Acoustic inversion of velocity fields using evolutionary algorithm

Marcos Reinan de Assis Conceição* (DGf-IGEO-UFBA), Átila Saraiva Quintela Soares (DGf-IGEO-UFBA) & Wilson Mouzer Figueiró (CPGG-UFBA)

Abstract

This work attempts to answer the question: “is it possible to invert seismic geophysical data by using differential evolution algorithms?”. Such algorithms compose a subclass of the more general set of evolutionary optimization algorithms, which implement several mechanisms inspired by biological evolution in order to minimize an objective function previously defined. As a recent method study, this work attempts to perform an inversion on a synthetic data, therefore goes through steps of the P-wave equation modeling process. Finally, four synthetic seismograms were successfully inverted through an implementation of the differential evolution algorithm.

Keywords: Evolutionary Algorithms, Differential Evolution, Seismic Inversion, Finite-difference Method, Accoustic Wave Equation, Seismic Velocities, Synthetic Seismograms, Frobenius Norm.

1 Introduction

This work's goal is to solve a inversion problem from synthetic seismogram data generated by a punctual source. This modeling was then be used to generate four 3-layered parallel-plane horizontal seismic models' seismograms. Those will serve as input to the final problem this work attempted to solve: recovering the models' parameters which gave origin to each of the seismic data modeled — task that was done by an evolutionary algorithm.

The modeling of seismograms was made by appling the finite-difference method solving the 2D acoustic wave equation over an arbitrary seismic velocity field. The basic reference for this part can be found on Santos (2002) and Andrade (2002).

With four synthetic seismograms, one for each model, it was time to invert the data: to find the original models capable of reproducing the seismograms calculated. This part was made by using a differential evolution algorithm — a subclass of the evolutionary algorithms class that base themselves on evolution in order to find an optimized solution for difficult non-linear algebra problems like most seismic inversion ones. Theoretical basis for implementing a differential evolution algorithm can be found on Storn and Price (1997). This work was also based on the works of Whitley (1994), which talks more openly about evolutionary computation, and Soares (2018), where the author uses differential evolution in order to solve his media porosity equation parameters.

It is important to use the inversion method proposed here among synthetic seismic data, once the inverted parameters can be compared to the true seismogram generator model ones, as they were known since the beginning. This is a first step in order to invert real seismic data to get real subsurface information.

2 Methodology

The methodology used in this work are strongly based on the fundamentals of inversion theory. The adopted forward modeling was the P-wave propagation equation, which was solved through the finite-difference method. Subsequently, the forward modeling was then used to generate synthetic seismic data from a velocity model with four horizontal plane-parallel layers. Finally, by using the differential evolution algorithm, the inverted velocity model was generated and compared with the original one.

2.1 Theoretical and Practical Tools Used

2.1.1 Direct and Inverse Modeling

On one hand, geophysical forward modeling is the process of predicting the expected physical response of the method through the usage of Physics equations along with the seismic model, which is usually described in a mathematical equation. On the other hand, geophysical inverse modeling is about finding the mathematically described model which produces a certain measurable physical response. Figure 1 ilustrates forward and inversion modelings.

In the first part of this work, the construction of a seismic forward modeling operator (g) is made. In the second part, the inversion operator (g^{-1}) is produced. For further reading on inversion theory, it is suggested to see the first chapter of Menke (2012).

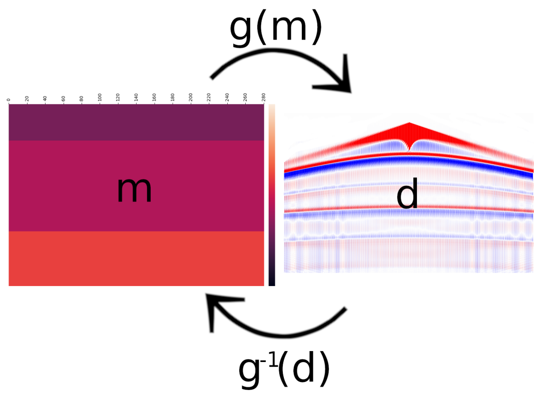

Figure 1: g is a function of the forward model, m, which returns a prediction of the measured data (a seismogram in this case), d. g^{-1} is its inverse function (a function that returns the velocity model m from the data d).

2.1.2 The Acoustic Wave Equation

Seismic data can be simulated by solving the wave equation, which is written as:

$$\frac{\partial^2 P}{\partial^2 t} = v^2 \nabla^2 P, \tag{1}$$

where P and v are the pressure and the media seismic velocity fields, respectively, and t, the time.

In this analytical form, the solutions that one could find to this wave equation, given the problem's initial and contour conditions, usually are very limited. However, given the complexity of both the subsurface media and the seismic pulse, the more general approach is to use numerical methods to solve the equation. Among the various methods to solve it, the finite-difference method was chosen.

2.1.3 The Finite-Difference Method

The finite-difference method (FDM) is a method to numerically solve partial differential equations for fixed variable intervals. The idea used is to approximate the limit that defines a derivative for a multi variable real function $f(x, y)$, as shown in the Eq. (2),

$$\frac{\partial f}{\partial x}(x, y) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x}, \tag{2}$$

an expression that is analytically equal to

$$\frac{\partial f}{\partial x}(x, y) = \lim_{\Delta x \to 0} \frac{f(x, y) - f(x - \Delta x, y)}{\Delta x}, \tag{3}$$

to

$$\frac{\partial f}{\partial x}(x, y) \approx \frac{f(x, y) - f(x - \Delta x, y)}{\Delta x}, \tag{4}$$

where Δx is, in Eq. (4), a very small fixed value. Similar approximations can be done with higher order derivatives — which will be the case bellow.

In the 2D plane xz (x denoting a horizontal direction and z the depth), Eq. (1) becomes:

$$\frac{\partial^2 P}{\partial t^2} = v^2 \left(\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial z^2} \right). \tag{5}$$

Making the approximations described in Eq. (4), the previous equation is translated to:

$$\begin{aligned} P(x, z, t) = 2P(x, z, t - \Delta t)- \\ P(x, z, t - 2\Delta t)+ \\ v^2 \Delta t^2 \nabla^2 P(x, z, t - \Delta t) \end{aligned} \tag{6}$$

where Δt is very small number, and $\nabla^2 P(x, z, t)$ is the laplacian of the P field at (x, z, t) or, in FDM terms:

$$\begin{aligned} \nabla^2 P(x, z, t) = \frac{1}{h^2}[&P(x - 1, z, t)+ \\ &P(x + 1, z, t)+ \\ &P(x, z - 1, t)+ \\ &P(x, z + 1, t)- \\ &4P(x, z, t)], \end{aligned} \tag{7}$$

h being a very small fixed real number.

The advantage of writing in this form is that the P-wave value at any point on the pressure field only depends on previous values (in time). Consequently, if the initial conditions of the problem are known, its future states can be calculated. Actually, as a 2nd degree differential equation solution, 2 values are needed for initial conditions. In the analytical form, these are the pressure value and the first derivative of the pressure in respect to time. In FDM terms, those two are translated to the initial pressure value and its values one Δt before — values that allow the calculation of the first derivative itself.

The next step is to create a mesh (be it a square one) over which the solution will be calculated by looping. If the solution wanted is the one after a given signal have disturbed the system, right before every loop, it is important to set the disturbance at the previous time frame calculated.

2.1.4 The Laplacian

In calculus, the laplacian of a field is a measurement of its "curvature" at a point. It is calculated over any orthogonal spacial directions (as x and z in last example). Looking at a 2D square mesh, there are 2 main ways to calculate the laplacian: using mesh concordant directions or the diagonal ones as illustrated in the Figure 2.

The points that were used to calculate the laplacian over the diagonal directions are $\sqrt{2}$ times more distant than the concordant ones. Hence, a more precise way to calculate the laplacian is to make a weighted average of both the diagonal and concordant ones, the weights being the inverse of the distance to the points used.

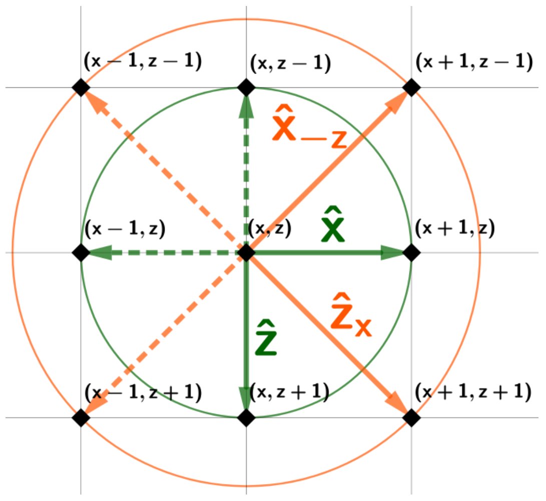

Figure 2: Concordant (green) and diagonal (orange) vectors to the mesh.

2.1.5 Unwanted Border Reflections

From the wave equation presented in Eq. (1) itself, not all the wave effects, like reflection and refraction, are trivially known. Nevertheless, many of then are intrinsic to the wave equation, as can be shown via FDM. This is usually much desirable; however, since the model is finite, therefore having borders, in order to avoid unwanted reflection on those (like the ones illustrated on Figure 3), a simulated borderless medium is needed. To do so, the pure FDM must be modified.

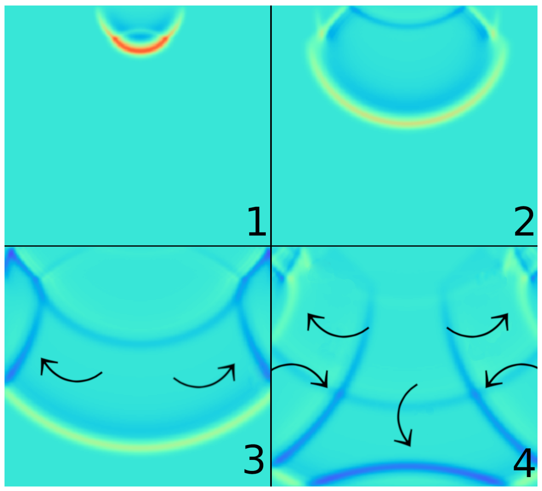

Figure 3: Seismic source shot and propagation through a model of two geological layers, purely calculated via FDM. Arrow point to unwanted border reflections. Picture plotted with Matplotlib.

A way to achieve this, is through dumping the wave propagation at the borders. It is made by multiplying a dumping factor to the $v^2 \Delta t^2 \nabla^2 P(x, z, t - \Delta t)$ term of Eq. (6) solely on the calculations of mesh cells in a layer near the border. This factor was calculated as in Santos (2002) rewritting of Cerjan et al. (1985): $w = e^{-\left(\text{fat } (n_{\text{amort}} - d)\right)^2}$ — where w is the dumping factor, fat is the dumping coefficient, n_{amort} is the dumping layer thickness and d is the point distance from the border.

The dumping diminishes the wave velocity each time more after it enters the layer, without sending sensible reflections back. Therefore, the wave is held near the border for some time and then it is unleashed. The best dumping coefficient depends on the length of the dumping layer, and the configuration used in this work was:

- dumping layer length: 100 mesh cells;
- dumping coeficient: 0.023.

The best configurations usually occurred when the product of dumping layer length and dumping coefficient was equal to 2.3.

The implemented method to remove border effects that seemed more appropriate was to not only use the wave propagation attenuation on the borders, but also to make the mesh bigger than the mesh that describes the model uses — in brief, all the attenuation was being done outside the model zone of the mesh.

2.2 Acquiring Synthetic Seismic Data

With the 2D P-wave function solved across time (t), and space (x and z), if one wants to get the response that a receptor in the surface would get, one could just choose z to be equal to the position of the receptor in the z axis, — the resulting solution is a seismogram, as shown in Figure 4 — and then set a horizontal position x in the mesh to receive the travel times of the arriving waves.

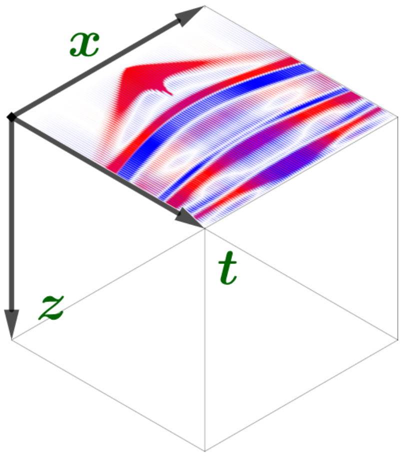

Figure 4: The seismogram is the xt face of P-wave solution prism.

2.3 Inversion with Differential Evolution (DE)

An evolutionary algorithm is a major set of algorithms whose structures are inspired by biological evolution Whitley (1994, p. 1), as making usage of key evolutionary processes such as random mutation and crossover. The differential evolution algorithm is part of this major set, being very useful optimization tools that can be used for searching in big continuous spaces. Their usual structure is:

1. generation of initial population of vectors covering the entire search space. They are called initial target vectors;
2. mutation of initial values of each vector by adding a weighted difference function of two other vectors from the initial population (this difference is a measure of the parameters mutation amplitude, which must converge with time);
3. crossover of the mutated vectors and target ones by randomly exchanging some parameters between both to generate trial vectors;
4. selection: each trial vector is evaluated by a fitness function and compared with the evaluation of its associated target vector, which will be substituted by the best suited between the two;
5. repeat steps 2 to 4 until evolution converges to a solution;

These steps can be expressed by Figure 5.

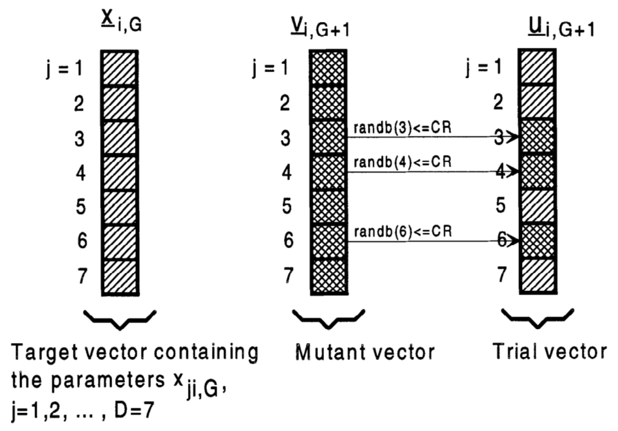

Figure 5: Scheme for a differential evolution loop (STORN; PRICE, 1997).

While an objective function is a function that is associated with a mathematical problem and has its critical values with the parameters that solve this problem, a cost function is an objective function that has a global minimum with its associated problem solution as parameters. Cost functions serve as ways to evaluate the opposite of the fitness of a parameter vector. Thus, selection would strive to get the parameter vectors that return the minimal cost function value (the maximum fitness).

This work's problem was finding four 3-layered parallel-plane horizontal seismic model's parameters based on their synthetic seismograms. In brief, the parameters were: 2 superior layer thicknesses, h_1 and h_2, and the P-wave propagation velocity of the 3 layers, V_1, V_2, V_3, as shown in Figure 6

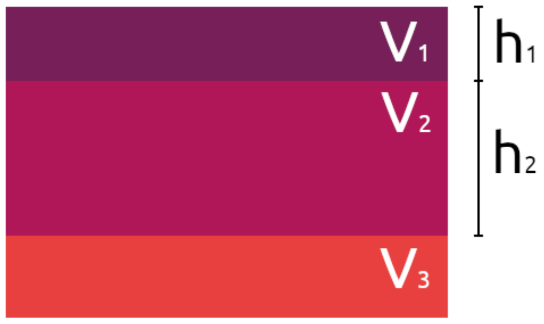

Figure 6: General 3-layered parallel-plane horizontal geologic model's seismic parameters scheme. Picture plotted with Matplotlib.

The cost function used was the Frobenius norm of the difference between the seismogram data (represented as a matrix inside the program) and the one generated from parameters of the imposed geologic model (another matrix). The Frobenius norm of a matrix (symbolized as $\|A\|_F$, A being the matrix) is calculated by:

$$\|A_{N\times N}\|_F = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}^2} \qquad (8)$$

and represents a distance from 0. Therefore, as the seismograms match, their difference must tend to 0, and when it happens, the parameters associated are the results of inversion.

Details on the entire differential evolution process can be found on Storn and Price (1997). The negative side of this method is that there is no guarantee that the most optimized solution will ever be found, since the evolution may get stuck in local minimums.

The maximum number of iterations used for the evolutionary algorithm was 50.

3 Results

3.1 Modeling

A wave propagation function was programmed in Python, using FDM. This function was used to generate seismograms. An example is plotted on Figure 8. For all generated seismograms, the spacing between geophones was set to 10 m.

The source signature used to be propagated was arbitrarily chosen to be $P_{\text{signal}} = t^{1/t}e^{-t^2}$, with $0 < t < 3$, and to last 300 ms in each shoot. The plot of this signal can be seen on Figure 7.

3.2 Inversion

There is an well implemented version of the differential evolution algorithm in Python, in scipy's optimization library. It is based on the work of Storn and Price (1997). Using the

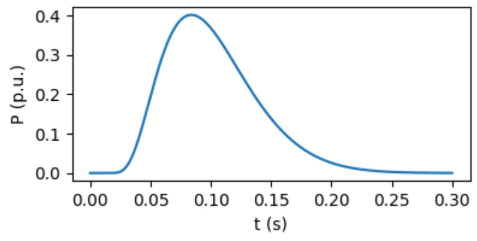

Figure 7: Source signature in pressure unit. Graph plotted with Matplotlib.

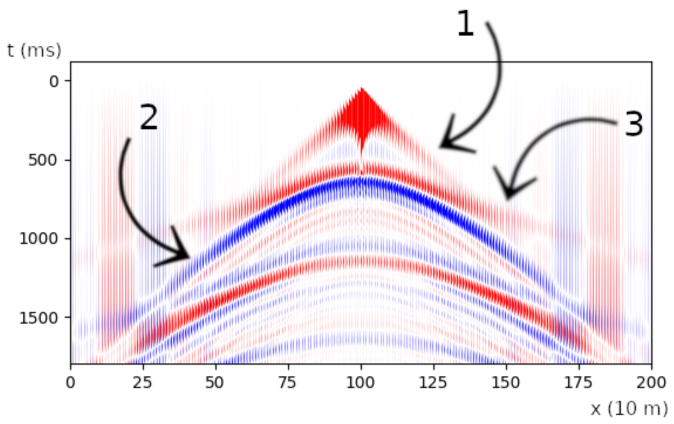

Figure 8: Example of seismogram generated via FDM. Numbers: three of the visible events — 1: direct wave; 2: reflected wave; 3: refracted wave. Picture plotted with Matplotlib.

algorithm to invert the seismic data led to Table 1. The used search interval bounds were 0 to 1000 m (counting in tens) for the thicknesses and 0 to 8000 m/s to the velocities.

Although inversions 2, 3 and 4 went well, getting very near the expected values, the first inversion process got stuck in a local cost function minimum: the first interface was ignored, and the second one was taken as first in its place. For this reason, the first inversion's h_1 parameter actually is related to the model's second interface depth (model's $h_1 + h_2$). Because of this, all the other parameters, but V_1 were mistaken.

The phenomenon occurred in first inversion is called premature convergence. In summary, the solution interval is sampled into a population of vector parameters in the differential evolution algorithm. None of the vectors were near to the global cost minimum as some were to the local one. In addition, the evolution kept converging to that leading solution until the iterations ended. Notice the very low contrast between V_2 and V_3: they were led to be computed to adapt to the multiple reflections residues, instead of the first real reflection, as seen in Figure 9. A way to diminish the probability to fall into a local minimum of this kind is to increase the vector initial population.

Although all the data above was calculated using the average laplacian, it is important to notice that there was a significant difference between seismograms generated with that operation and the ones generated with the traditional concordant laplacian. The Frobenius norm of the difference of two seismograms generated with the same parameters, but with different laplacian calculation methods, was evaluated to be about 0.36. This value, when compared with the cost function values of the inverted seismograms, might be significant.

M\P	h_1 (m)	h_2 (m)	V_1 (m/s)	V_2 (m/s)	V_3 (m/s)	C
M_1	500	500	2500	3000	5700	
$M_{1,\mathrm{i}}$	960	880	2630	5242	5551	0.015
M_2	400	600	4000	3000	7000	
$M_{2,\mathrm{i}}$	410	590	4095	2954	7000	0.006
M_3	300	900	5700	2500	3000	
$M_{3,\mathrm{i}}$	300	880	5692	2451	3091	0.001
M_4	450	850	2500	3000	5700	
$M_{4,\mathrm{i}}$	450	870	2503	3062	6044	0.004

Table 1: Seismogram paramaters and associated inversion parameters gotten. $M_j, j = 1,2,3,4$ stands for the j^{th} target model, while $M_{j,\mathrm{i}}, j = 1,2,3,4$ stands for the inverted model of M_j, based on its synthetic seismogram. In the table, C is the cost function of the inverted model.

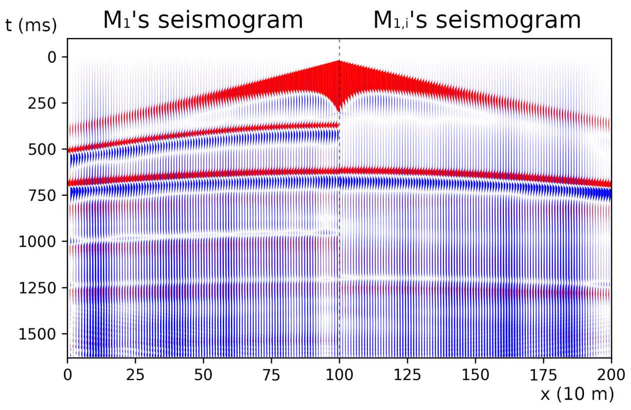

Figure 9: M_1 and $M_{1\mathrm{i}}$ associated seismograms. Picture made with Matplotlib.

Conclusions

In the present work, it was possible to do forward modeling through solving the 2D acoustic equation in order to obtain seismograms associated with different seismic models. It was made by using the finite-difference method. From this achievement, 4 plane-parallel horizontal 3-layered model seismograms were generated.

It was shown that differential evolution algorithms can indeed solve geophysical inversion problems on synthetic data. Nevertheless, it is important to know their risks: these methods are not guaranteed to get the most optimized model, as seen in the first inversion results. Increasing parameter vectors population can diminish this risk.

Two ways of calculating the laplacian have been utilized, in spite of both being versions of the finite-differences method of solving the wave equation. The first was to calculate on the concordant discrete laplacian (as traditionally calculated) and the second was the average discrete laplacian (a weighted average of the mesh concordant and diagonal laplacian values). By taking the Frobenius norm of the difference between seismograms generated with both discrete laplacian calculation methods returned a significant value difference, when compared to the cost function calculated at the end of the inversions that were made.

Further scientific research is now necessary to show the validity of evolutionary computing methods for inversion in other geophysical methods. In the near future, it may be also applied for real data problems.

Acknowledgements

Authors thank to DGf-IGEO-UFBA and CPGG-UFBA.

References

ANDRADE, Protásio Nery. **Imageamento Sísmico Através do Método de Expansão Rápida nos Domínios do Tempo e da Frequência**. 2002. p. 92. Tese de Doutorado – IGEO/UFBA.

CERJAN, Charles et al. A nonreflecting boundary condition for discrete acoustic and elastic wave equations. **Geophysics**, Society of Exploration Geophysicists, vol. 50, no. 4, pp. 705–708, 1985.

MENKE, William. **Geophysical data analysis: Discrete inverse theory**. 3. ed. [**sineloco**]: Academic press, 2012.

SANTOS, Jorge Luiz Rebelo. **Modelagem da equação da onda acústica aplicada ao imageamento de estruturas geológicas**. 2002. p. 92. Civil Engeneering Master Thesis – Universidade Federal do Rio de Janeiro/COPPE.

SOARES, Átila Saraiva Quintela. **Geração de modelo de meio poroso fractal para determinação de permeabilidade em rochas argilosas**. 2018. Undergraduate Thesis – IGEO/UFBA.

STORN, Rainer; PRICE, Kenneth. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. **Journal of global optimization**, Springer, vol. 11, no. 4, pp. 341–359, 1997.

WHITLEY, Darrell. A genetic algorithm tutorial. **Statistics and computing**, Springer, vol. 4, no. 2, pp. 65–85, 1994.