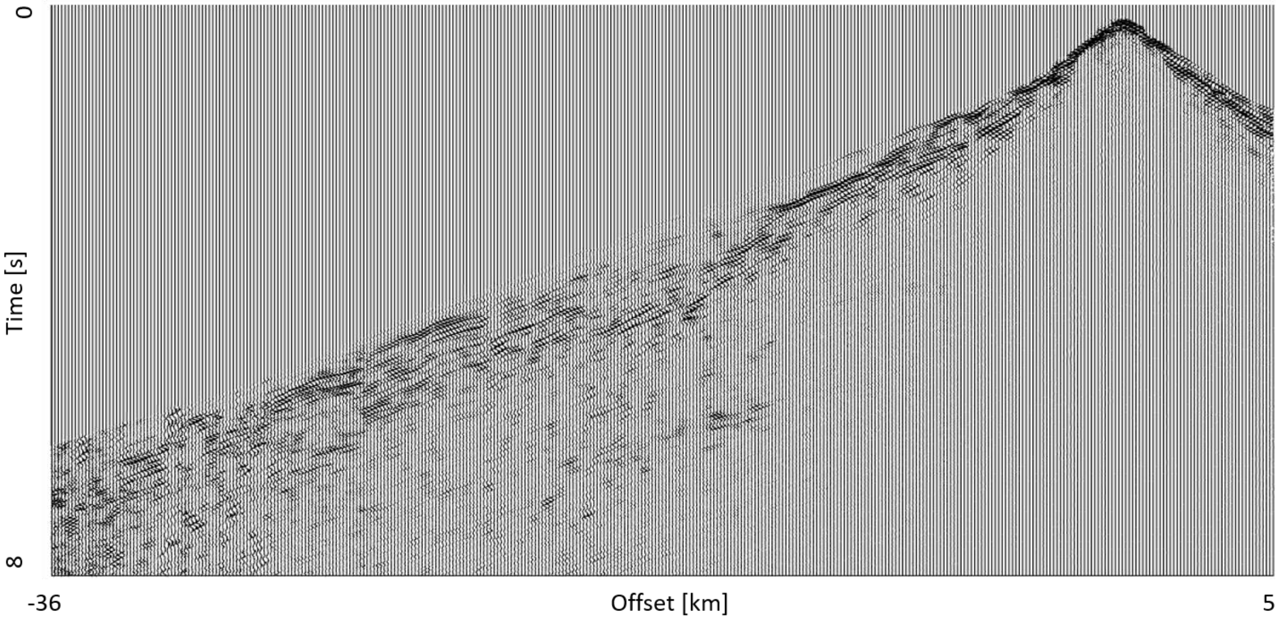


Figure 8.10: Backward ES experiment of Figure 8.6b, after the noise removal operations. To improve the visualization, a trace by trace normalization is applied. Note the different offset of the ES in comparison to the shot of Figure 8.9.

8.3.2 Processing operations

To perform a FWI on the real data, it is necessary to design a specific processing sequence to carry out on the observed and synthetic data. At first step, to focus the inversion on the first arrivals, a **muting mask** was applied on the seismograms. This mask, specific for each shot, is characterized by a time length of about 1s for all the traces and a ramp of 30ms at the beginning and the end of the mute mask, to avoid an abrupt transition to zero of the signal. Then, to consider low frequencies during the inversion procedure, a **band-pass filter** is applied with a bandpass between 5 Hz and 10 Hz followed by a **trace envelope**. Finally**,** a **trace normalization** was applied overcoming the fact that the modelling is 2D and cannot reproduce 3D geometrical spreading. Figure 8.11 and Figure 8.12 show the shot of Figure 8.9b and the backward ES of Figure 8.10 respectively, after the processing sequence, previously described.

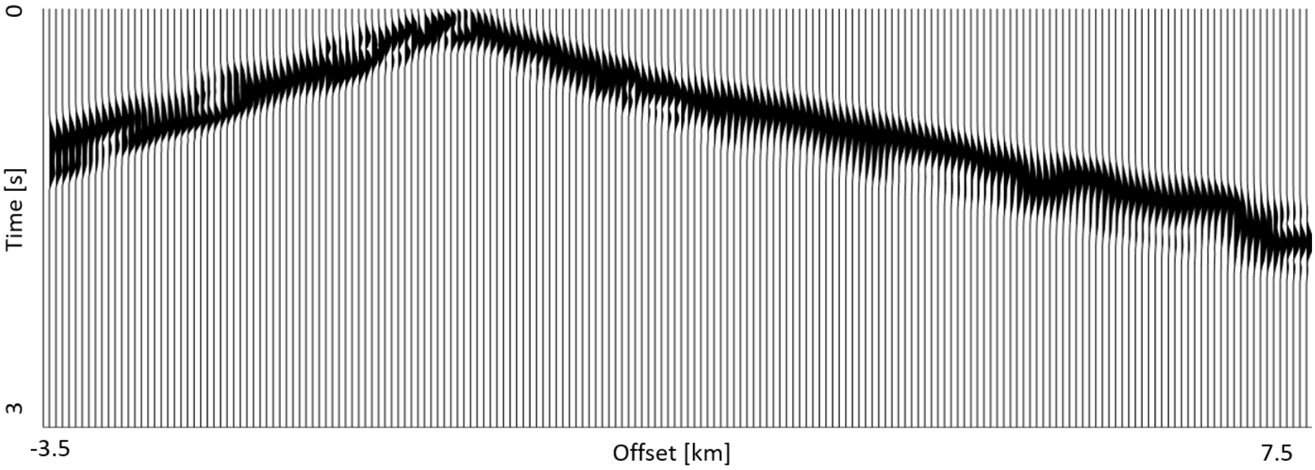


Figure 8.11: Seismogram of Figure 8.9a, after the processing sequence (muting, filtering, trace envelope and trace normalization).

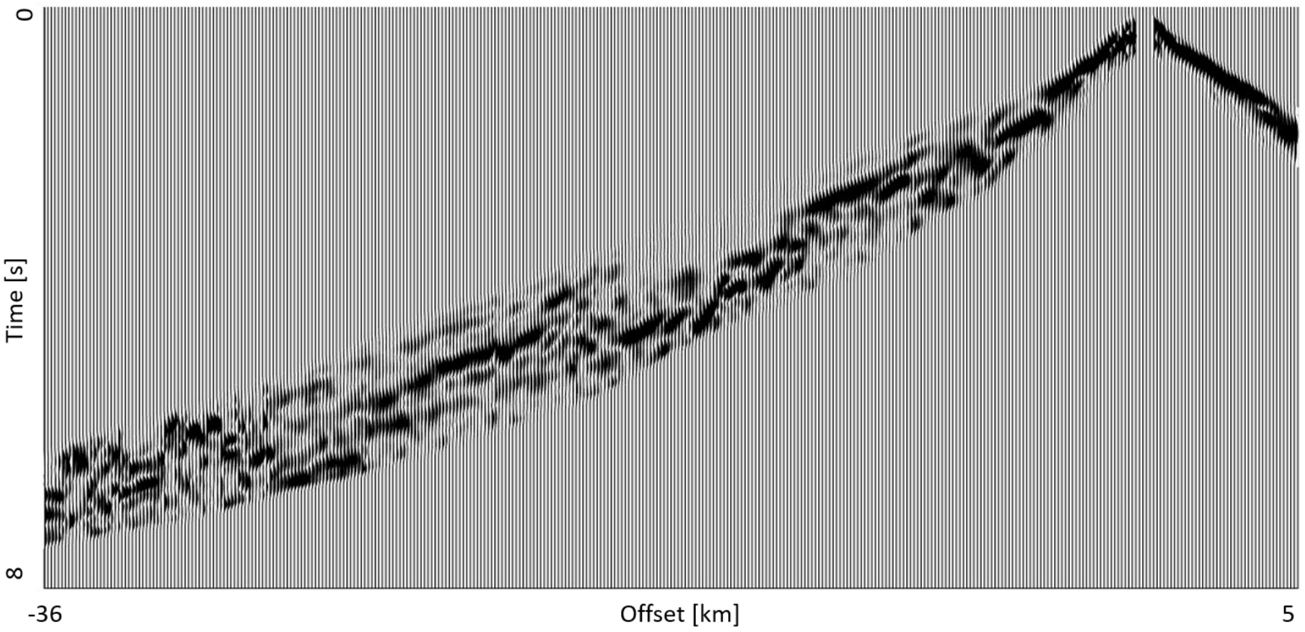


Figure 8.12:Seismogram of Figure 8.10, after the processing sequence (muting, filtering, trace envelope and trace normalization).

The set of the processing operations can be expressed as the application of a processing operator G:

$$G\big(u_0(\boldsymbol{x},t)\big) = G_4\left(G_3\left(G_2\left(G_1\big(u_0(\boldsymbol{x},t)\big)\right)\right)\right), \tag{8.1}$$

where:

- $u_0(\boldsymbol{x},t)$ are the observed seismograms, with applied the noise removal operations;
- G_1 is the muting operator;
- G_2 is the filtering operator;
- G_3 is the trace envelope operator;
- G_4 is the trace normalization operator.

Note that all these operations involve only the recording time and are independent from the position of the receivers.

8.4 Setting up of the modelling to compute the synthetic seismograms

To use the modelling code *fullwave2D* for an acoustic FWI procedure, it is necessary to set all the modelling parameters. In particular, I define:

- the discretization in space (dx, nx, nz, p_{ord});
- the discretization in time (dt, T);
- the seismic wavelet $s(t)$;
- the source and receiver positions on the modelling grid;
- the boundary conditions.

8.4.1 Discretization in space

To set the dimensions of the velocity model and the corresponding modelling grid, it is necessary to consider the length of the seismic profile, and the presence of some a priori knowledge about the acoustic velocities of the area, for the possible implications concerning the numerical dispersion.

I fixed the length of the velocity model to $X = 42\ km$, that is the length of the seismic profile. Fixing the model depth Z is not simple because it depends on the possible illumination I can expect from the shot and the ES data I consider.

In [81] a tomography velocity analysis of the first beaks has shown that the maximum depth penetration of the rays is approximately $1.5km$ for the shot and $3.5km$ for the ES data. Starting from this analysis, but maintaining a certain caution, I consider a depth of $Z = 6km$ of the model, where the zero level correspond to the maximum elevation of the seismic profile, which is $450m$ above the sea-level. To include the topography effects, the velocity of the part of the model above the topography level is 0 km/s.

As modelling grid parameters, I consider a spatial sampling of $dx = 30m$ (with $dx = dz$), with $nx = 1563$ and $nz = 200$. By these modelling parameters, the numerical dispersion relation Equation (2.19) is satisfied for a minimum velocity $v_{min} = 2km/s$ and maximum temporal frequency of about $f_{max} = 10Hz$, using a global order $p_{ord} = 4$. However, to reduce the computational time, I use a local order of approximation of the spatial derivative $p_{ord} = p_{loc}$.

Figure 8.13 shows the modelling grid, where the blue colour area on the top of the model corresponds to the part over the topography level.

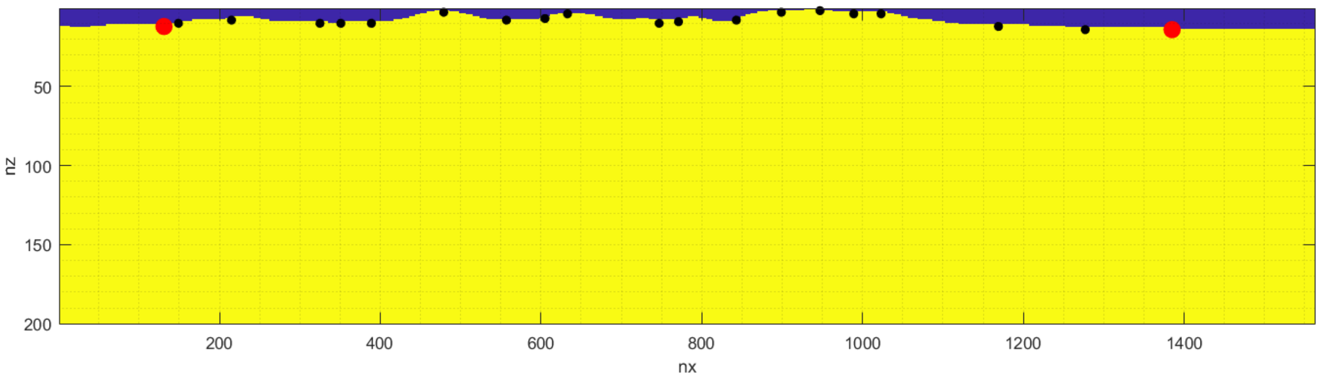


Figure 8.13: Modelling grid, with nx=1563, nz=200,and dx=dz=30m. The blue colour area corresponds to the part above the topography level. The red points represent the position of the ES sources, whereas the black ones represent the position of the 18 shots.

8.4.2 Discretization in time

To fix the time step dt, it is useful to have a raw estimation of the acoustic velocities of the area for the possible implications concerning the numerical stability. In particular, the Equation (2.18) is satisfied, if $dt \leq 0.002s$, considering a maximum acoustic velocity of 7.5km/s. Thus, I consider a time sampling of $dt = 0.002s$, that equals to the recording sampling of the observed seismograms.

Because the mute masks for the ES experiments extend until 8s, I consider a modelling time of $T = 8s$ for these two shots. On the contrary, since the mute masks of the shots extend until 3s, I consider a modelling time of $T = 3s$ for these shots.

8.4.3 Estimation of the seismic wavelet

To simulate the seismic propagation on the modelling grid, it is necessary the knowledge of the seismic wavelet $s(t)$. Since the CROP18a profile data do not provide any information about it, an estimation of the wavelet was necessary from the seismic data. To do this, I apply the following steps:

- the extraction of the seismic traces from each trace with a maximum offset of $-2000m$ and $2000m$;
- the flattening of the first arrivals using a velocity estimated from the picking of the first arrivals, specific for each shot, and zeroing the seismic traces after $600ms$;
- the normalization and sum of all the seismic traces to obtain a mean wavelet;
- the cross-correlation of the mean wavelet with all the traces to compute a value to apply to each trace for a better alignment;
- the removal of all the seismic traces with a static value greater than $40ms$;
- the sum of all the seismic traces to obtain a mean wavelet and zeroing the values after $0.13s$;

Figure 8.14a and Figure 8.14b show the estimated wavelet as a function of time and frequency, respectively.

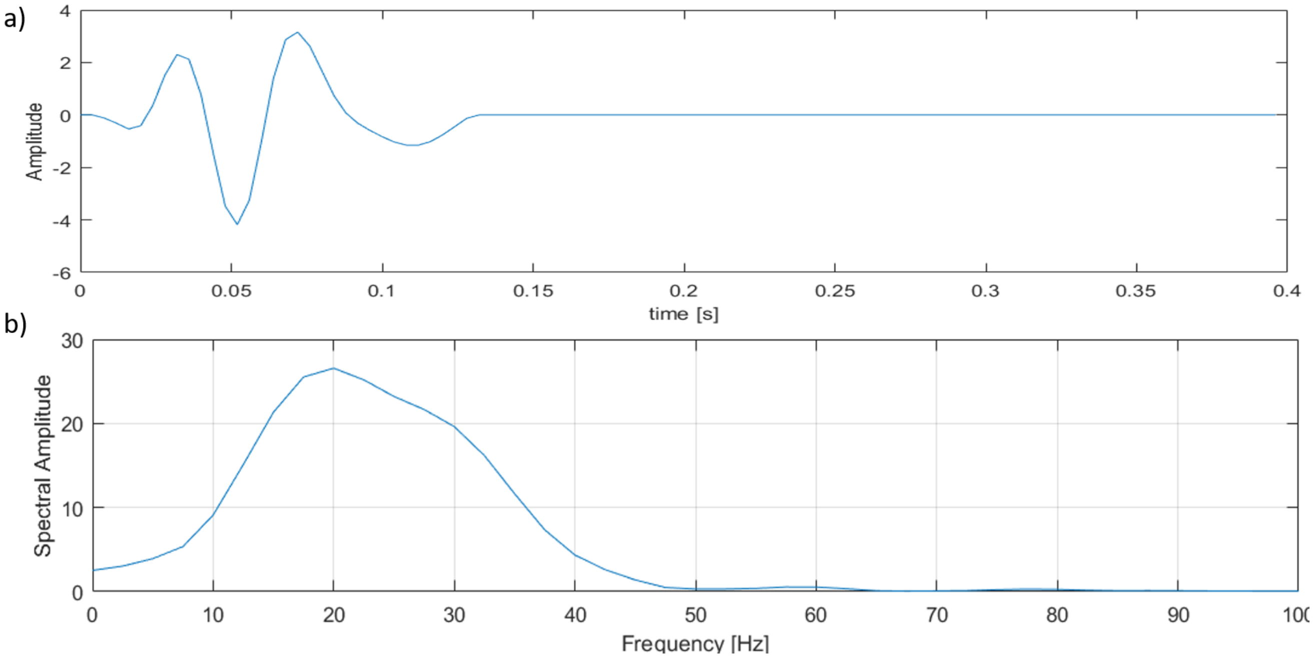


Figure 8.14: Estimated wavelet as a function of a) time and b) temporal frequency.

8.4.4 Source and receivers position

The position of the sources and receivers of all the 20 shots were extracted from the headers of the seismic traces. The positions along the *x*-direction are approximated using the header information relative to the CDP point, whereas the positions along the *z*-direction are approximated using the header information regarding the elevation for the receivers, and the elevation and borehole depth for the sources.

The corresponding grid node positions are obtained sampling these measures by $dx = 30m$. Table 8.3 lists the grid node positions of the sources. Using this information is possible to build the source and receiver text files.

source	*nx*	*nz*
1	131	12
2	149	10
3	215	8
4	325	10
5	351	10
6	389	10
7	479	3
8	557	8
9	605	7
10	633	4
11	747	10
12	771	9
13	843	8

14	899	3
15	947	2
16	989	4
17	1023	4
18	1169	12
19	1277	14
20	1385	14

Table 8.3: Positions of the 20 sources on the modelling grid.

8.4.5 Boundary conditions

A reflecting boundary condition is considered for the top side of the model to simulate the air-land contrast of velocity and density. On the other hand, an absorbing boundary condition is considered for left, right and bottom sides of the model, with absorbing boundary layers of 30 grid nodes for each side.

8.5 Setting the misfit function and the inversion grid

To apply a FWI procedure to invert the observed seismograms, I set up:

- the **misfit function** used to measure the difference between the predicted and the observed seismograms;
- the **inversion grid** that is the set of the unknows used for the inversion algorithm;
- the **inversion procedure** used to estimate the models.

8.5.1 Misfit function

As misfit function, I use the sum of the all the L^2-norm difference between the observed and the synthetic seismograms

$$F(v) = \sum_{s=1}^{20}\left(\frac{1}{2}\sum_{r=1}^{n_{s,r}}\left(\sum_{k=1}^{n_t}\left|G\left(u(v,t_k,\boldsymbol{x}^{r,s},\boldsymbol{x}^s)\right) - G\left(u_0(\boldsymbol{x},t_k,\boldsymbol{x}^{r,s},\boldsymbol{x}^s)\right)\right|^2\right)dt\right). \tag{8.2}$$

where:

- $n_{s,r}$ is the number of receivers for each source (192 for the shot, 750 for the forward ES and 714 for the backward ES)
- $\boldsymbol{x}^{r,s}$ are the positions of the receivers, different for each source;
- $\boldsymbol{x}^s$ are the positions of the sources;
- t_k are the temporal samples of the seismograms;
- $u_0(\boldsymbol{x},t_k,\boldsymbol{x}^r,\boldsymbol{x}^s)$ are the observed seismograms, with the application of the noise removal operations;
- $u(v,t_k,\boldsymbol{x}^r,\boldsymbol{x}^s)$ are the synthetic seismograms;
- G is the processing operator, as in Equation (8.1), that is the set of the processing operations of Section 8.3.2.

Note that the operator G is applied to the observed and synthetic seismograms, whereas the noise removal operations are applied only to the observed ones.

8.5.2 Inversion grids

I make use of two different inversion grids: a **coarse inversion grid** and a **fine inversion grid**. The coarse inversion grid (the black points in Figure 8.15), consists of 105 grid nodes, organized in 6 rows. The number of grid per row decreases as a function of the depth, from 30 nodes for the first row, to 5 nodes for the last one. The first row of the inversion grid is located on the topography level. The last row is located on the bottom of the modelling grid. The other rows are located between the topography level and the bottom of the modelling grid, at increasing depth from the first row and with the distance between the adjacent rows that increases as a function of the depth. The use of a non-regular grid helps the inversion to focus on the upper part of the model, where the illumination is high, trying to make the velocity variations of the bottom of the model more sensible, i.e. where the illumination is poor. I used a three-step interpolation procedure to pass from the coarse to the modelling grid. In the first step each coarse row is interpolated linearly along row-direction to obtain a row of 1563 grid nodes, the same horizontal dimension of the modelling grid rows. Then each grid node is translated downward of a certain quantity, function of the row direction, to simulate the topography of the model. Eventually, each column is interpolated along the depth direction to obtain the column dimension of the modelling grid.

The fine inversion grid (the red points in Figure 8.15) consists of 2260 grid nodes, organized in 20 rows. Also in this case the number of grid nodes per rows decreases as a function of depth from 151 nodes for the first row to 75 nodes for the last one, and I use a similar three-step interpolation procedure to obtain the modelling grid from the coarse one.

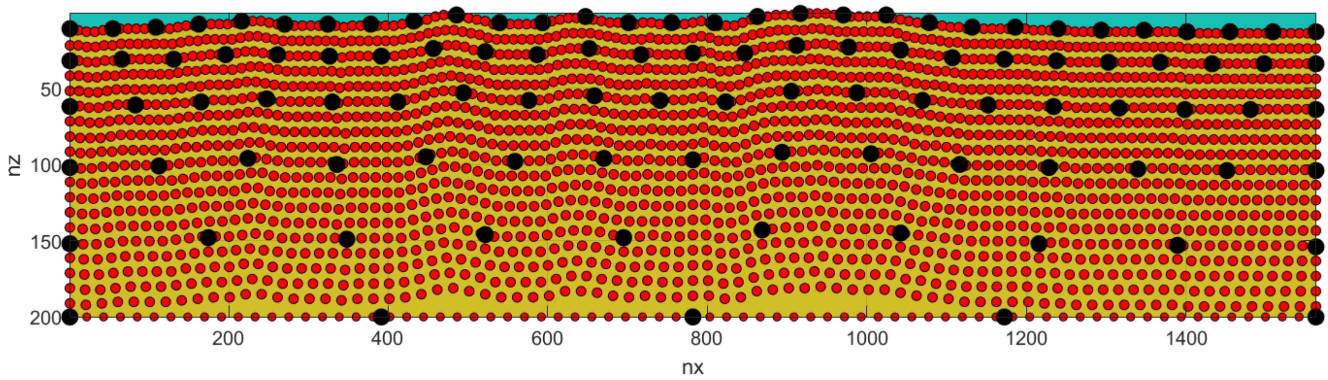


Figure 8.15: Fine grid (red points) and coarse grid (black points) superimposed on the modelling grid.

8.5.3 Inversion procedure

As inversion procedure, I use the following two-step approach:

- in the first step I use the GA algorithm to estimate a smooth model on the coarse inversion grid;
- in the second step I use the conjugate gradient algorithm on the fine inversion grid, using the best GA model as starting model.

8.6 The genetic algorithms parameters

As we saw in Chapter 6, the GA algorithms have some parameters that must be tuned up to perform the optimization efficiently.

The search domain (Figure 8.16) consists of the search ranges of the 105 unknows, that varies from 2 km/s to 6 km/s for the unknowns of the first row of the coarse grid and from 5.5 km/s to 7.7 km/s for the unknowns of the last row. The range of the unknows decrease with the depth, passing from 4km/s for the first row to 2km/s for the last one, to avoid possible high-velocity variations of the model that are not possible to estimate from the observed data.

The dimension of the GA initial population is set to $n_{pop} = 1050$ and the models of initial population are chosen randomly inside the velocity ranges, the selection rate is set to 0.8 and the mutation rate is set to 1/105, according to the value of *Table 6.3*, with $n = 105$.

Finally, I parallelize the computational of 840 offsprings of the new population using 7 servers Intel(R) Xeon(R) CPU E5-2630 v4 (20x 2.20GHz) and 128 GB DDR4 RAM.

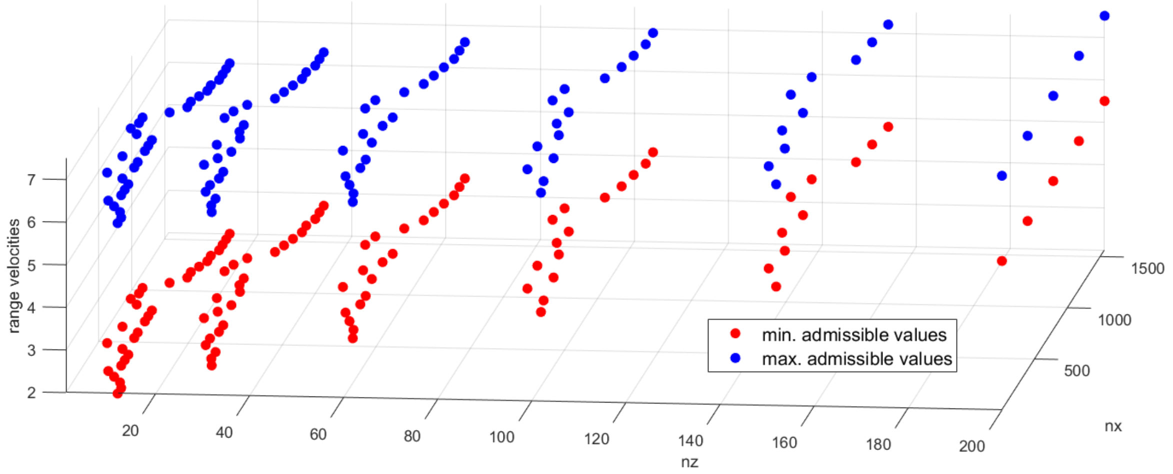


Figure 8.16: Search domain consisting of the search range, that varies from 2 km/s to 6 km/s for the unknowns of first row of the coarse grid and from 5.5 km/s to 7.5km/s for the unknowns at last row.

8.7 Global FWI using GA

In the first step, I consider a GA minimization on the coarse inversion grid (Figure 8.15). 200 generations are performed, corresponding to an initial population of 1050 individuals and 168000 offsprings, 840 for each generation.

Figure 8.17 shows the evolution of the data misfit during the GA minimization procedure. The red and the blue curves represent the mean and the lowest misfit of the various generations, respectively. All these values are divided by the lowest misfit value of the first generation. We can note:

- a decrease of the blue curve, that passes from 1 to about 0.47 at the 200 generation, with a substantial flattening of the curve after 160 generations;
- a decrease of the distance between the blue and the red curves, which passes from 0.24 to about 0.0064 at the last generation;

These two facts could imply a concentration of the individuals of the population in a specific zone of the search domain, where the global minimum of the misfit function could be. To verify this, in Figure 8.18 the average value of the distance between the individuals of the population as a function of the generation is showed, divided by the average distance of the first generation. We can note a significant reduction of this value, which passes from 1 to 0.1, and a flatting of the curve after 190 generations.

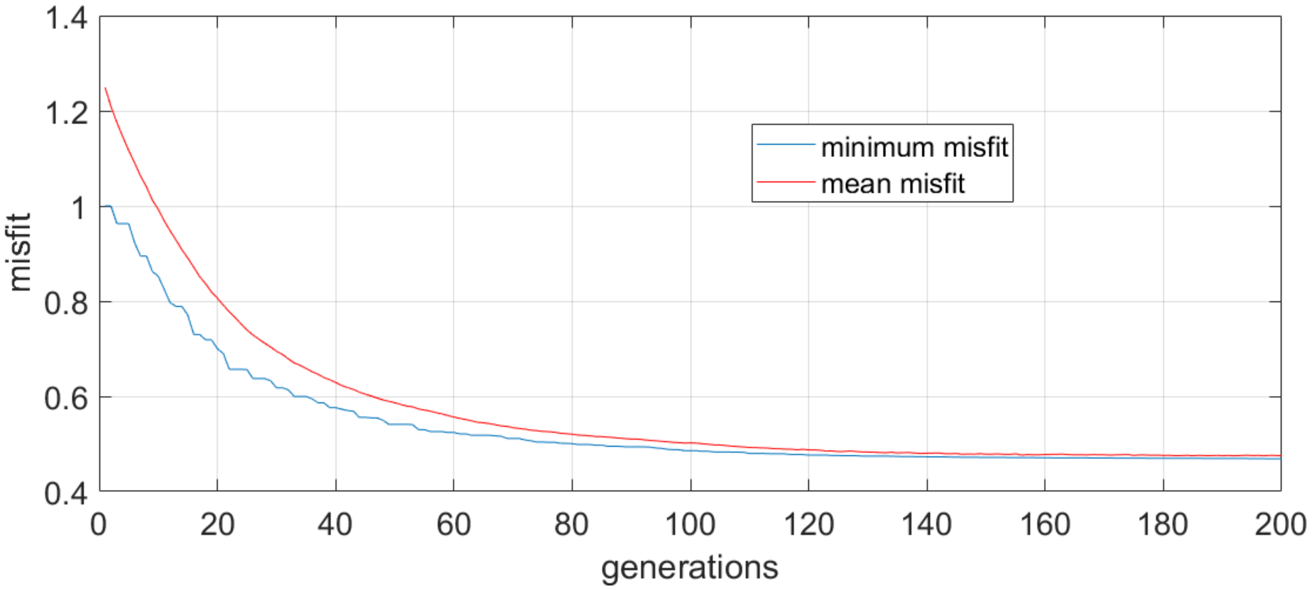


Figure 8.17: Evolution of the misfit as a function of generations. The blue curve represents the lowest misfit obtained by each generation, whereas the red curve represents the mean misfit. All these values are divided by the lowest misfit value of the first generation.

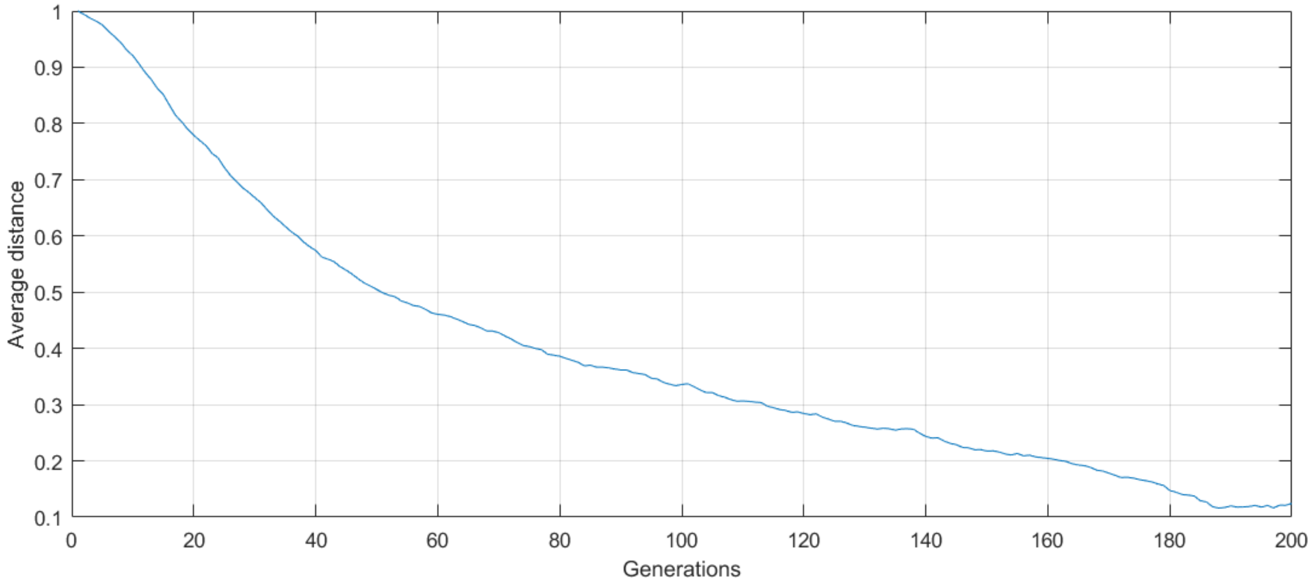


Figure 8.18: Evolution of the average distance between the individuals of the population as a function of the generation, divided by the average distance of the first generation.

Figure 8.19 shows the GA model with the lowest misfit value at the first generation (Figure 8.19a), and the last generation (Figure 8.19b), interpolated on the modelling grid. To check the sensitivity of the misfit function as a function of the different parts of final model, in Figure 8.19c I represented the absolute values of the diagonal elements $\left|\frac{\partial^2 f}{\partial {x_i}^2}\right|$ of the Hessian matrix, interpolated on the modelling grid. A large element indicates that the corresponding unknowns x_i is well resolved, which means that the misfit function increases substantially if the value of this unknown is changed.

As we expect, there is a low sensitivity of the unknows on the lateral sides of the model because of the poor illumination of the seismic data in these zones. Besides, there is a decrease of the sensitivity range along the x-direction as a function of the depth.

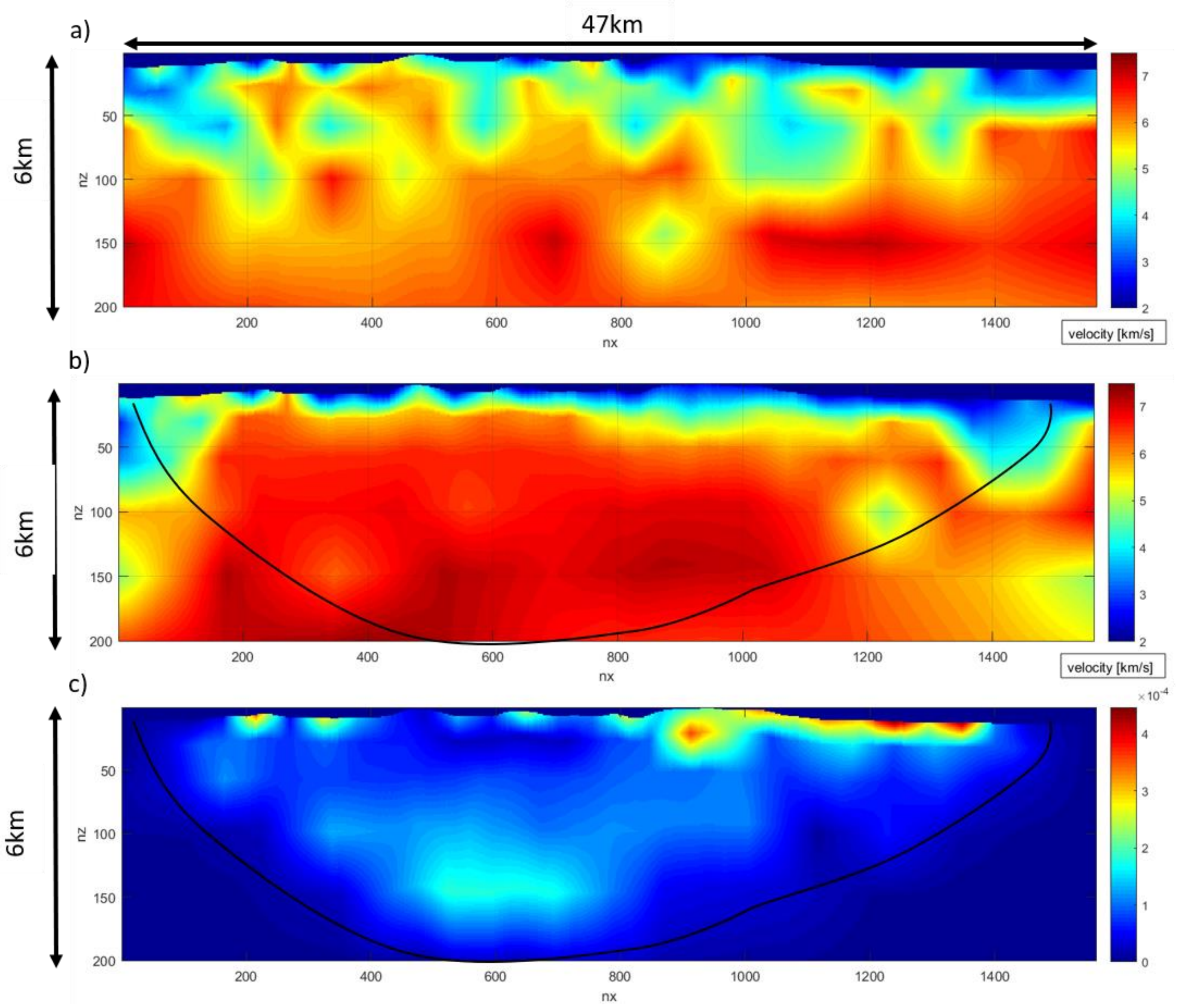


Figure 8.19: a) Best GA model at first generation on the modelling grid; b) best GA model at last generation on the modelling grid; c) the values of the diagonal elements of the Hessian matrix interpolated on the modelling grid. The misfit function is not sensible to the region outside the black curve.

The model obtained shows a velocity range between $2.5 km/s$ in the shallow part and 6 km/s at about 4-5 km, in agreement with the geological context [82] [83]. The upper part (above 500 m) is dominated by Neogene continental and marine sediments and by the Ligurian units with a velocity between 2.5 km/s and 4 km/s. At a depth of $500m$ until $1km$, the velocities contrast indicates the metamorphic basement, which is considered very shallow in this area and characterized by a velocity slightly higher than $5 - 6\ km/s$,. The deeper part of the model presents a high-velocity zone (about 7 km/s), which climbs up from North to South. This area could be associated with the emplacement of magmatic bodies [84].

The inversion velocities situated on the bottom corners of the model are not reliable because of the very low sensitivity of the misfit function to these parts of the model. Finally, there is a low-velocity zone situated between 36 and 39 km at a depth of $3km$, given by the value of 87th unknown of the coarse

grid, which has a value of 4.7 km/s. This value represents a significant anomaly compared to the surrounding area that has a mean velocity of about 6 km/s. However, it is likely a numerical artefact caused by the random mutations of the GA and the low sensitivity of the misfit function for this area.
To check this fact, in Figure 8.20 I studied the behaviour of the misfit function, varying only the value of the unknown that causes this anomaly in a range between 3 km/s and 6.5 km/s. We can note also the presence of a pronounced local minimum around the velocity of 5.7 km/s in accordance with the velocity of the surrounding zone.

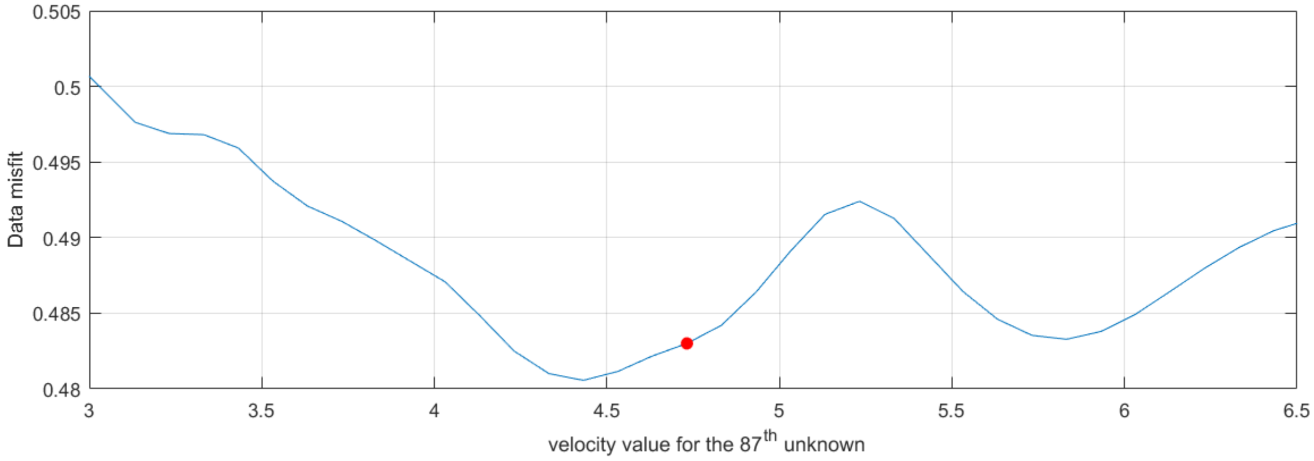


Figure 8.20 The behaviour of the misfit function, varying the value of the 87th unknown in a range between 3 km/s and 6.5 km/s, and using the values obtained by the GA algorithm at the last generation for the other unknowns. The red point represents the value obtained by the GA algorithm.

On the seismograms, replacing the value of the 87th unknown with the velocity of 5.7 km/s, no significant difference is noted except for the part of backward ES experiment at long offset, which is characterized by a low S/N ratio. This also suggests the non-reliability of this inversion velocity.
The Figure 8.21 shows the model obtained by the one of Figure 8.19b, replacing the value of the 87th unknown with the velocity of 5.7 km/s. I will consider this last model as the **best GA model**.

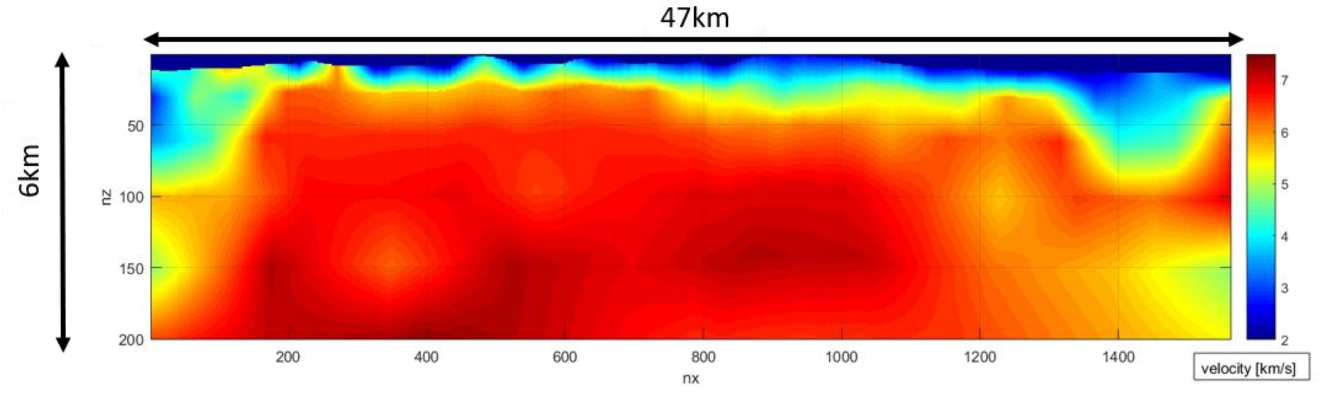


Figure 8.21: The model obtained by the one of Figure 8.19b, replacing the value of the 87th unknown with the velocity of 5.7 km/s, interpolated on the modelling grid. This is considered as the best GA model.

Finally, Figure 8.22 and Figure 8.23 show a comparison between the predicted and the observed seismograms relative to leftmost shot for the model with the lowest misfit value at the first generation of GA (Figure 8.22) and the best GA model (Figure 85). a) and b) in each figure display the predicted seismogram and the difference (in module) between the observed and predicted seismogram, respectively. Note a general partial match of some of the main events on the seismograms. A similar comparison of the predicted and the observed seismograms for the backward ES experiment is in Figure 8.24 and Figure 8.25.

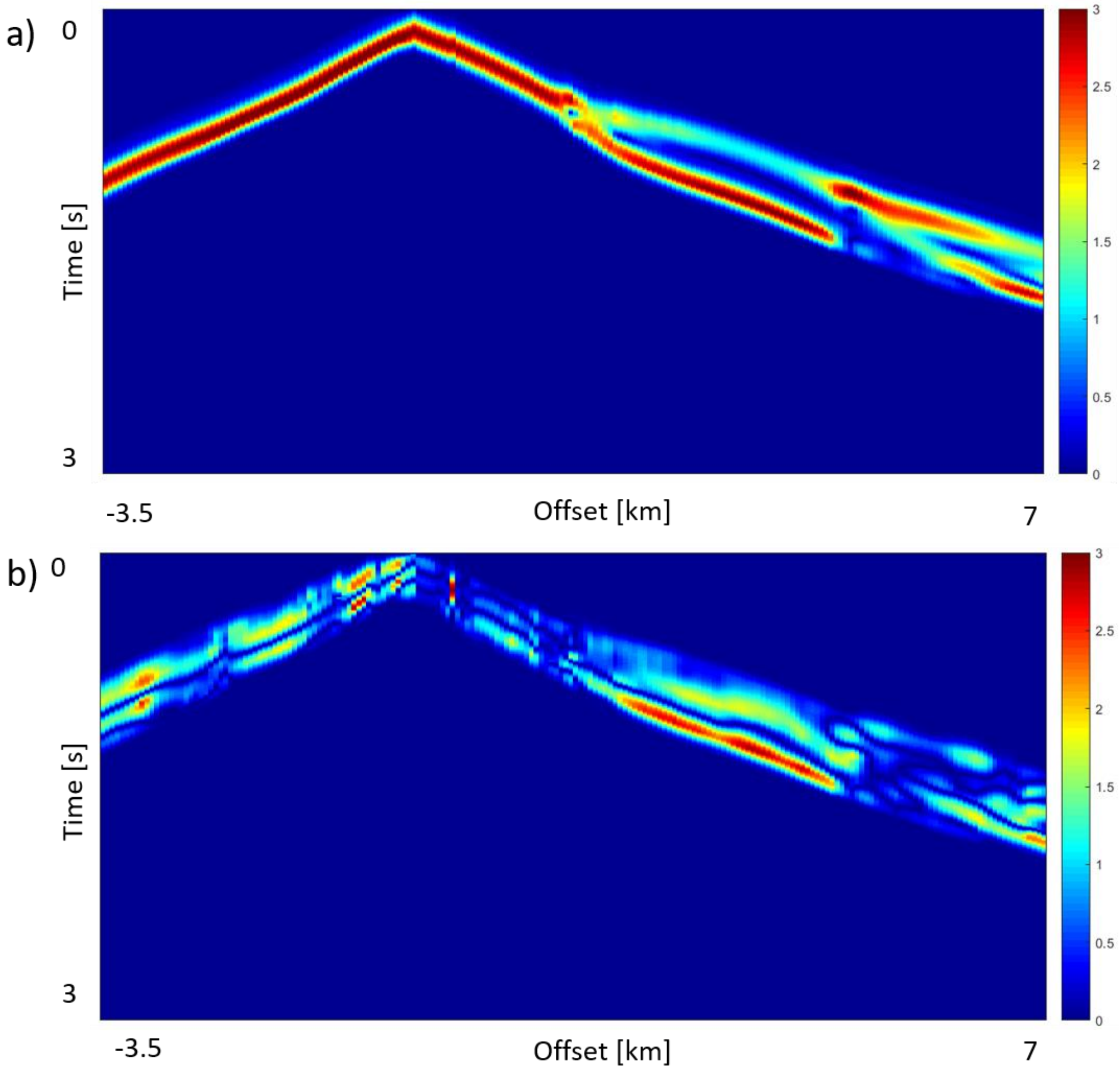


Figure 8.22: a) The predicted data of the leftmost shot, using the best model of the first generation of GA (Figure 8.19a) and b) the difference between the observed and the predicted seismograms.

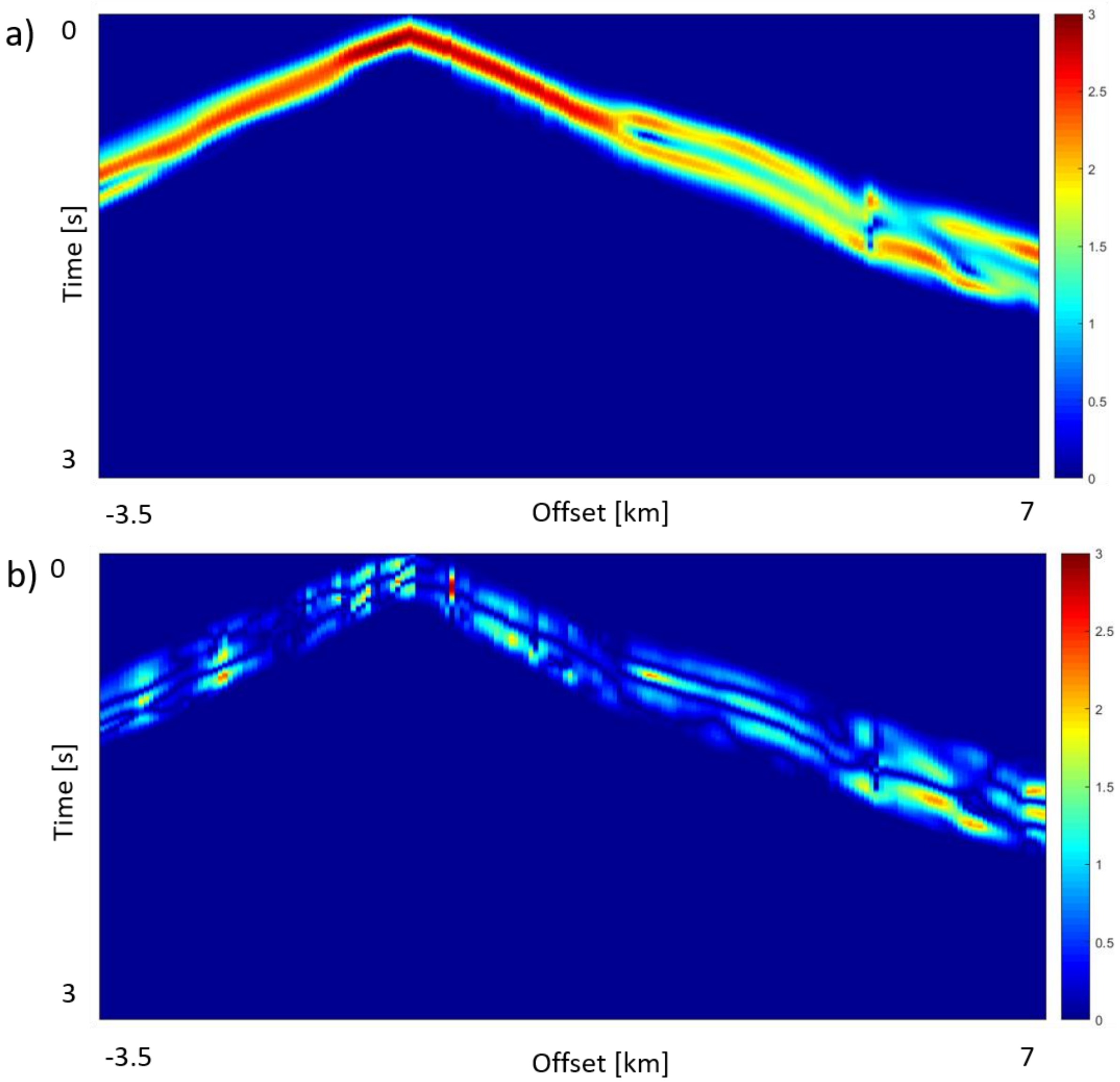


Figure 8.23: a) The predicted data of the leftmost shot, using the best model of the best GA model (Figure 8.21) and b) the difference between the observed and the predicted seismograms.

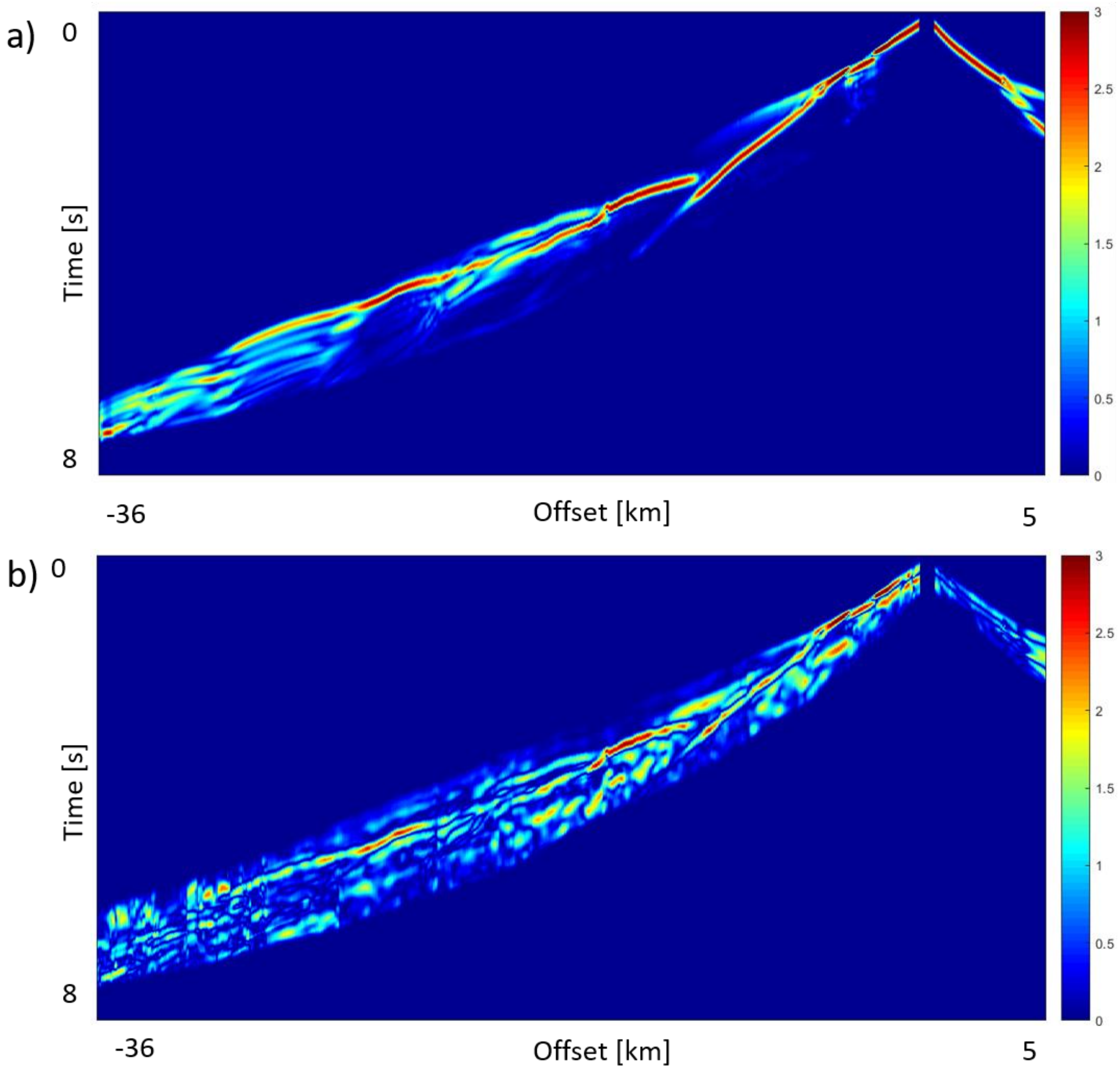


Figure 8.24: a) The predicted data of the backward ES, using the best model of the first generation of GA (Figure 75a) and b) the difference between the observed and the predicted seismograms.

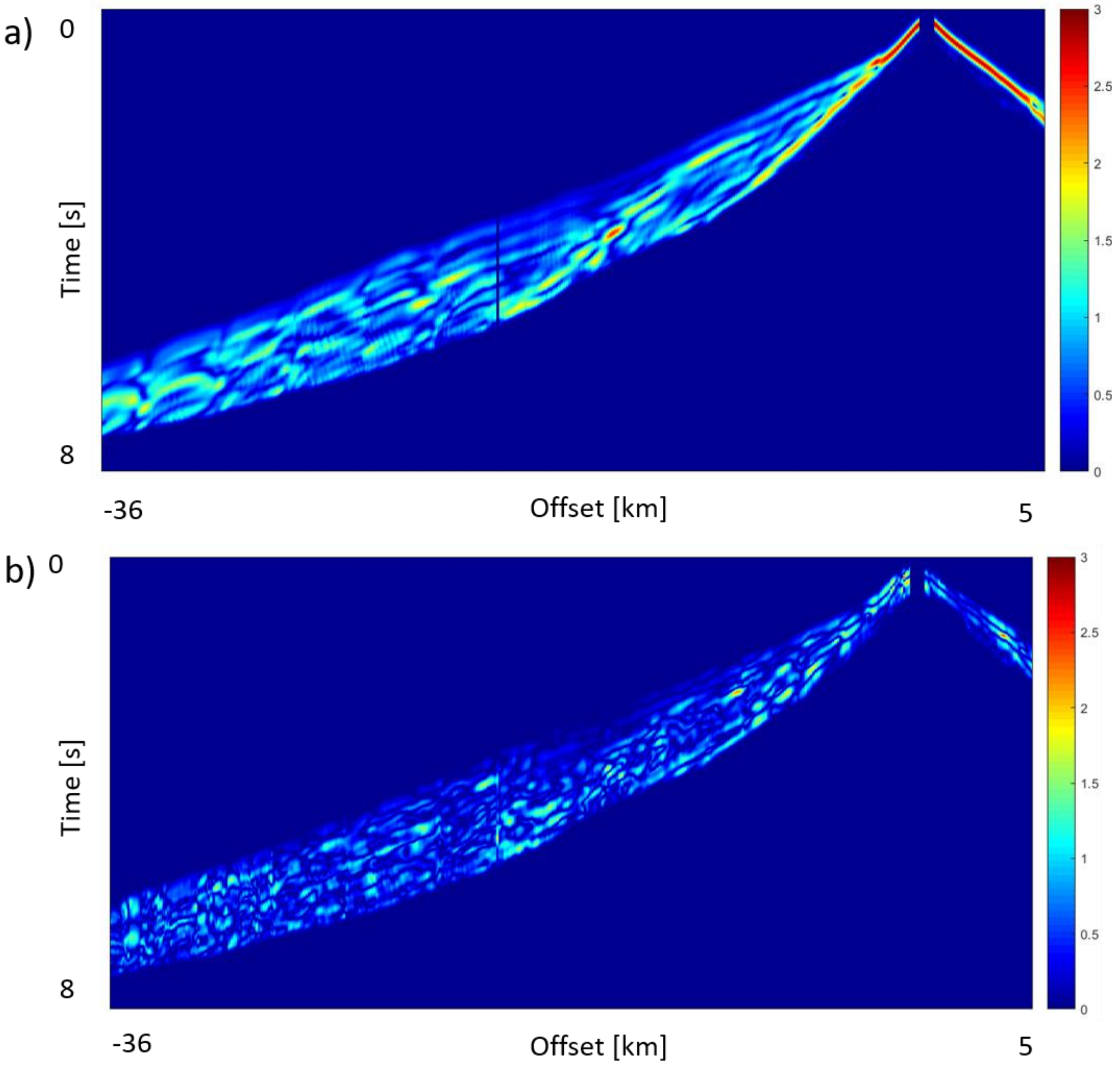


Figure 8.25: a) The predicted data of the backward ES, using the best model of the best GA model (Figure 8.21) and b) the difference between the observed and the predicted seismograms.

8.8 Local FWI starting from the best GA model

For the second step, I used the best GA model (Figure 8.21) obtained in the previous step as starting model on the fine inversion grid (the red points in Figure 8.15), for a local optimization of the misfit function. As a local algorithm, I make use of the Conjugate gradient method [66], where the descend direction h_k and the step length γ_k are computed as described in the previous chapter in Section 7.5. Figure 8.26 displays the evolution of the data misfit, in which we can note a decrease of about 49% of the initial value. However, after 200 iterations the minimization procedure seems to reach a stable value of the data misfit, and this is therefore we stop the iterative procedure.

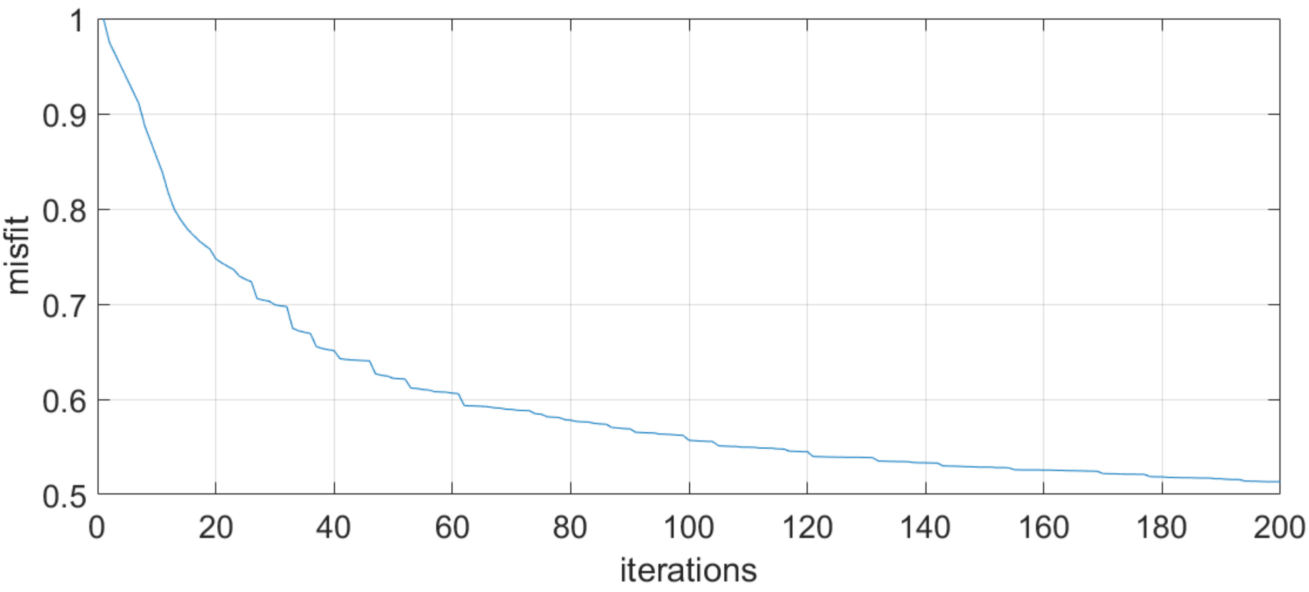


Figure 8.26: Evolution of data misfit for the local minimization procedures on the fine inversion grid.

Figure 8.27a shows the final model obtained at the end of the procedure, whereas Figure 8.27b shows the difference between the final and the starting model.

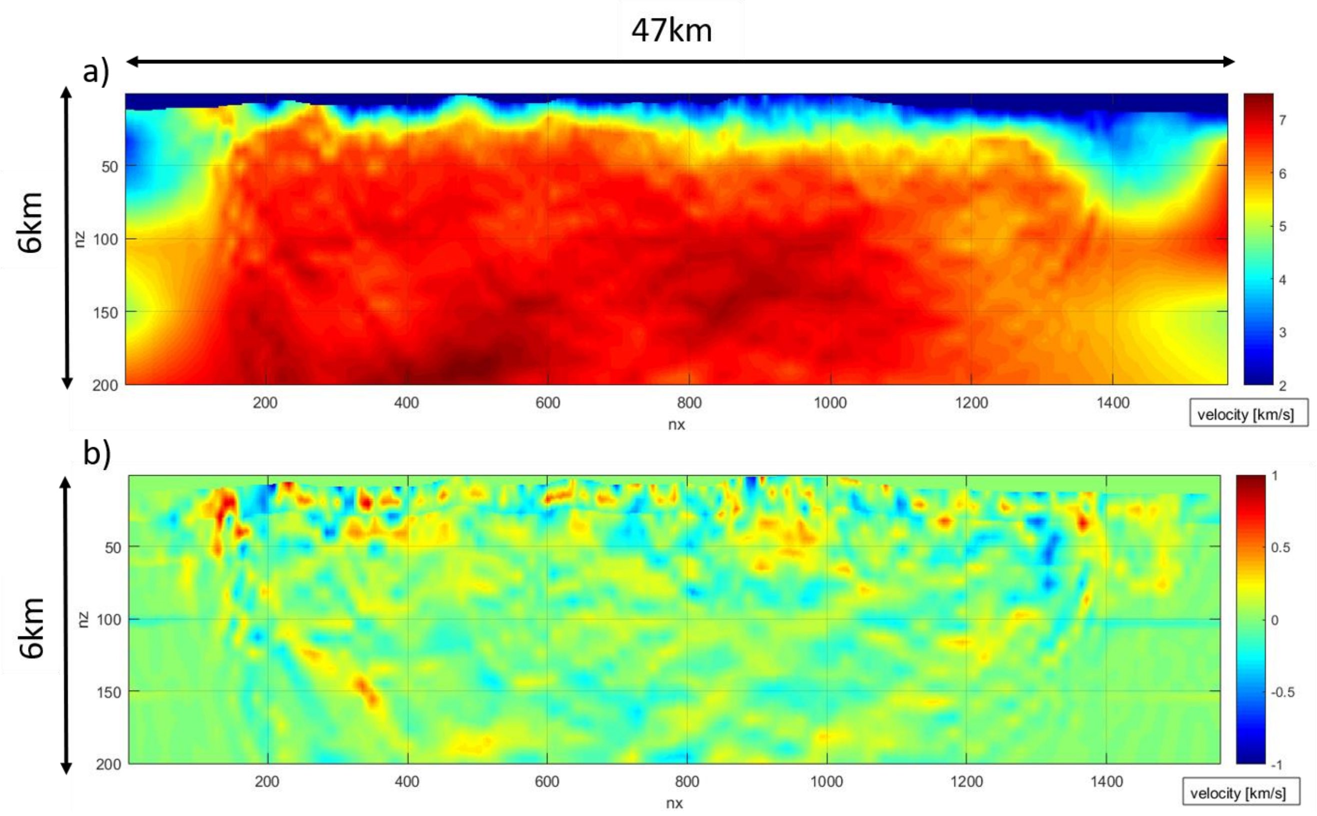


Figure 8.27: a) The final model obtained by the local minimization interpolated on the modelling grid and b) the difference between the final local model and the starting one (best GA model interpolated on the modelling grid).

In the upper part of the model until a depth of 2 km there is a significant change of the velocity values, but the long-wavelength structures of the starting model are not significantly changed. Finer details are added to the bottom part of the model where there is the high velocity zone of 7 km/s. Eventually, no significant changes are observed in the zone of velocity anomaly previously analyzed.

As proof of the effectiveness of the two-step method described, in Figure 8.28 I show final predicted seismograms (Figure 8.27a) and the difference between the predicted and the observed seismograms (Figure 8.27b), relative to the leftmost shot. A similar comparison for the backward ES experiment is in Figure 8.29.

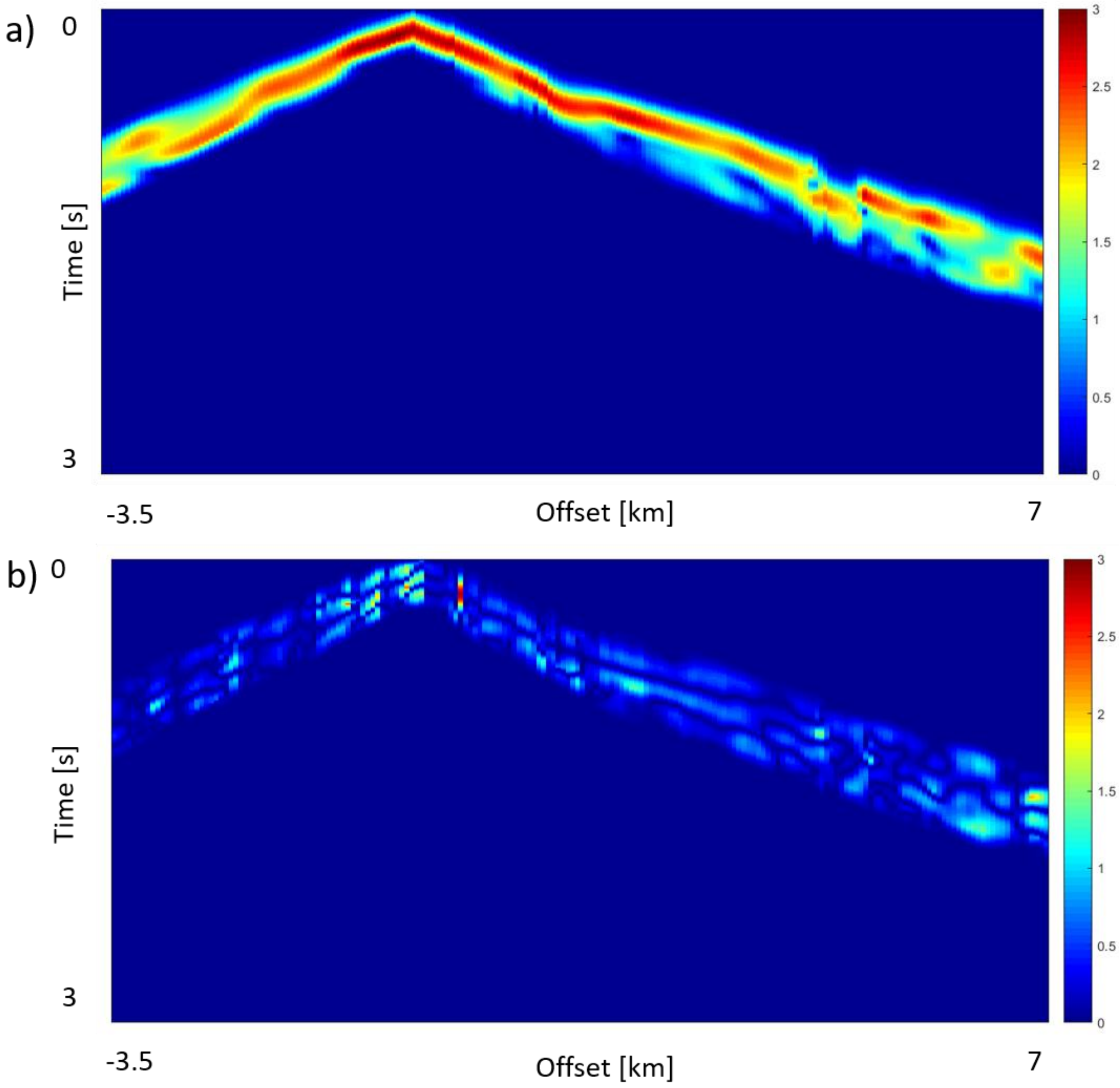


Figure 8.28: a) The predicted data of the leftmost shot, using the final model of the local minimization (Figure 8.27a), and b) the difference between the observed and the predicted seismograms.

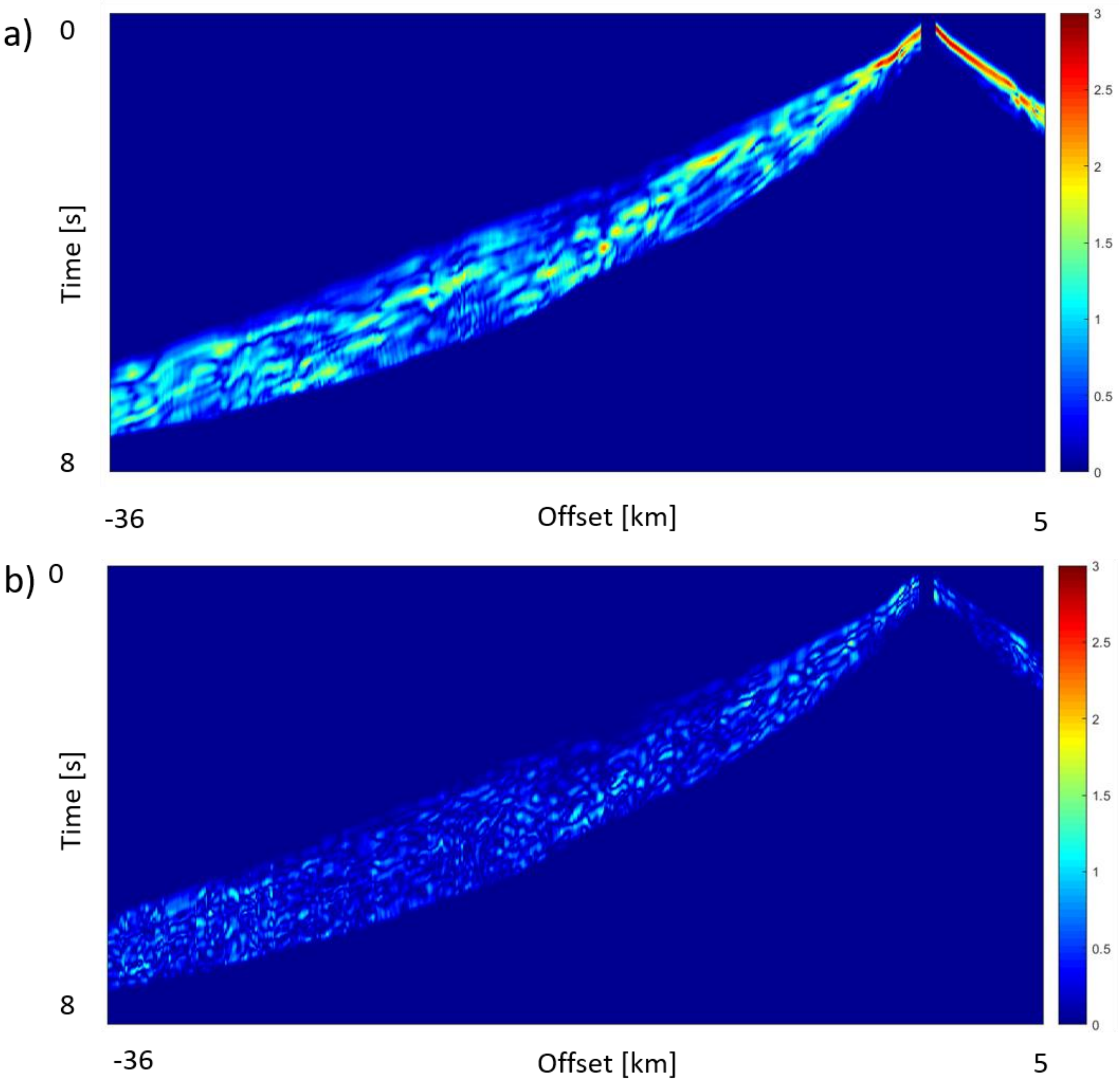


Figure 8.29: a) The predicted data of the backward ES experiment using the final model of the local minimization (Figure 8.27a), and b) the difference between the observed and the predicted seismograms.

8.9 Post-stack migration using the best FWI model

Using the final velocity model obtained by the FWI procedure (Figure 8.27a), I perform a post-stack depth Kirchoff migration of the available stacked seismic section, with the final velocity model obtained by the FWI procedure shown Figure 8.27a. Figure 8.30a shows the migrated section obtained up to the depth $Z = 6km$, whereas Figure 8.30b shows the final model superimposed to the migrated section. As can be noted, the main stronger events are in accordance with the higher velocity contrasts observed in the model and located at a depth between 0.25 and 2 km.

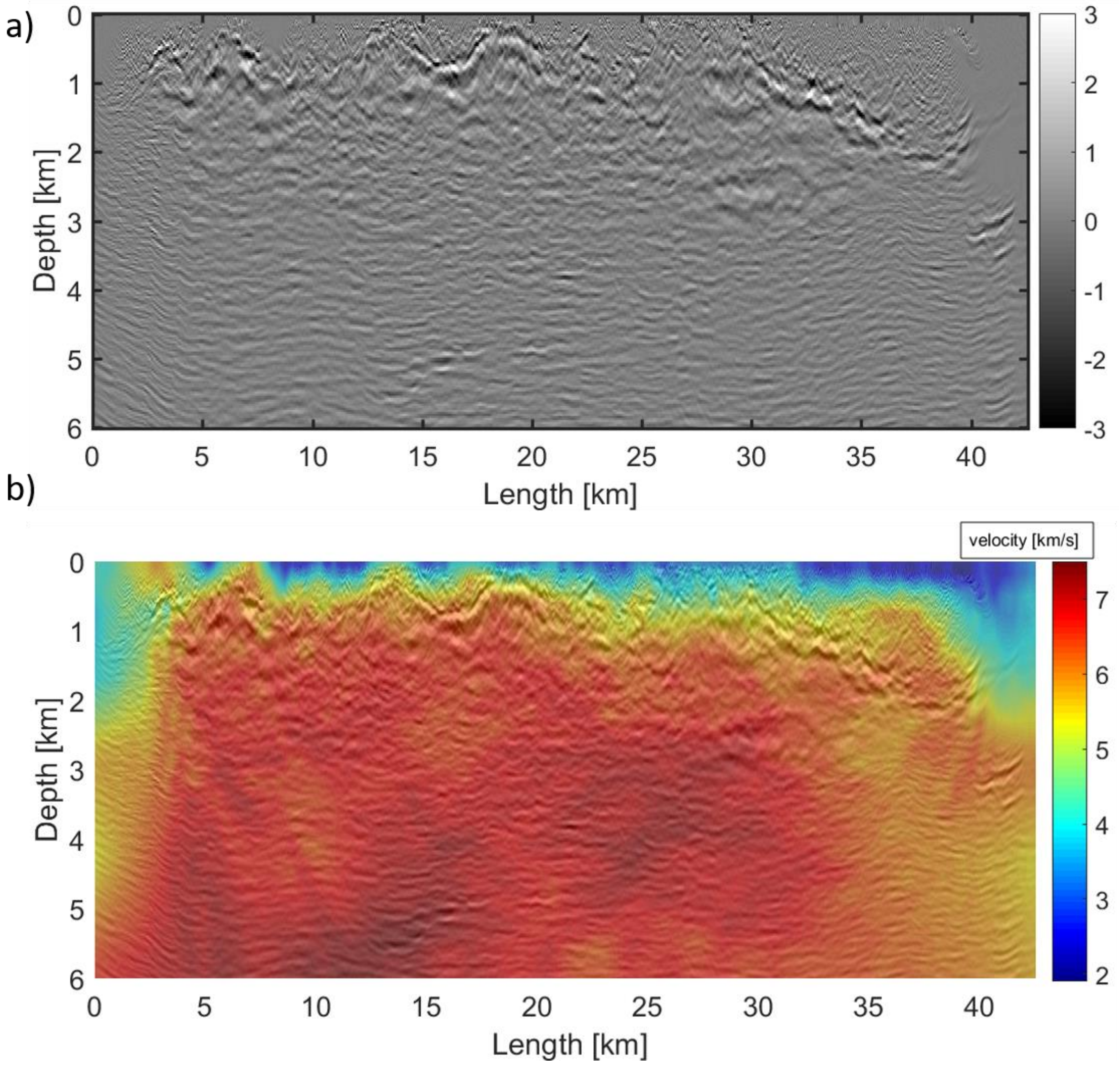


Figure 8.30: a) Depth section obtained after the post-stack depth migration of the time section until T=3s, using the model in Figure 8.27a. b) Depth section superimposed on the velocity model. Note the good correspondence between the main stronger events and the higher velocity contrasts observed in the model and located at a depth between 0.25 and 1 km.

Because of the low S/N ratio of the seismic data, it was not possible to obtain additional information from the Common Image Gathers, or CIGs, computed by the pre-stack depth migration of the seismic data.

8.10 Conclusions and results

In this chapter, I described an acoustic FWI experience made on a 2D seismic land data. In particular, I used a two-step procedure based first on the application of the GA on a coarse grid, and then of the conjugate gradient algorithm on a fine inversion grid, using the best GA model as starting model.

The final model agrees with the geological context, and the quality of the result is assessed by the good correspondence between the predicted and the observed data. Moreover, the depth section obtained by the post-stack migration reveals some events in accordance with the higher velocity contrasts observed in the final model.

Chapter 9.

Experience of local FWI on real marine data

In this final chapter, I illustrate the application of local FWI on a 2D marine seismic profile to estimate an acoustic velocity field for a pre-stack depth migration of the seismic data. As a starting model of the local procedure, I have used a model presented in [10] by a global optimization method based on GA. The synthetic seismograms and the gradient of the misfit function are computed using the *fullwave2D* code. See [85] for more details. The authors wish to thank ENI for the permission to present the results.

9.1 The seismic data

The seismic data pertains to an inline 2D data set extracted from a 3D marine acquisition, acquired by ENI. The total length of the seismic profile is about 11.5km and consists of 765 shot gathers. The sources are located at a depth of $12.5m$ and separated by the same distance $\Delta_s = 12.5$m. Each shot gather is formed by 159 receivers located at a depth of $12.5m$, with a uniform spacing between receivers $\Delta_r = 25m$. The minimum offset is $150m$, while the maximum offset is about $4.1km$. The time sampling is $dt = 4\ ms$, while the recording time is $T = 4.6s$. The seabed is considered flat with a depth of approximately $300m$ and the water velocity is equal to $1.48km/s$.

To reduce the computational cost of the FWI, the length of the examined area is cut to the part of the seismic profile between the CDP 161 and 737, for a total length of about $7.2km$. Besides, I have selected only 56 shot gathers distributed along this part of the profile, with a uniform distance of $\Delta_s = 125m$. Figure 9.1 shows a sketch of the seismic profile: the red row represents the part of the profile used for the inversion, whereas the black and red points represent the position of the sources considered.

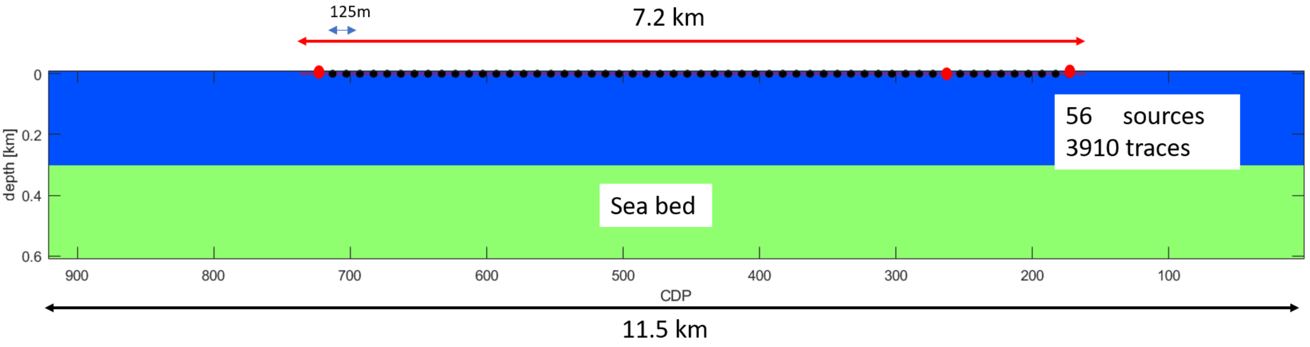


Figure 9.1: Sketch of the seismic profile: the red row represents the part of the profile used for the inversion, whereas the black and red points represent the position of the sources considered. The seismograms of the red point sources are displayed in Figure 9.2.

Eventually, I have reduced the record length to $T = 1.6s$ and the maximum offset to about 2km, to focus the inversion on the diving waves and the shallow reflections of the data. In this way, the number of traces for shot gather changes from a minimum of 7 traces for the first shot to a maximum of 80 for the shots between 16 and 56, for a total number of 3910 traces.

Figure 9.2 shows three of the shot gathers at the begin, at the middle and the end of segment considered for the inversion procedure. Note that the number of seismic traces of each shot depends on its position along the profile.

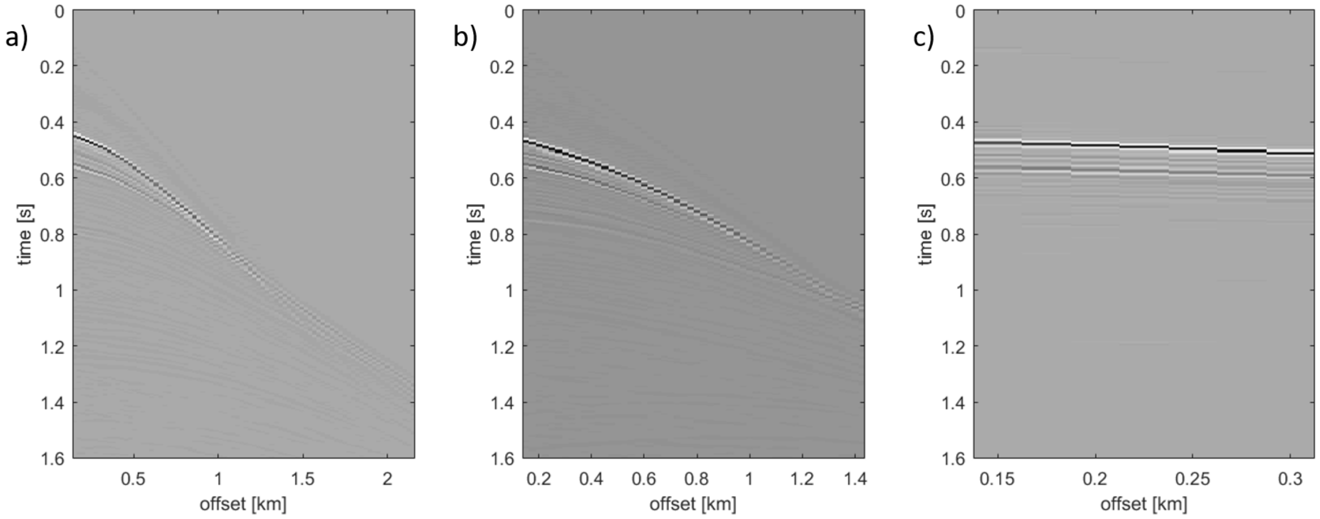


Figure 9.2: Three of the 56 shot gathers considered for the inversion procedure. Note the number of seismic traces of each shot depends on its position along the profile.

9.2 Processing of the seismic data for 2D acoustic FWI

To perform a FWI I design a specific processing sequence to carry out on the observed and synthetic data, similar to that of the previous chapter, consisting of:

- a **muting mask** to focus the inversion on the diving waves and the shallow reflections of the data, with a time window length that varies from a minimum of 0.1 s to a maximum of $0.5\ s$;
- a **low-pass filter** up to $10Hz$, to reduce the cycle skipping effect and to consider low frequencies during the inversion procedure;
- a **trace envelope** to reduce the non-linearity of the misfit function, and enforce the information of low frequencies;
- a **trace normalization** to enforce the information of the seismograms at longest offset and because the modelling is 2D and cannot reproduce 3D geometrical spreading.

The set of the processing operations can be expressed as the application of a processing operator G:

$$G\big(u_0(\boldsymbol{x},t)\big) = G_4\left(G_3\left(G_2\left(G_1\big(u_0(\boldsymbol{x},t)\big)\right)\right)\right), \tag{9.1}$$

where:

- $u_0(\boldsymbol{x},t)$ are the observed seismograms;
- G_1 is the muting operator;
- G_2 is the filtering operator;
- G_3 is the trace envelope operator;
- G_4 is the trace normalization operator.

Note that all these operations involve only the recording time and are independent of the position of the receivers. Figure 9.3 shows the gather of Figure 9.2, after the processing sequence.

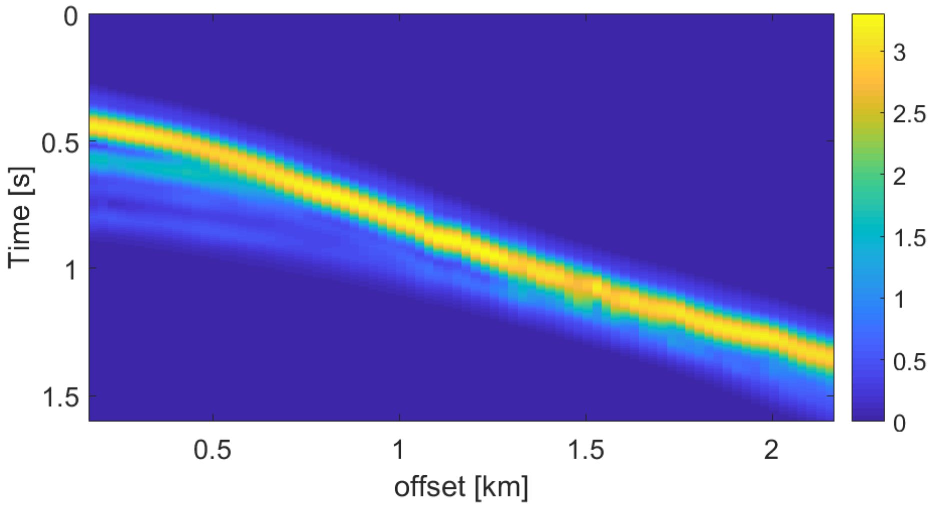


Figure 9.3: Seismogram of Figure 9.2a, after the processing sequence (muting, filtering, trace envelope and trace normalization).

9.3 Preparation of the modelling to compute the synthetic seismograms

As in the previous chapter, to use the modelling code *fullwave2D* for an acoustic FWI procedure, it is necessary the set the modelling parameters.

9.3.1 Discretization in space

I fixed the length of the velocity model to $X = 7\ km$ and the depth to $Z = 1.2\ km$, where the zero level corresponds to the sea-air interface. As modelling grid parameters, I consider a space sampling of $dx = 30m$ (with $dx = dz$), with $nx = 242$ and $nz = 40$, and I use a local order of approximation of the spatial derivative $p_{ord} = p_{loc}$. Figure 9.4 shows the modelling grid.

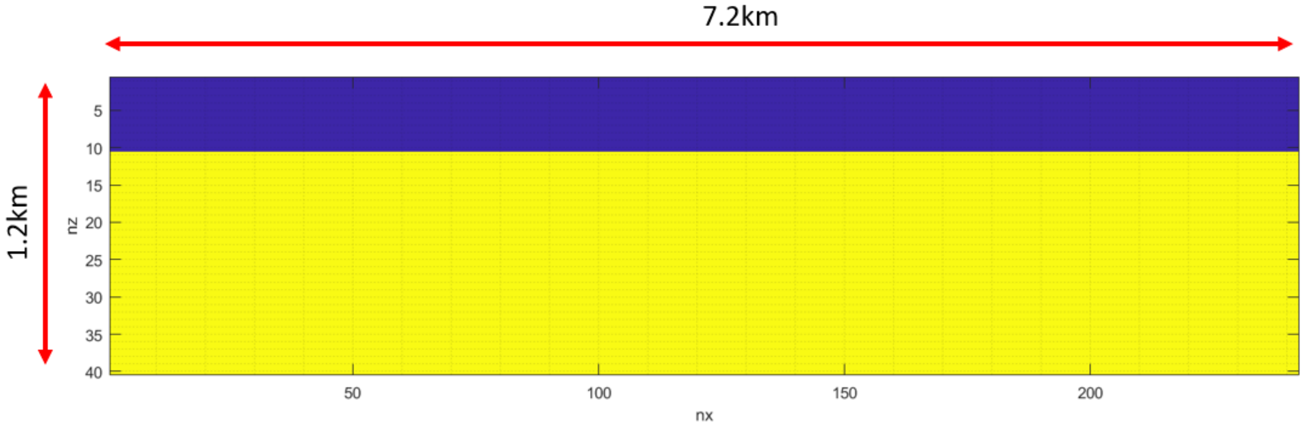


Figure 9.4: Modelling grid with n_x=242, nz=40, and dx=dz=30m. The blue colour area corresponds to the water layer.

9.3.2 Discretization in time

To assure numerical stability, Equation (2.18) must be satisfied, that means $dt \leq 0.0043s$, where I consider a maximum acoustic velocity of $3.5km/s$. Due to this fact, the time sampling is fixed to $dt = 0.004s$, that is the same of the recording sampling of the observed seismograms. Besides, I consider a modelling time of $T = 1.6s$.

9.3.3 Estimation of the seismic wavelet

The source wavelet is estimated using a similar sequence of the previous chapter, but on the sea-bed reflection. Figure 9.5a and Figure 9.5b show the estimated wavelet as a function of time and frequency, respectively.

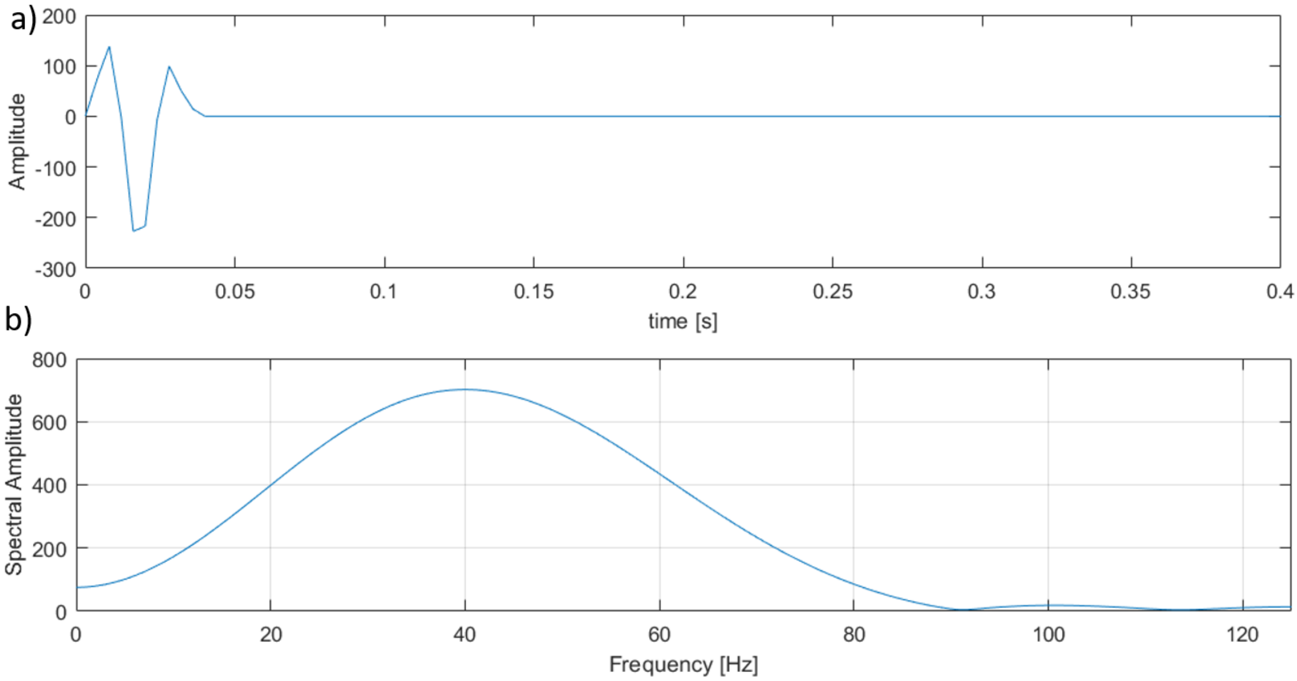


Figure 9.5: Estimated wavelet as a function of a) time and b) temporal frequency.

9.3.4 Source and receivers position

The position of the sources and receivers of all the 56 shots were extracted from the headers of seismic traces. The positions along the x-direction are approximated using the header information relative to the CDP points, whereas the positions along the z-direction are 12.5m, under the sea surface. The corresponding grid node positions are obtained sampling these measures by dx=30m.

9.3.5 Boundary conditions

I consider a reflecting boundary condition for the top side of the model to simulate the sea-air contrast of velocity and density, whereas, for left, right and bottom sides of the model I consider absorbing boundary conditions, with absorbing boundary layers of 30 grid nodes.

9.4 Setting the misfit function and the inversion grid

9.4.1 Misfit function

As misfit function I use the mean of all the L^1-norm difference between the observed and the synthetic seismograms:

$$F(v) = \sum_{s=1}^{56}\left(\sum_{r=1}^{n_{s,r}}\left(\sum_{k=1}^{n_t}\left|G\left(u(v,t_k,\boldsymbol{x}^s,\boldsymbol{x}^{r,s})\right) - G\left(u_0(\boldsymbol{x},t_k,\boldsymbol{x}^s,\boldsymbol{x}^{r,s})\right)\right|\right)dt\right). \tag{9.2}$$

where:

- $n_{s,r}$ is the number of receivers for each source;
- $\boldsymbol{x}^s$ are the positions of the sources;
- $\boldsymbol{x}^{r,s}$ are the positions of the receivers, different for each source;
- t_k are the temporal samples of the seismograms;
- $u_0(\boldsymbol{x},t_k,\boldsymbol{x}^r,\boldsymbol{x}^s)$ are the observed seismograms;
- $u(v,t_k,\boldsymbol{x}^r,\boldsymbol{x}^s)$ are the synthetic seismograms;
- G is the processing operator, as in Equation (9.1).

Note that the operator G is applied to the observed and synthetic seismograms.

9.4.2 Inversion grid

As inversion grid, I make use of a **fine inversion grid**, which is a subset of the modelling grid of Figure 7.3, for a total number of 7260 unknows. Note that the first ten rows of the modelling grid are not included in the fine inversion grid, because we suppose to know the water layer velocity of 1480 km/s up to the seabed at $300m$. In Figure 9.6 the set of red points represents the fine inversion grid.

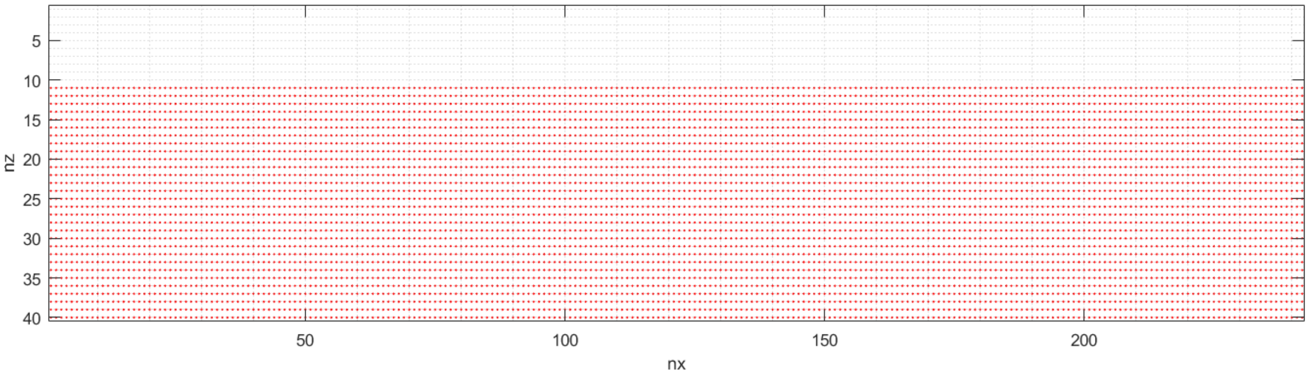


Figure 9.6: Fine inversion grid (the red points) and modelling grid.

9.4.3 Inversion procedure and initial model

As local algorithm I make use of the conjugate gradient method [66], where the descend direction h_k and the step length γ_k are computed as described in Section 7.5. The search domain is very general and consists of the search ranges of the 7260 unknows, that varies from 1300 m/s to 3750 m/s for all the unknowns.

9.4.4 The initial model

We have seen that the initial model ${v_p}^0$ plays an essential role in a highly non-linear inverse problem such as FWI. As initial model, in this test, I have used the velocity model obtained from a previous inversion (Figure 9.7), that makes use of the application of genetic algorithms on a coarse grid. The details and the results can be found in [86] and [10]. The model accuracy is checked through the degree of flattening of the events on the CIGs, obtained after the pre-stack Kirchhoff depth migration (PSDM).

Figure 9.11a shows 11 CIGs evenly spaced along the profile up to 1.2 km of depth (the maximum depth of the modelling grid), computed with the starting model of Figure 9.7. In this figure, a trace-by-trace normalization and a gain function are applied for display purposes. A preliminary alignment of some events can be observed, but the gathers still present complex move-outs that can be reduced using a local FWI based on a gradient line-search method.

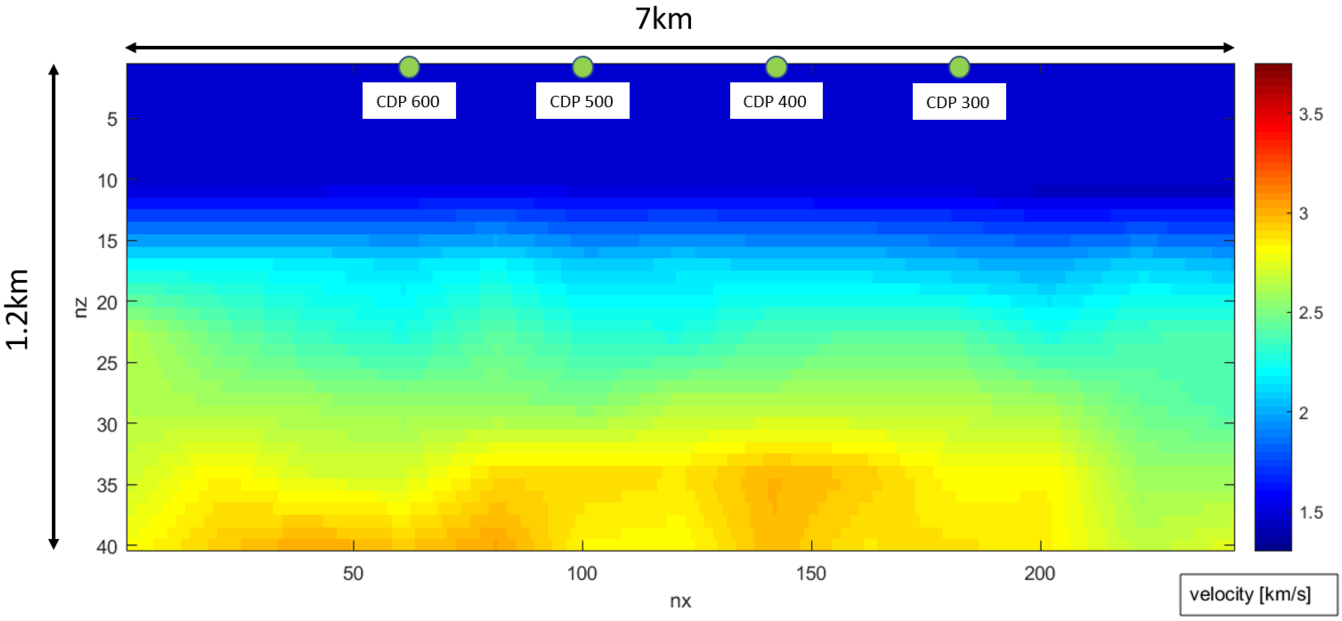


Figure 9.7: Starting model for the local minimization procedure, pre-estimated by a global optimization procedure using the GA.

9.5 Local FWI starting from a pre-estimated model

I perform 200 iterations of the minimization procedure. Figure 9.8 displays the evolution of the data misfit, with a decrease of about 53 % at the last iteration. Figure 9.9 shows a comparison of four vertical velocity profiles obtained at the end of the inversion process, related to four CDP positions along the seismic profile (the green points in Figure 9.7). The long-wavelength structure of the starting model is not significantly changed, except for the upper part just below the seabed where a significant change of the velocity values can be noted. Figure 9.10 shows the difference between the observed and predicted data for two shot gathers before and after the optimization procedure, where the decrease of the differences of the inverted data can be observed.

Finally, Figure 9.11b shows the CIGs obtained by pre-stack depth migrating the data, using the final velocity model of the minimization procedure. Comparing Figure 9.11a and Figure 9.11b, a significant improvement of the horizontal alignment of the events can be noted, especially for the events just below the seabed reflection and located in the central part of the model, between $300m$ and $600m$ of the depth.

9.6 Conclusions and results

In this chapter, I have described an acoustic FWI experience made on a 2D seismic marine data. In particular, I have used an inversion procedure based on a local optimization algorithm, and a specific processing sequence applied either on the observed and the predicted data to reduce the non-linearity of the misfit function. This procedure has allowed us to make the whole procedure more robust against the cycle skipping problem, and applicable at low seismic frequencies where the S/N ratio could be low. Starting from a velocity model obtained by a previous global optimization procedure based on genetic algorithms I have estimated a final model whose quality is assessed by the good correspondence between the predicted and the observed data and by the improvements of the horizontal alignment of the events in the CIGs, obtained by a PSDM Kirchoff depth migration.

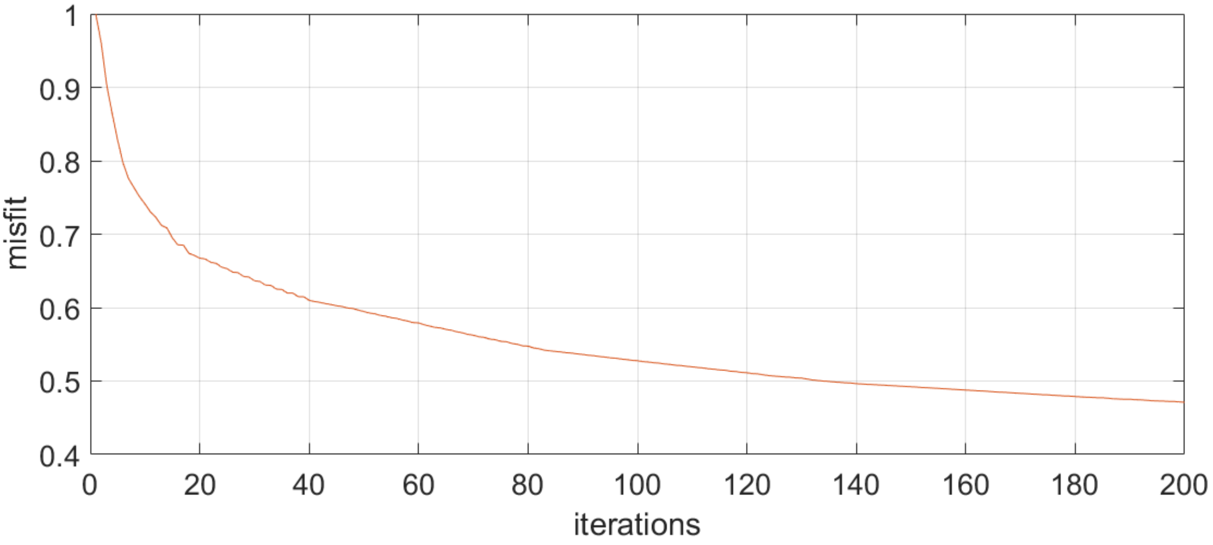


Figure 9.8: Evolution of the data misfit for the local minimization procedures.

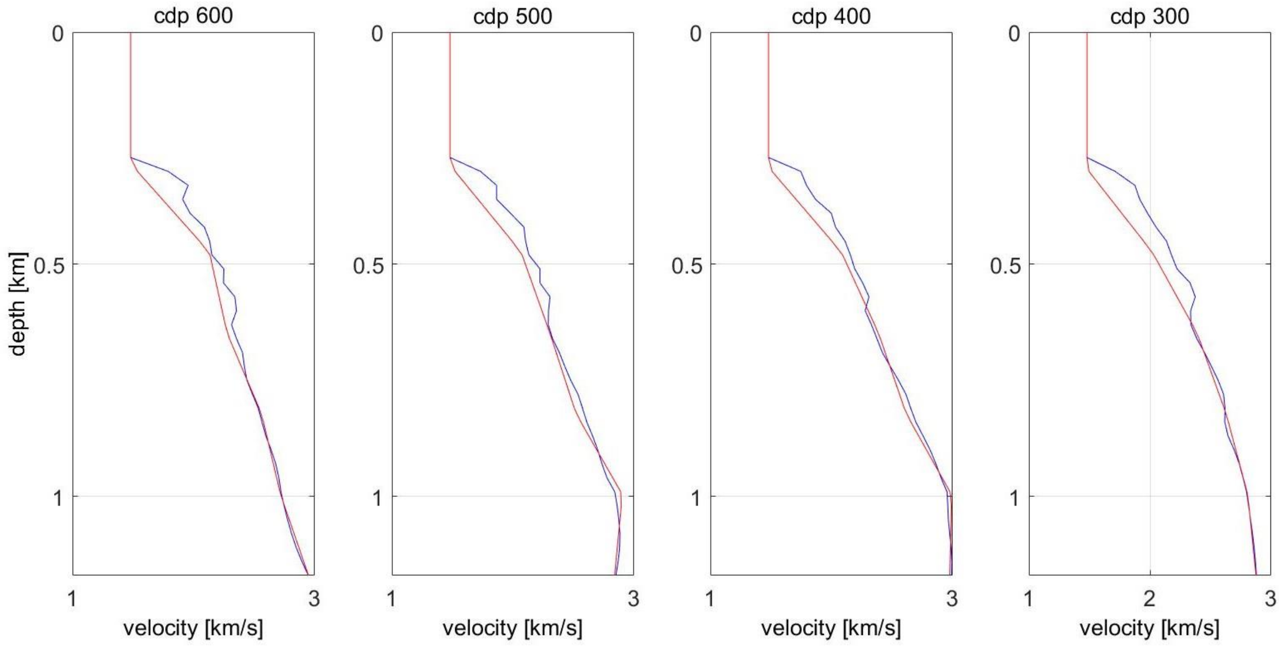


Figure 9.9: Comparison of four vertical velocity profiles: the red curves represent the velocity profiles related to the starting model; the blue curves represent the velocity profiles related to the final model.

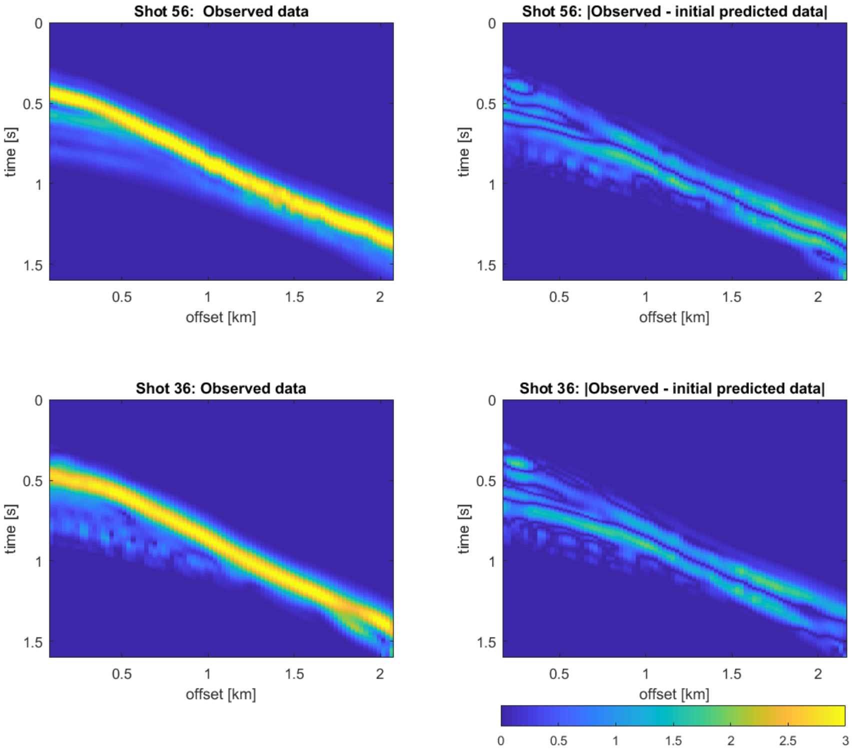


Figure 9.10: Observed data for two shot gathers (left) and the difference between the predicted and the observed data before (center) and after (right) the local inversion.

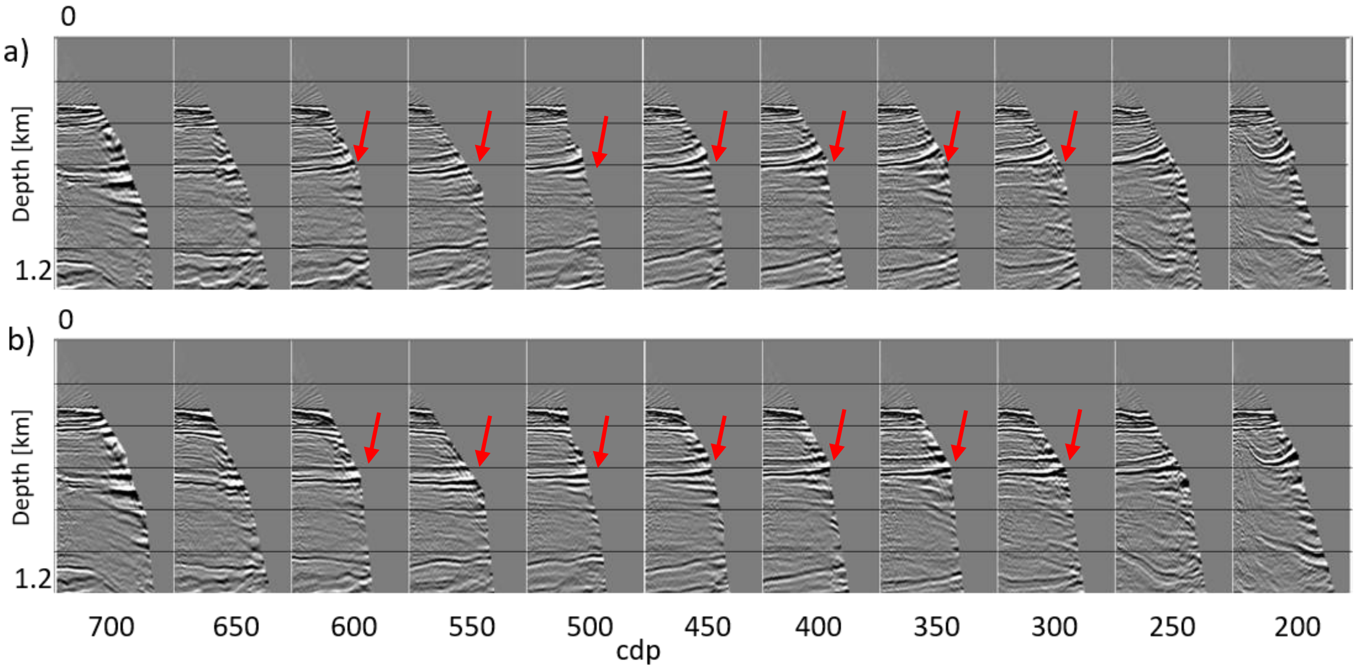


Figure 9.11: CIGs derived from PSDM (Kirchoff) using a) the starting velocity model for the local optimization and b) the final velocity model obtained at the end of the local optimization procedure.

Conclusions

In my Ph.D. I have obtained the following outcomes concerning different topics of the Full Waveform Inversion:

- an algorithm for the solution of the 2D acoustic wave equation to model the predicted seismograms in the context of the seismic exploration. This scheme is characterized by a local order of approximation of the spatial derivatives to reduce the computational time and the approximation error;
- an efficient algorithm to compute the gradient of a misfit function using the well-known adjoint method. This algorithm considers the L^n-norm difference between the synthetic and the observed seismograms as misfit function, and includes also the possibility to execute a simple processing sequence on the synthetic seismograms;
- a comparison of the performances of two global optimization algorithms, the Adaptive Simulated Annealing and the Genetic algorithms, on some analytic test functions representing different scenarios of the misfit function;
- an application of FWI on a classical synthetic seismic inversion problem on the Marmousi model;
- an application of FWI on a 2D seismic line acquired onshore in South Tuscany, characterized by data with a very low S/N ratio;
- an estimation of an acoustic velocity field for a 2D marine seismic profile, using a local FWI procedure, and a specific processing sequence on the seismograms;
- an implementation of the developed algorithms in a tool, called *fullwave2D,* to use in the context of seismic inversion problems.

Nowadays the Full Waveform Inversion represents an important research topic in seismic exploration. In my Ph. D I studied some of the most important aspects concerning the modelling and the inversion techniques, under the hypothesis of a 2D acoustic approximation. The efficient procedures implemented for the solution of the wave equation, the adjoint method, and the global and local optimization techniques allow solving complex FWI problems both on synthetic and real data.

Moreover, the applications of these tools on real data allow testing the reliability and the efficiency of these modelling and inversion techniques, that otherwise would remain unproven working only on synthetic data.

Future steps of my research will be the analysis and the development of new tools in more complex scenarios such as 3D and/or elastic context.

Appendix A. Implementation details of *fullwave2D*

In this appendix, I describe the details of the implementation of the code *fullwave2D*, presented in Chapter 4. In particular, I consider the software and hardware requirements, the code organization, the input and output files, and how to compile and execute the software.

A.1 Software and hardware requirements

1. **Operating system**: the code is designed to run on a generic version of **Linux** or Unix. It has been tested on Ubuntu, Centos and Slackware version of Linux and on the BSD version of Unix that underlies MAC OS X. It is not designed to run under Microsoft Windows. However, it is possible to run the code on Linux version installed on Microsoft Windows **virtual machine**.
2. **Compiler**: the code is written in C++ and has been compiled and tested under the gnu compiler suite *g++* version 4.4. It does not require specific scientific library, such as GNU Scientific Library (GSL), but only the standard libraries of *gcc* compiler. All the names of the libraries can be found in the file *library.h*.
3. **Hardware**: the solver is designed to run on every computer with a generic version of Linux or MAC OS X. It was tested on machines with different type of CPU (*i5-dual core, i7-quadcore, i-Xeon-multicore)* and different RAM sizes (8GB, 16GB, 32GB and 64GB). The clock-speed influences the computational time of the code. The size of the RAM influences the dimension of problems that can be managed.

A.2 Code organization

The code is organized into the following directories:

1. *src/,* that contains all the source codes, that is the *.cpp* files and *.h* file of the code to be compiled by the *Makefile;*
2. *utils/,* that contains some octave codes for utilities;
3. *work_example/,* that contains an example to run the code for the Marmousi model.

The *Makefile* file contains the instructions to compile the code. The README file provides more specific information about the internal structure of the code.

A.2.1 Files in src/

The source code is divided into many *.cpp* files to better organize the specific functions and for clarity reason. Here is a list of the more important file:

- *main.cpp:* this file represents the main function, which starts of the program. It initializes many global parameters, prints the modelling information on terminal, and calls all the other functions;
- *read_runfile.cpp*: this function reads the *Runfile* file, in which the parameters required to run the code are contained;
- *matrixK_acoustic.cpp:* this file contains the functions to compute the stiffness matrix, that is the matrix of finite difference coefficients for the approximation of the spatial derivatives;

- *modelling.cpp:* this file contains the functions that controls the information about the sources, the receivers, and calls the functions to compute the stiffness matrix;
- *receivers.cpp:* this file contains the functions to read the receiver coordinates from the receiver file;
- *source.cpp*: this file contains the functions to read the source coordinates from the source file and to read the source wavelet from the wavelet file;
- *boundary.cpp*: this file contains the functions to build the boundary conditions, reflecting or absorbing ones;
- *model_param_acoustic.cpp*: this file contains the functions to read the velocity model file;
- *read_obs.cpp*: this file contains the function to read the observed seismograms in case of misfit or gradient computation task;
- *read_mute.cpp*: this file contains the function to read the possible mute file to be applied to the observed and/or synthetic seismograms;
- *processing.cpp*: this file contains the function to apply the possible set of processing operations to the observed and/or synthetic seismograms;
- *read_filter.cpp*: this file contains the function to read the possible filter file to be applied to the observed and/or synthetic seismograms;
- *filter_bp.cpp*: this file contains the function to apply the possible filter operation to the observed and/or synthetic seismograms;
- *envelope.cpp*: this file contains the function to apply the possible envelope operation to the observed and/or synthetic seismograms;
- *adjoint_source.cpp*: this file contains the function to compute the adjoint source in case of the computation of the gradient of a misfit function related to the velocity model;
- *forward.cpp*: this file contains the function for the computation of the predicted seismograms;
- *misfit.cpp*: this file contains the function for the computation of the misfit between the predicted and the observed one;
- *gradient_acoustic.cpp*: this file contains the function to compute the gradient of a misfit function related to the velocity model.

In the *library.h* file there are the declarations of all the C++ libraries necessary to run the code, the prototypes of all the *c.pp* functions.

A.2.2 File in utils/

In this directory, there are some useful functions, written in Matlab code, to build the file necessary to run the code:

1. *write_source(source,name_source)*: this function builds the wavelet file with a name specified by *name_source*, and one or more wavelets specified by the struct *source*;
2. *write_Vp(V, V_name):* this function builds the velocity model with a name specified by *V_name* and the 2D model specified as a matrix *V;*

3. *seismogram=read_seismogram(name_file)*: this function reads the synthetic seismograms, whose name is specified by *name_file* and returns as output a structure, formed by the different seismograms, organized as matrixes;
4. *build_filter(name_file,nt,dt,[fl,fh])*: this function builds a filter file, whose name is specified by *name_file*, which has a sampling time of *dt*, a number of samples of *nt* and a bandpass between *fl* and *fh*.

More details about these functions can be found in the first lines of the code of each function

A.2.3 File in work_example/

In this directory, there is an example of working directory for the code.

1. *MarmousiOrig.bin* is a binary file representing the velocity model, that is the Marmousi model ($nx = 384, nz = 122$);
2. *MarmousiSmooth* is a binary file representing a smooth version of the Marmousi model;
3. *Source.txt* and *Receivers.txt* are text files, representing the acquisition layout of 2 sources and 384 receivers;
4. *Wavelet.bin* is a binary file, representing the wavelet, that is a Ricker function with a peak frequency of $6Hz$;
5. *Runfile.txt* is a text file, in which the information about the modelling parameters are specified ($dx = 24m, dt = 0.002, T = 3s$, etc…).

A.3 Input files

A.3.1 The Runfile file

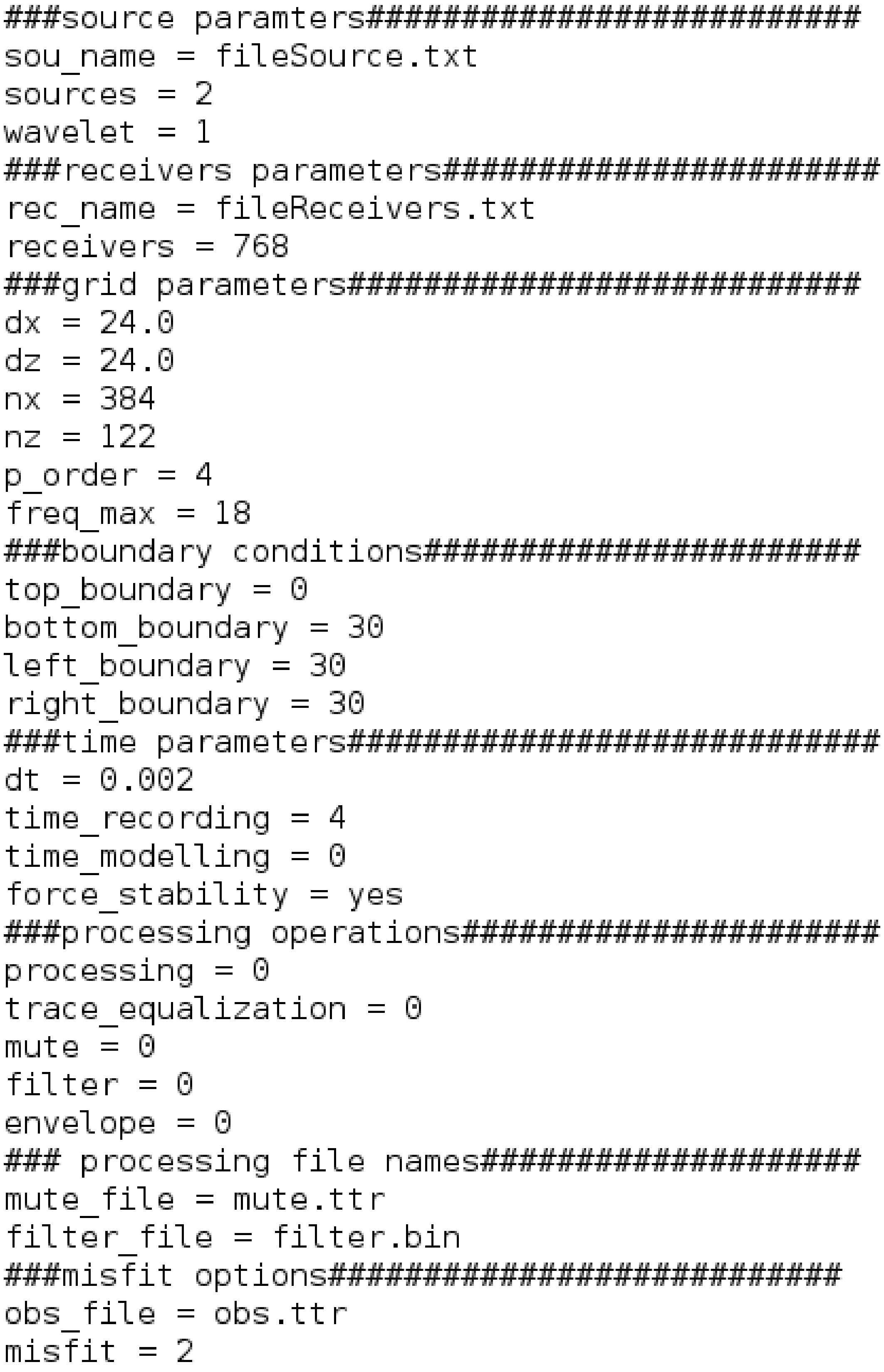
The runfile is a text file that specifies the modelling parameters to run the code and the possible processing operations to run on the synthetic data. The runfile file for Marmousi model, located in the subdirectory *work_example/*, is sketched in Figure A.1.

Each command line is organized as "*name_command = value_of_command"*. A blank space before and after the equal sign is necessary to ensure the correct reading of the runfile by the code.

The runfile supports the following parameters:

- *Source_name*: name of the source file;
- *Source:* number of sources, specified in the source file;
- *Wavelet:* number of wavelets, specified in the wavelet file. The value of this parameters can be 1 or equal to the number of sources. In the first case, there is a unique wavelet for all the sources; In the second case, there are many wavelets as many sources;
- *Receivers_name*: name of the receivers file:
- *Receivers:* number of total receivers, that is the sum of all the numbers of receivers of all the seismograms;
- *dx:* space sampling along the length;
- *dz:* space sampling along the depth;
- *nx:* number of grid nodes along the length;
- *nz:* number of grid nodes along the depth;

- *p-order:* order of approximation of the spatial derivatives, that can be an even number between 2 and 24 or "*optimized",* if you want to use a local order of approximation of the spatial derivatives*;*
- *freq_max:* specifies the highest frequency considered during the modelling (this parameter is active only if *p-order = optimized)*;
- *top_boundary:* number of boundary layers at the top of the model. 0=free surface; >0 absorbing layers with specified thickness in grid points;
- *bottom_boundary:* number of boundary layers at the bottom of the model. 0=free surface; >0 absorbing layers with specified thickness in grid points;
- *left_boundary:* number of boundary layers at the left of the model. 0=free surface; >0 absorbing layers with specified thickness in grid points;
- *right_boundary:* number of boundary layers at the right of the model. 0=free surface; >0 absorbing layers with specified thickness in grid points;
- *dt:* time sampling in seconds;
- *time_recording:* recording time in seconds;
- *time_modelling:* if 0 modelling time is equal to the recording time; if >0 the modelling time is equal to the length of the wavelet/wavelets;
- *force_stability:* if >0 code forces the numerical stability, subsampling the time sampling until stability condition is reached;
- *processing:* if 0 the processing operations are performed only on the synthetic data; if 1 also on the observed data;
- *trace_equalization:* if 0 no trace equalization, if $n > 0$ then the trace equalization is applied to the synthetic and/or observed seismograms as nth processing operation;
- *mute:* if 0 no mute operation, if $n > 0$ then the mute operation is applied to the synthetic and/or observed seismograms as nth processing operation;
- *filter:* if 0 no filter operations, if $n > 0$ then the filtering is applied to the synthetic and/or observed seismograms as nth processing operation;
- *envelope:* if 0 no envelope, if $n > 0$ then the envelope operation is applied to the synthetic and/or observed seismograms as n^{th} processing operation;
- *mute_file:* name of the muting file;
- *filter_file:* name of the filtering file;
- *obs_file:* name of the observed seismograms file;
- *misfit:* vector norm used for the misfit computation between the observed and the synthetic seismograms;

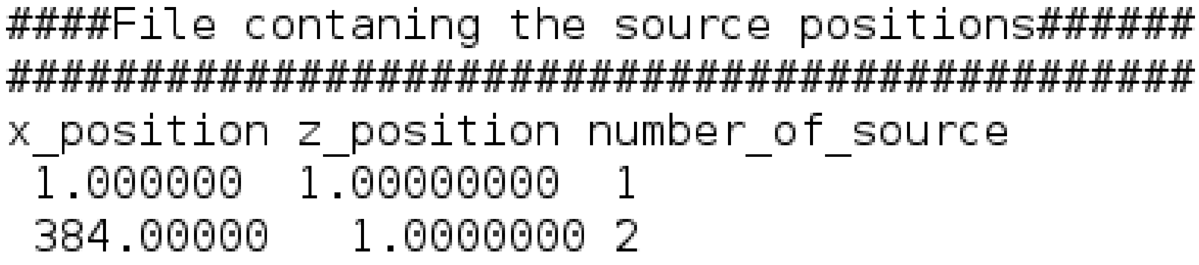


```
###source paramters##########################
sou_name = fileSource.txt
sources = 2
wavelet = 1
###receivers parameters#######################
rec_name = fileReceivers.txt
receivers = 768
###grid parameters###########################
dx = 24.0
dz = 24.0
nx = 384
nz = 122
p_order = 4
freq_max = 18
###boundary conditions#######################
top_boundary = 0
bottom_boundary = 30
left_boundary = 30
right_boundary = 30
###time parameters############################
dt = 0.002
time_recording = 4
time_modelling = 0
force_stability = yes
###processing operations######################
processing = 0
trace_equalization = 0
mute = 0
filter = 0
envelope = 0
### processing file names####################
mute_file = mute.ttr
filter_file = filter.bin
###misfit options###########################
obs_file = obs.ttr
misfit = 2
```

Figure A.1: Sketch of the Runfile for Marmousi model, located in the subdirectory work_example.

A.3.2 The Source file

The information about the position of the sources on the modelling grid are contained in a text file, whose name is specified in the *runfile* file. The source file, located in *work_example/* is sketched in Figure A.2.



```
####File contaning the source positions######
############################################
x_position z_position number_of_source
 1.000000  1.00000000  1
 384.00000   1.0000000 2
```

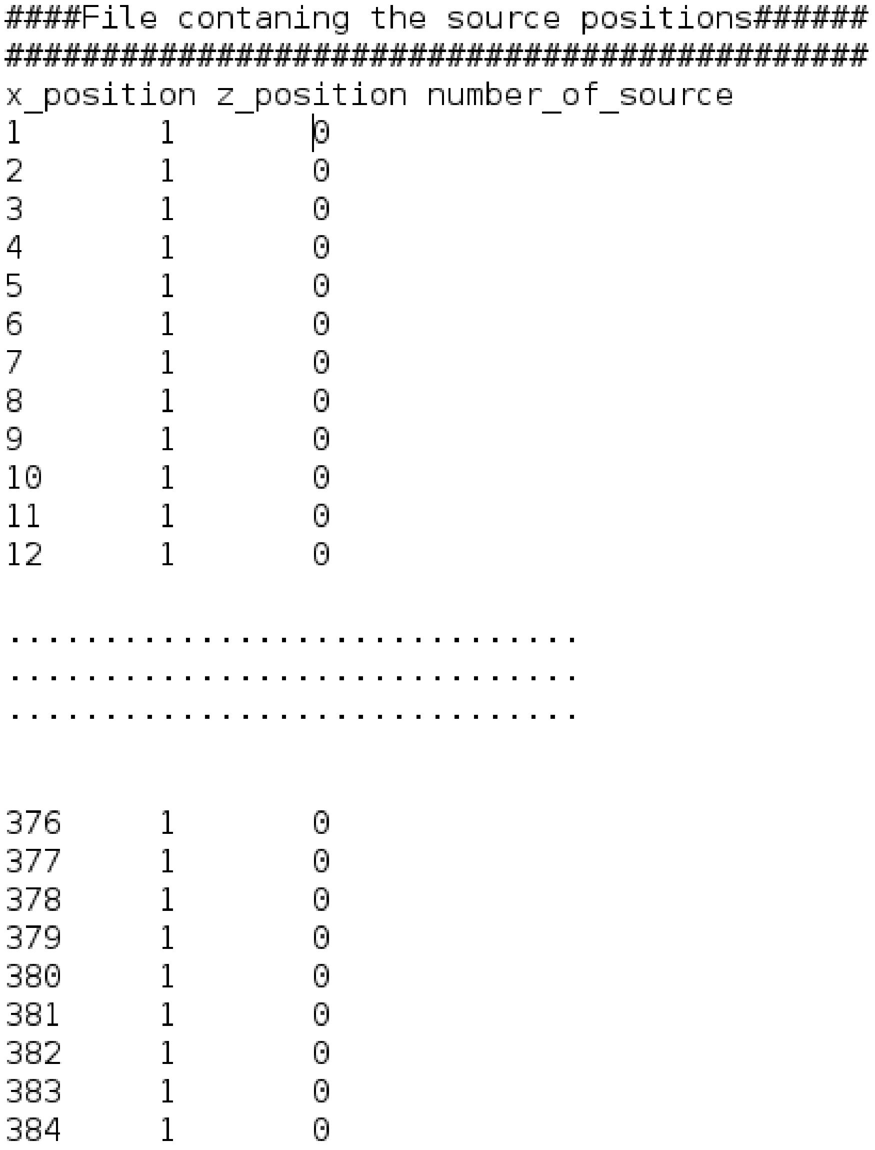
Figure A.2: Sketch of the Source file, located in the subdirectory work_example.

The first column contains the position of the sources along the length of the grid. The second column contains the position of the sources along the depth of the grid. The third column contains the number that identify each source. The code checks if the positions of the sources are consistent with the dimension of the grid and if the number of sources in the source file is consistent with the number of sources declared in the runfile.

A.3.3 The receiver file

The information about the position of the receivers on the modelling grid are contained in a text file, whose name is specified in the *runfile* file. The receiver file, located in *work_example* is sketched in Figure A.3. The first column contains the position of the receivers along the length of the model. The second column contains the position of the receivers along the depth of the model. The third column contains the number of source to which each receiver belongs.

The code checks if the positions of the receivers are consistent with the dimension of the grid and if the total number of receivers in the receiver file is consistent with the number of receivers declared in the runfile.



```
####File contaning the source positions######
##############################################
x_position z_position number_of_source
1       1       0
2       1       0
3       1       0
4       1       0
5       1       0
6       1       0
7       1       0
8       1       0
9       1       0
10      1       0
11      1       0
12      1       0

..............................
..............................
..............................


376     1       0
377     1       0
378     1       0
379     1       0
380     1       0
381     1       0
382     1       0
383     1       0
384     1       0
```

Figure A.3: Sketch of the Receiver file, located in subdirectory work_example.

A.3.4 The velocity model file

The information about the velocity model is contained in a binary file, whose name is specified during the execution of the command. The byte structure is organized as follow: the first eight byte are reserved to the dimension of model, nx and nz,stored as integers. Figure A.4 shows a sketch of the organization of the velocity model file. Then, the model samples are stored for rows as doubles. The code checks that the model sizes are correct and that the maximum velocity in the model is consistent with the numerical stability condition.

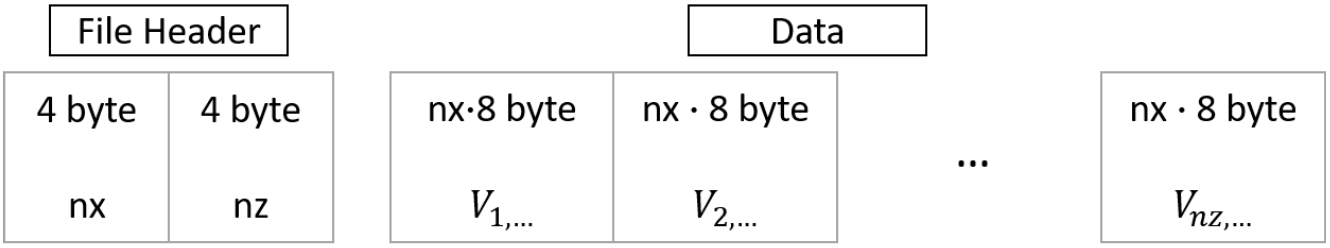


Figure A.4: Sketch of the organization of the velocity model file.

A.3.5 The wavelet file

The information about the wavelet is contained in a binary file, whose name is specified during the execution of the command (see on the Section A.5). The byte structure can be of two types:

- in case of single wavelet for all the sources, first the length of the wavelet is specified as integer; then the wavelet samples are stored as double;
- in case of the number of wavelet is equal to the number of sources, first each length of each wavelet is specified as integer; then each wavelet is stored as double;

The code checks that the dimensions of the wavelets are consistent.

A.3.6 Some other files

In case of a misfit or gradient computation problem, a file of the observed seismograms must be present with the name specified in the runfile. The format and the dimension of this file must be the same of one of the synthetic seismograms. The code checks if its dimensions are correct.

In case of a muting operation on the synthetic and /or observed seismograms, a muting file must be present with the name specified in the runfile. The format and the dimension of this file must be the same of one of the synthetic seismograms. The code checks if its dimensions are correct.

In case of a filtering operation on the synthetic and /or observed seismograms, a filter file must be present with the name specified in the runfile. The build of the filter file can be done using the function *build_filter* located *utils*.

A.4 The output file

The output data, whose name is specified during the execution of the command, is contained in an output file that depends on the type of problem the code has solved.

In case of a modelling problem, the output file is formed by all the synthetic seismograms, stored in a binary format used by the code, very similar to the format of a *sgy* file, and organized as follow: first the information about the acquisition layout is specified in a *File Header*, then all the seismic traces are stored as float, starting from the ones of the first source, and ordered as in the receiver file. In the *File Header* first the number of sources, receivers and recording samples are stored as integer. Then the recording sampling is stored as a float.

In case of a misfit problem, the output file is a binary file, that contain the value of the misfit, stored as a double.

In case of a gradient computation, the output file is a binary file, with a byte structure like the structure of the velocity model, with the values of the gradient in place of the value of the velocities.

A.5 Compiling and running the fullwave2D code

1. **Compilation**: compiling the code is done simply by typing *make* on a terminal in the main directory. This command executes the *Makefile file*, compiling all the *.cpp* files contained in the *src* directory and producing a binary file named *fullwave* on the main directory. The removal the binary file is done by typing *make clean*.
2. **Execution**: the code is run with the command:

   ./fullwave type_of_problem runfile velocity_model wavelet output,

   with:

   - *Type_of_problem* is the type of problem, that is *modelling*, *misfit* or *gradient;*
   - *Runfile* is the name of the runfile;
   - *Velocity_model* is the name of the velocity model;
   - *Wavelet* is the name of source wavelet;
   - *Output* is the name of the output file of the code, that can be the seismograms, the misfit value or the gradient of a misfit function for the *velocity_model*.

List of Figures

List of Tables

References

[1] O. Yilmaz, Seismic Data Analysis: Processing, Inversion and Interpretation of Seismic Data, Society of exploration geophysicists, 2001.

[2] R. E. Sheriff and L. P. Geldart, Exploration seismology, Cambridge University Press, 1995.

[3] "The Open University, Walton Hall, Milton Keynes, MK7 6AA," The Open University, Walton Hall, Milton Keynes, MK7 6AA, [Online]. Available: http://www.open.edu/openlearn/.

[4] A. Tarantola, "A strategy for nonlinear elastic inversion of seismic reflection data," *Geophysics,* vol. 51, pp. 1893-1903, 1986.

[5] J. Virieux and S. Operto, "An overview of full waveform inversion in exploration geophysics," *Geophysics,* vol. 74, pp. 127-152, 2009.

[6] A. Fichtner, Full Waveform Modelling and Inversion, Berlin: Springer-Verlag, 2010.

[7] J. Tromp, C. Tape and Q. Liu, "Seismic tomography, adjoint methods, time reversal, and banana-doughnut kernels.," *Geophysical Journal International,* vol. 160, pp. 195-216, 2005.

[8] E. Bozdag, J. Trampert and J. Tromp, "Misfit functions for full waveform inversion based on instantaneous phase and envelope measurements," *Geophysical Journal International,* vol. 185, pp. 845-870, 2011.

[9] H. Igel, H. Djikpéssé and A. Tarantola, "Waveform inversion of marine reflection seismograms for P impedance and Poisson's ratio," *Geophysical Journal International,* pp. 363-371, 1996.

[10] A. Mazzotti, N. Bienati, E. Stucchi, A. Tognarelli and A. Sajeva, "Two-grid genetic algorithm full-waveform inversion.," *The Leading Edge,* vol. 35, pp. 1068-1075, 2017.

[11] J. Nocedal and S. Wright, Numerical Optimization, New York: Springer-Verlag, 2006.

[12] P. Wolfe, "Convergence Conditions for Ascent Methods.," *SIAM Review (Society of Industrial and Applied Mathematics),* vol. 11, pp. 226-235, 1969.

[13] W. B. Beydon and A. Tarantola, "First Born and Rytov approximations: Modeling and inversion conditions in a canonical example," *The Journal of the Acoustical Society of America,* vol. 83, pp. 1045-1055, 1988.

[14] B. Galuzzi, E. Zampieri and E. Stucchi, "A Local Adaptive Method for the Numerical Approximation in seismic wave modelling," *Communication in Applied and Industrial Mathematic (CAIM),* 2017.

[15] K. Aki and P. G. Richards, Quantitative Seismology, University Science Books, 2002, p. 704.

[16] G. Ranalli, Rheology of the Earth, Springer Netherlands, 1995.

[17] J. M. Reynolds, An Introduction to Applied and Environmental Geophysics, Chichester: John Wiley and Sons, 2011.

[18] N. Bleinstein, "Two-and-One-Half Dimensional in-Plane Wave Propagation," *Geophysical Prospecting,* pp. 686-703, 1986.

[19] H. Ryan, “Ormsby, Klauder, Butterworth-a choice of wavelets,” *CSEG Recorder,* vol. 19, no. 7, pp. 8-9, 1994.

[20] B. Y. Yi, G. H. Lee, H. J. Kim, H. Jou, D. G. Yoo, B. J. Ryu and K. Lee, “Comparison of wavelet estimation methods,” *Geosciences Journal,* vol. 17, pp. 55-63, 2013.

[21] A. R. Mitchell and D. F. Griffiths, The Finite Difference Method in Partial Differential Equations, New York: John Wiley & Sons, 1980.

[22] J. C. Strickwerda, Finite differences schemes and partial differential equations, The Wadsworth & Brooks/Cole Mathematics, 1989.

[23] K. W. Morton and D. F. Mayers, Numerical solution of partial differential equations: an introduction, Cambridge: Cambridge University Press, 2005.

[24] B. Fornberg, “Generation of finite difference formulas on arbitrarily spaced grids,” *Mathematics of Computation,* vol. 51, pp. 699-706, 1988.

[25] R. Courant, K. Friedrichs and H. Lewy, “On the Partial Difference Equations of Mathematical Physics,” *IBM Journal,* vol. 11, pp. 215-234, 1967.

[26] R. M. Alford, K. R. Kelly and D. M. Boore, “Accuracy of finite-difference modeling of the acoustic wave equation,” *Geophysics,* vol. 39, no. 6, pp. 834-842, 1974.

[27] B. Galuzzi, E. Stucchi and E. Zampieri, “Optimal parameters for finite difference modeling of 2D,” in *Atti del 34-esimo Convegno del Gruppo Nazionale di Geofisica della Terra Solida,* Trieste, 2015.

[28] G. J. Hicks, “Arbitrary source and receiver positioning in finite-difference schemes using Kaiser windowed sinc functions,” *Geophysics,* vol. 67, pp. 156-165, 2002.

[29] J. F. Keiser, “Nonrecursive digital filter design using the I-sinh window function,” in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS’74),* 1974.

[30] F. Bowman, Introduction to Bessel Functions, New York: Dover Publications , 1958, pp. 1-52.

[31] A. Cerjan, D. Kosloff, R. Kosloff and M. Reshef, “A non reflecting boundary condition for discrete acoustic and elastic wave euqtion,” *Geophysics,* vol. 50, pp. 705-708, 1985.

[32] J. Lions and E. Magenes, Nonhomogeneous boundary value problems and applications, Berlin: Springer-Verlag, 1972.

[33] G. Chavent, “Identification of functional parameters in partial differential equations,” in *Joint Automatic Control Conference,* 1974.

[34] P. Lailly, “The seismic inverse problem as a sequence of before stack migration,” in *Conference of Inverse Scattering. Theory and Application,* Philadelphia, 1983.

[35] A. Tarantola, “Inversion of seismic reflection data in the acoustic approximation,” *Geophysics,* pp. 1259-1266, 1984.

[36] O. Gauthier, J. Virieux and A. Tarantola, “Two-dimensional nonlinear inversion of seismic waveforms: numerical results,” *Geophysics,* vol. 51, no. 7, pp. 1387-1403, 1986.

[37] R. Plessix, “A review of the adjoint-state method for computing the gradient of a functional with geophysical applications.,” *Geophysical Journal International,* pp. 495-503, 2006.

[38] A. Fichtner, B. N. Kennet, H. Igel and H.-P. Bunge, “Full seismic waveform tomography for upper-mantle structure in the Australasian region using adjoint methods,” *Geophysical Journal International,* vol. 179, pp. 1703-1725, 2009.

[39] D. Peter, D. Komatitsch, Y. Luo, R. Martin, N. Le Goff, P. Le Loher, F. Magnoni, Q. Liu, C. Blitz, T. Nissen-Meyer, P. Basini and J. Tromp, “Forward and adjoint simulations of seismic wave propagation on fully unstructured hexahedral meshes,” *Geophysical Journal International,* pp. 721-739, 2011.

[40] D. Komatitsch, Z. Xie, E. Bozdag, E. Sales de Andrade, D. Peter, Q. Liu and J. Tromp, “Anelastic sensitivity kernels with parsimonious storage for adjoint tomography and full waveform inversion,” *Geophysical Journal International,* vol. 206, pp. 1467-1478, 2016.

[41] T. Weise, Global Optimization Algorithms – Theory and Application, Self-Published, 2009.

[42] A. H. Land and A. G. Doig, “An automatic method of solving discrete programming problems,” *Econometrica,* vol. 28, no. 3, pp. 497-520, 1960.

[43] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, “Optimisation by simulated annealing,” *Science,* pp. 671-680, 1983.

[44] A. Corana, M. Marchesi, C. Martini and S. Ridella, “Minimizing multimodal functions of continuous variables with the “simulated annealing” algorithm,” *ACM Transactions on Mathematical Software (TOMS),* vol. 13, no. 3, pp. 262-280, 1987.

[45] W. K. Hastings, “Monte Carlo sampling methods using Markov Chains and their applications,” *Biometrika,* pp. 97-109, 1970.

[46] H. Aguiar, O. Junior, L. Ingber, A. Petraglia, M. Rembold Petraglia and M. Augusta Soares Machado, Stochastic global optimization and its applications with fuzzy adaptive simulated annealing, Springer-Verlag Berlin Heidelberg, 2012.

[47] L. Ingber, “Adaptive simulated annealing (ASA): Lesson learned,” *Control and Cybernetics,* vol. 25, no. 1, pp. 33-54, 1996.

[48] L. Ingber, “Very fast simulated re-annealing,” *Mathematical and Computer Modelling,* vol. 12, no. 8, pp. 967-973, 1989.

[49] J. H. Holland, Adaptation in natural and artificial system: An introductory analysis with applications to biology, control and artificial intelligence, University Michigan Press, 1975.

[50] D. Schlierkamp-Voosen and H. Muhlenbein, “Predictive models for the breeder genetic algorithm,” *Evolutionary Computation,* vol. 1, no. 1, pp. 25-49, 1993.

[51] M. Mitchell, An introduction to the genetic algorithms, MIT press, 1998.

[52] S. N. Silvanandam and S. N. Deepa, Introduction to Genetic Algorithms, Springer Berlin Heidelberg., 2008.

[53] C. Z. Janokow and Z. Michalewicz, “An experimental comparison of binary and floating point representations in genetic algorithms,” in *Proceedings of international conference on genetic algorithms,* 1991.

[54] T. Back and F. Hoffmeister, “Extended selection mechanisms in genetic algorithms,” in *Proceddings of the Fourth International Conference on Genetic Algorithms*, 1991.

[55] J. E. Baker, “Reducing Bias and Inefficiency in the Selection Algorithm,” in *Proceedings of the Second International Conference on Genetic Algorithms and their Application*, Cambridge, Massachusetts, USA , 1987.

[56] A. Sajeva, M. Aleardi, B. Galuzzi, E. Stucchi, E. Spadavecchia and A. Mazzotti, “Comparing the performances of four stochastic optimisation methods using analytic objective functions, 1D elastic full-waveform inversion, and residual static computation,” *Geophysical Prospecting,* 2017.

[57] A. E. Eiben and T. Back, “Empirical investigation of multiparent recombination operators in evolution strategies,” *Evolutionary Computation,* vol. 5, no. 3, pp. 347-365, 1997.

[58] K. D. De Jong, “An analysis of the behavior of a class of genetic adaptive systems.,” *PhD thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor,* 1975.

[59] H. H. Rosenbrock, “An automatic method for finding the greatest or least value of a function,” *The Computer Journal,* vol. 3, no. 3, pp. 175-184, 1960.

[60] L. A. Rastrigin, “Extremal Control Systems,” *Theoretical Foundations of Engineering Cybernetics Series,* vol. 3, 1974.

[61] H. P. Schwefel, Numerical Optimization of Computer Models, New York, NY, USA : John Wiley & Sons, Inc., 1981.

[62] B. Galuzzi, E. Zampieri and E. Stucchi, “Global optimization procedure to estimate a starting velocity model for local Full Waveform Inversion,” in *Optimization and Decision Science: Methodologies and Applications,* Springer International Publishing, 2017.

[63] R. Versteeg and P. Lailly, “EAEG workshop report: Practical aspects of seismic data inversion,” *First Break,* vol. 9, no. 2, pp. 75-80, 1991.

[64] R. Versteeg, “The Marmousi experience: Velocity model determination on a synthetic complex data set,” *The Leading Edge,* vol. 13, no. 9, pp. 927-936, 1994.

[65] G. Verrier and F. Castello Branco, “La fosse Tertiare et le gisament de Quenguela-Nord (Bassin du Cuanza),” *Revue de L'Institut Francais du Petrole,* vol. 27, pp. 51-72, 1972.

[66] R. Fletcher and C. M. Reeves, “Function minimization by conjugate gradients,” *The Computer Journal,* vol. 7, no. 2, pp. 149-154, 1964.

[67] M. R. Hestenes and E. Stiefel, “Methods of conjugate gradients for solving linear systems,” *Journal of Research of the National Bureau of Standards* , pp. 409-436, 1952.

[68] B. Galuzzi, A. Tognarelli, E. Stucchi and A. Mazzotti, “Stochastic FWI on Wide-angle Land Data with Different Order of Approximation of the 2D Acoustic Wave Equation,” in *78th EAGE Conference and Exhibition 2016,* Vienna, 2016.

[69] F. Bizzi, B. Galuzzi, A. Tognarelli and E. Stucchi, “Experience of acoustic FWI on seismic Land data,” in *Atti del 35-esimo Convegno del Gruppo Nazionale di Geofisica della Terra Solida,* Lecce, 2016.

[70] P. Elter, G. Giglia, M. Tongiorgi and L. Trevisan, “Tensional and compressional areas in the recent (Tortorian to Present) evolution of the Northern Appenines,” *Bollettino di Geofisica Teorica ad Applicata,* pp. 3-18, 1975.

[71] F. Mongelli, G. Zito , N. Ciaranfi and P. Pieri, “Interpretation of heat flow density of the Apenine chain, Italy,” *Tectonophysics,* vol. 164, pp. 267-280, 1989.

[72] F. A. Decandia, A. Lazzarotto, D. Liotta, L. Cernobori and R. Nicolich, “The CROP03 traverse: insights on post-collisional evolution of northern Appennines,” *Memorie della Società Geologica Italiana,* pp. 427-439, 1998.

[73] M. Boccaletti and G. Guazzone, “Gli archi appenninici, il mare Ligure ed il Tirreno nel quadro della Tettonica dei Bacini margini di retro-arco,” *Memorie della Società Geologica Italiana,* pp. 201-216, 1972.

[74] L. Carmignani and R. Kligfield, “Crustal extension in the northern Appenines: the transition from compression to extension in the Alpi Apuane core complex,” *Tectonics,* pp. 1275-1303, 1990.

[75] F. Batini, G. Cameli, A. Lazzarotto and D. Liotta, “Line CROP18: Southern Tuscany,” *Memorie descrittive Carta Geologica d'Italia,* pp. 137-144, 2003.

[76] L. Bertelli, A. Mazzotti and S. Persoglia, “Planning, acquisition and processing of the CROP seismic data:,” *Memorie Descrittive Carta Geologica d'Italia,* pp. 47-54, 2003.

[77] D. Scrocca, CROP Atlas: seismic reflection profiles of the Italian crust, Ist. Poligrafico e Zecca dello Stato, 2003.

[78] I. R. Finetti, Deep Seismic Exploration of the Central Mediterranean and Italy, Elsevier, 2005.

[79] E. Stucchi, L. Zanzi and A. Mazzotti, “Lessons learned from the acquisition and the processing of the CROP C-ALPS/b, CROP03, CROP04, CROP18 NVR and expanding spread seismic data.,” *Memorie Descrittive Carta Geologica d'Italia,* pp. 75-88, 2003.

[80] F. Bizzi, *Esperienze di Full Waveform Inversion acustica su dati sismici terrestri,* Pisa, 2016.

[81] R. Bazzichi, “Tomografia dai primi arrivi della linea sismica CROP18a (Toscana meridionale),” Pisa, 2016.

[82] A. Brogi, A. Lazzarotto, D. Liotta, G. Ranalli and CROP18 working group, “Crustal structures in the geothermal areas of southern Tuscany (Italy): Insight from the CROP 18 deep seismic reflection lines.,” *Journal of Vulcanology and Geothermal Research,* pp. 60-80, 2005.

[83] G. Bertini, M. Casini, G. Giannelli and E. Pandeli, “Geological structures of a long-living geothermal system, Larderello, Italy,” *Terra Nova,* pp. 163-169, 2006.

[84] G. Bertini, M. Casini, G. Gianelli and E. Pandeli, “Geological Structure of a long-living geothermal system, Larderello, Italy,” *Terra Nova,* vol. 18, no. 3, pp. 163-169, 2006.

[85] B. Galuzzi, A. Tognarelli and E. Stucchi, “Experience of FWI on marine seismic data using a robust op-timization procedure,” in *Atti del 36-esimo Convegno del Gruppo Nazionale di Geofisica della Terra Solida*, Trieste, 2017.

[86] A. Tognarelli, E. Stucchi, N. Bienati, A. Sajeva, M. Aleardi and A. Mazzotti, “Two-grid Stochastic Full Waveform Inversion of 2D Marine Seismic Data,” in *77th EAGE Conference and Exhibition*, London, 2015.

[87] H. Mühlenbein and D. Schlierkamp-Voosen, “Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization,” *Evolutionary Computation,* vol. 1, no. 1, pp. 25-49, 1993.