Modelling of Near field Properties of Seismic Waves in an Anisotropic Medium

Sergio Damas Arroyo

EPCC–SS94–02

Modelling of Near field Properties of Seismic Waves in an Anisotropic Medium

Sergio Damas Arroyo

EPCC–SS94–02

September 1994

Abstract

In the seismic monitoring of hydrocarbon reservoirs, the process of excitation, propagation and recording of seismic data is far from satisfactory. In a heterogeneous medium, many effects exist which cannot be directly assigned to either the source, medium or receiver. Further analysis is required to help make this distinction clearer, so that the target zone may be resolved with confidence. The simulation of the body wave radiation from the seismic source plays an important role in such work. In a homogeneous isotropic medium (2 elastic constants) this computation does not pose any special problems because the analytic form of the Green's function exists. However no near-field analytic solutions are as yet available for the case of a general anisotropic medium (21 elastic constants) which provides a way of simulating heterogeneities and fractures around the source region. Fortunately, computations for the anisotropic medium can be achieved numerically. A 2D finite difference method based upon the elastodynamic equation of motion has been developed to study this problem. The objective of the current work is to apply parallel computation to this method in order to optimize the results obtained.

1 Introduction

Since many important results remain discovered, the seismic exploration is one of the most studied areas in seismology. Many people would argue that this does not make exploration seismology important. Oil industry seismic research directs much of its effort towards using seismic imaging as a tool in both exploration and production phases of reservoir development. Each year more and more resolution is demanded of these techniques, and more emphasis is placed upon alternative acquisition techniques or improving data quality. One way in which the latter can be accomplished is by accurate modelling of the interaction of the source with the near-surface. However, it is not possible to obtain satisfactory analytic solutions for an arbitrary complexity, so that the most flexible approach is numerical modelling. Understanding the source helps eliminate or correct for it in the data.

This project forms part of an overall initiative to develop this modelling capability. This work differs from previous modelling studies in that we intend to specifically address the parallel optimisation of the code. In this way, our developments take advantage of faster code with less inherent inaccuracy.

The study of the near field properties in an homogenous anisotropic medium is one of the most unknown fields in seismology due to two factors: on the one hand researchers do not tipically consider the near field contribution to the wave behaviour. On the other hand, since no analytical solutions are as yet available for complete body waves in general anisotropic media or even for the free-surface of an isotropic medium.

This document is divided into four different parts:

- Background: Our problem and its motivations are described
- Description of approach taken: The way we applied parallelism, the problems encountered and results obtained are discussed.
- Work for the future
- Conclusions

2 Background

2.1 Description of our problem

2.1.1 Some useful concepts

Before starting to understand our problem it is important to define some useful concepts.

- Source

  A source is a point or place from which seismic waves originate. There are different kinds of sources and different ways to model the interaction between source and medium (for example: linear single force source —exploration source, linear dipole force source —an earthquake source only, symmetric source of compressional waves). Each of these forces is a point force; that is they act at a single location or particle of the medium. We may imagine a single node or grid point being acted upon by some sort of excitation. This has a knock-on effect on the other nodes. Here is a brief description of each of the sources mentioned above:

  Single force source The node is pushed in one specific direction (see figure 1).

  Dipole force source Two adjacent nodes are involved. One is pushed in one direction, the other in exactly the opposite direction (see figure 1).

  Symmetric source of compressional waves All directions are pushed at the same time - an outward waves of pressure.

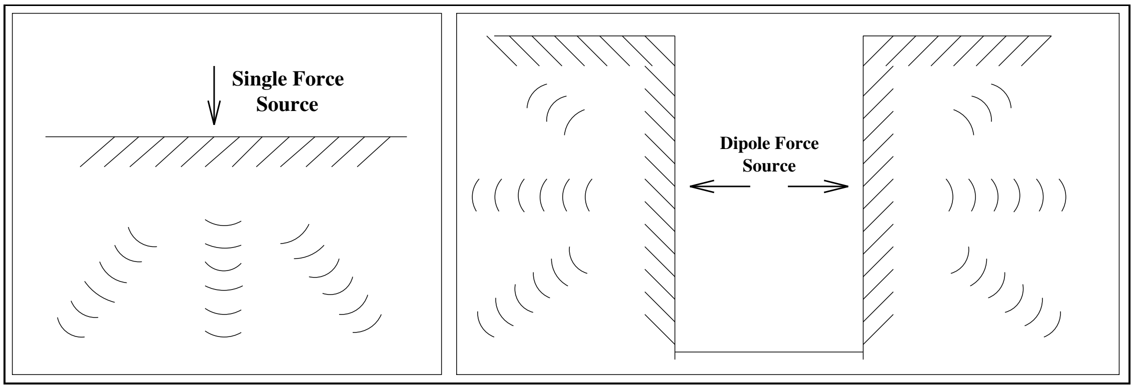


Figure 1: Different kind of sources

In our problem we shall concentrate on the single force source. Another type of source is the displacement source which is slightly more complicated, but means that instead of feeling a force via Newton's law:

$$Force/Spring = Constant \tag{1}$$

the effect is made up straightaway, so that a displacement source of two units gives two units when we feel it. A force of two units may give 2/K units, or some more complicated effect.

The idea is that we're not able to simulate in the same way an earthquake, an explosion or the stress applied to the surface of the Earth by a big truck, although all them are valid sources. In addition, we can have different kinds of earthquakes, of explosions or of trucks and we'll have different results as well.

- Receiver

  A receiver is just a detector is used to study the interaction between medium and source. Source behaviour is fixed, and receivers which record this behaviour.

  Physically, a typical receiver is a magnet attached to the earth, and a coil attached to an amplifier. As the ground moves an electromotive force (emf) is induced due to Lenz's law, and this is amplified and recorded. The emf is directly proportional to velocity of ground motion, so we are recording the ground velocity, which we can convert to displacement by integration.

- Near Field Terms

  The near field is a very short distance from the source, such that the distance/wavelength is less than two or three. At such distances the various arrivals from the source are all quite strong and many have not decayed, so that the behaviour is very complicated. As we move further away some terms decay rapidly, leaving a much simpler wavefield - actually only a few arrivals - P, and two S waves.

- Isotropic vs. Anisotropic Medium

  Finally, we are looking at the difference between an isotropic and an anisotropic medium. The most obvious difference is the way waves are propagated in the medium: in an isotropic medium a wave is propagated with the same velocity in every direction. But, this is not true in the case of an anisotropic medium. As we can see in figure 2, the velocity of the wave in each direction is able to cause different deformations or strains in the medium.

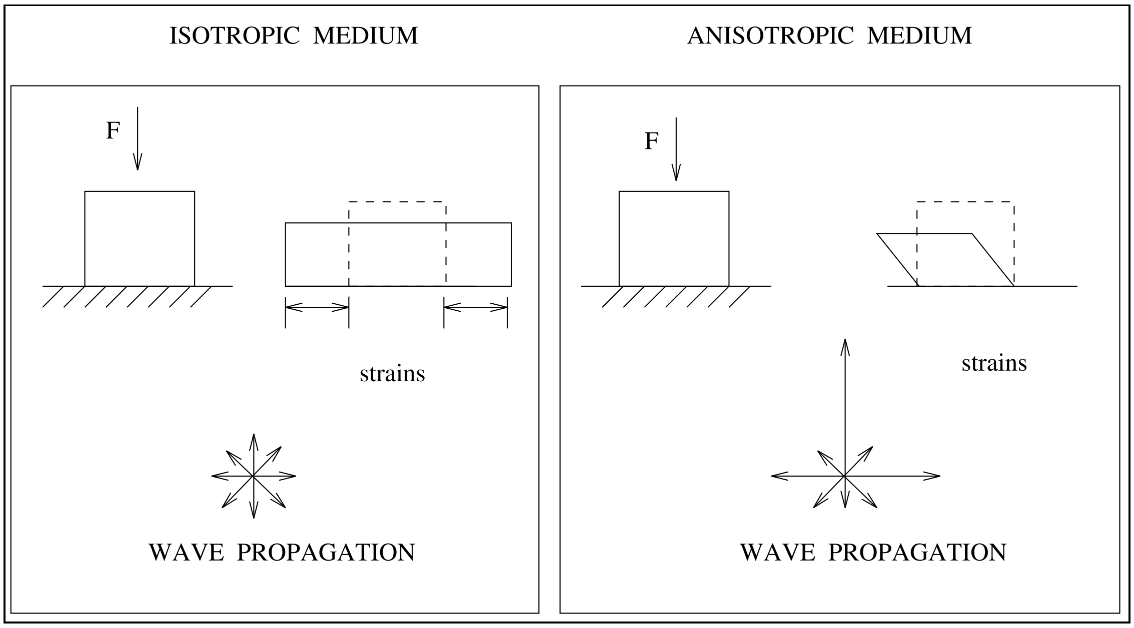


Figure 2: Wave propagation: isotropic vs. anisotropic medium

2.1.2 Finite-difference method

Finite-difference equation for a homogeneous anisotropic medium As mentioned above a Finite-Difference method is needed if we want to study a homogeneous anisotropic medium. In the following we will confine ourselves to the solution of the 2-D problem, i.e. we will simplify the problem so that the resulting displacement field $u_i(x_j, t)$ depends only on two space coordinates and time:

$$u_i = u_i(x, y, t), i = 1, 2, 3 \tag{2}$$

In converting the derivatives in the elastodynamic equation [Aki and G., 1980a] into finite differences, we will apply the standard centred-difference formulae of second-order accuracy [Aki and G., 1980b] and we will use the so-called explicit scheme. This is important, as it means that we can use the PUL-RD utility library (see section 3). The implicit scheme uses all nodal contributions to make a decision and isn't suitable - but in many cases is more accurate.

Thus the system of finite-difference equations reads:

$$u_i(t + \Delta t) = \Delta t^2/\rho[f_i + c_{ijkl}\Delta^2 u_k/\Delta x_j \Delta x_l] - 2u_i(t) + u_i(t - \Delta t), i = 1, 2, 3 \tag{3}$$

Holding for

$$j = l = 1 : \Delta^2 u_k/\Delta x_j \Delta x_i = \Delta^2 u_k/\Delta x^2 = [u_k(x + \Delta x, y) - 2u_k(x, y) + u_k(x - \Delta x, y)]/\Delta x^2, \tag{4}$$

$$j = l = 2 : \Delta^2 u_k/\Delta x_j \Delta x_i = \Delta^2 u_k/\Delta y^2 = [u_k(x, y + \Delta y) - 2u_k(x, y) + u_k(x, y - \Delta y)]/\Delta y^2, \tag{5}$$

$$\begin{aligned} & j = 1, l = 2 \ or \ j = 2, l = 1 : \\ \Delta^2 u_k/\Delta x_j \Delta x_i = \Delta^2 u_k/\Delta x \Delta y = \quad & [u_k(x + \Delta x, y + \Delta y) - u_k(x + \Delta x, y - \Delta y) - \\ & u_k(x - \Delta x, y + \Delta y) + u_k(x - \Delta x, y - \Delta y)]/4\Delta x \Delta y, \end{aligned}$$

$$j = 3 \; or \; l = 3 : \Delta^2 u_k / \Delta x_j \Delta x_i = 0 \tag{6}$$

where Δt is the time step;Δx, Δy is the space step; c_{ijkl} are the components of Hooke's tensor of elastic coefficients; $u_k = u_k(x, y, t)$ are the components of the displacement vector;ρ is the density of the medium; and $f_i = f_i(x, y, t)$ are the components of the acting body force.

Stability conditions of the finite-difference scheme A physically meaningful numerical computation presumes the stability of the finite-difference algorithm [Mitchell, 1969]. This condition is satisfied if the inequality:

$$\Delta t \leq \Delta x / (\nu_p^2 + \nu_s^2)^{1/2} \tag{7}$$

[Alterman and Loewenthal, 1970] where ν_p, ν_s are velocities of P- and S-waves, is valid. In the case of an anisotropic medium ν_p, ν_s denote the maximum values of P- and S-wave group velocities.

A condition connected with grid dispersion is added to formula 7:

$$\lambda_{min} \geq 10 \Delta x \tag{8}$$

Satisfaction of both inequalities 7 and 8 cannot be taken strictly as a sufficient condition for stability, but rather to the effect that the better they are satisfied, the more exact a solution we will obtain.

Boundary and initial conditions As mentioned above, we will study the polarization properties of waves propagating through an unbounded and bounded medium. We will practically satisfy this condition by dealing only with the direct wave. The receiver position relative to the boundary of the finite-difference model will be chosen in such a manner that the waves reflected from this boundary will arrive at times sufficient for the separation of the direct wave.

In order to define the boundary conditions, we will specify a rigid boundary, i.e. for every time level we will put all the three displacement components equal to zero at the boundary points. Physically this means that the boundary is subject to no deformations.

Zero displacement for times $t \leq 0$ and all the grid points will be chosen as the initial conditions.

2.2 Motivations of this project

In seismology, each year more and more resolution is demanded of seismic imaging, and more emphasis is placed upon alternative acquisition techniques or improving data quality.

This project tries to make a first step in exploring near field properties in an anisotropic medium, as a way of satisfying these demands: making data much simpler to understand in terms of subsurface structure. By doing so, not only will we have a better resolution of reservoir properties but we'll also be able to characterise the near-surface layers for environmental geophysics surveys. In addition, we may find a way of making better sources based upon our experiences. Moreover, this project may also help to produce better and more reliable predictions of earthquakes through more accurate resolution of the source mechanism.

Modelling heterogeneities of the Earth is important as this is what it is in reality. This requires flexible and accurate computation of seismic wave propagation. So, the application of parallel computation is likely to be one of the most important advances in seismology because of the importance of the increased resolution and faster results it facilitates.

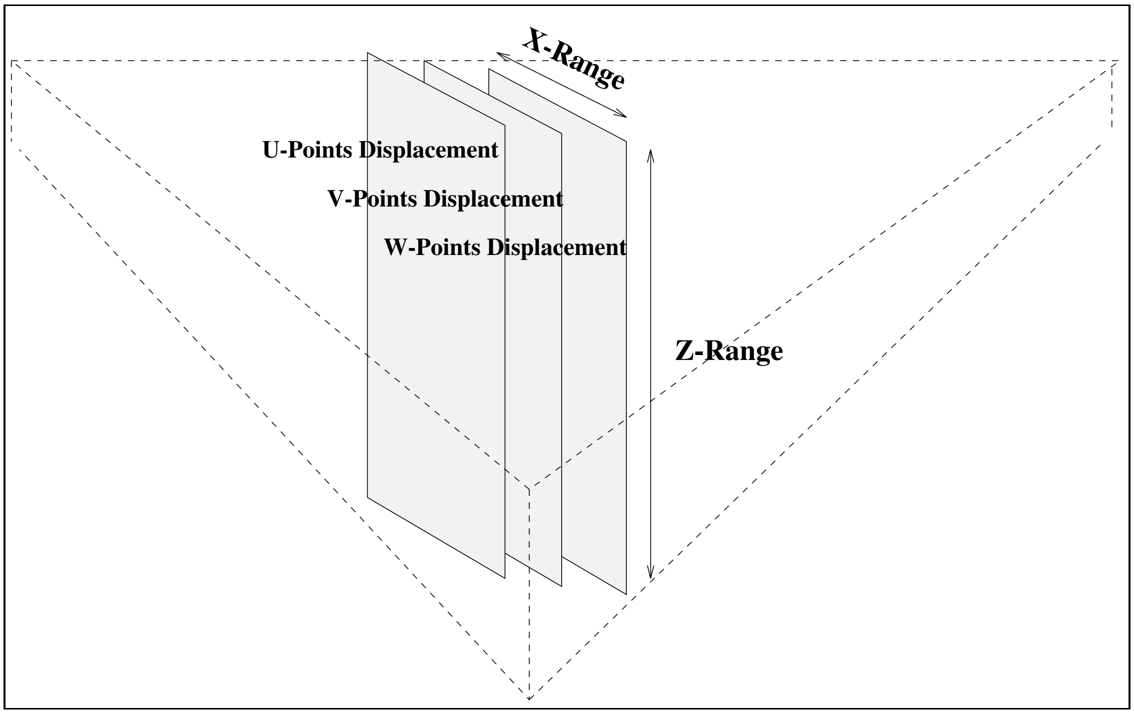


Figure 3: Basic data structures used

3 Description of the approach taken

3.1 Data structures used

In order to understand the approach taken during this project is very important to describe some important data structures used to design our system.

As we could see in section (2.1.2) our work will be based on the computation of the displacement in each direction for each of the points of our model. To do so will work on three different arrays u, v, and w corresponding to X, Y, and Z space directions (see figure 3) where we store the corresponding displacement of the points. These arrays are the set of points we are discretising in our simulation and some of them will be our receiver points.

Notice that each of this points is considered as a two-dimensional one, with only two different coordinates (x-axis and z-axis), therefore we are working on a surface of points to compute the displacements considered.

This computation is based on the equation development done in section 2.1.2 and as we could see, to compute the future value of each point we'll need to know:

- The current value of the point.
- The current values of the points which surround the one we are studying.
- The previous value of the point we are studying.

In other words: the idea is that if we want to update the value of the point we are considering we have to look at its actual and previous local configuration and then decide whatthe behaviour of this point at the next time level will be (in the future). See figure 4. This means that we'll need three additional structures where to store the previous displacements.

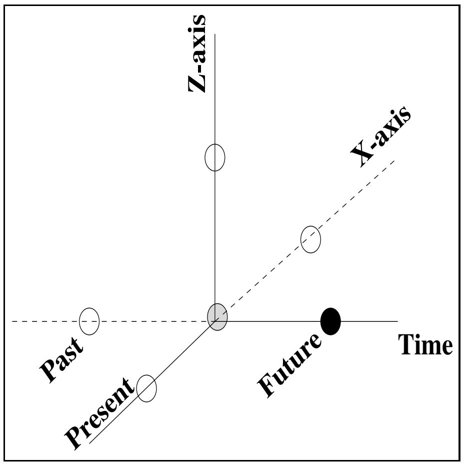


Figure 4: Updating one point

3.2 Algorithm applied

The objective of this section is not to give a detailed explanation of each line of the project implementation, but after reading this section the reader should have an idea of the different steps followed to design our code.

First, we fix the source behaviour through time. As commented in section 2.1.1 the source we'll consider in our project will be a single force. Then we are able to run our simulation: in each time level we'll have to:

- Update the values of the points which surround the source.
- Update the values of the rest of the points we consider in our model.
- Store the values at the receivers in a special data structure (seismograms).

After doing this simulation, after our simulation we'll have stored the displacement of the receivers at each time level in a special structure and we'll be able to work on the computation of these values. So, in addition to the data structures in which we compute the displacement values of each point, we'll have other structures (seismographes) where we store the displacements of each receiver.

3.3 Most important characteristics of a PUL-Regular Domain Decomposition (PUL-RD) problem

In order to understand the way parallelism has been applied it is very important to understand the basic characteristics of a problem domain to which PUL-Regular Domain Decomposition (PUL-RD) may be applied. The following list describes these characteristics (see PUL-RD user guide):

- A global n-dimensional data set that is completely populated and has rectilinear shape. The position of each data element within a global data set therefore maps directly to the position of the represented data-point in space. RD also contains facilities to simultaneously handle several such data sets, of the same dimensionality and logical size.
- A single operation that is to be applied to all data elements. The calculation to update an individual data element may depend on the values of some set of neighbouring data elements. However, the set of neighbouring elements must be expressible abstractly as a stencil, i.e. a mask which represents the pattern of access into the global data set relative to the position of the data element to be updated. It is also a requirement that the area of data access represented by the stencil must

specify the furthest extent of data access required for updating any of the data elements in the global data set. The stencil must remain constant across space but may vary over the time.

3.4 Way of applying parallelism

3.4.1 What to parallelize

Once we have understood the important background about our problem is and the most important characteristics of a PUL-RD problem, we can assert that it seems these characteristics are very close to the description of our system. So, the most important part of the parallelization, will be done using this utility library.

First of all, we've got to think which operations are important to optimise. As we've seen in section 3.2 basically we have to do four different operations. Let's analyse them:

- Fixing the source behaviour: since this operation will be done just once during the whole simulation, it doesn't seem to be the most important one. It's obvious the operation we are searching for must be done at least once in each time level. So it could be any of the three which follow:
- Updating the points which surround the source: this operation will involve only a very small set of elements which have locations which are close to the source.
- Update the rest of the points: we'll have to update each of the points of each of the two-dimensional arrays in each time level. It seems we'll have to do a large computation to get these results.
- Save the displacements of the receiver points: since we only have to store the values we have computed previously and the number of receiver will always be much less than the total number of points we are considering, this operation doesn't seem to be so difficult to compute.

3.4.2 How to parallelize

The idea will be, since in each time level we are doing a local updating of each of the three arrays (corresponding to each of the three X,Y and Z directions of the space) we can parallelise this operation and work with smaller arrays on different proccesors. This will be the way of speeding up our computations.

The point is: on the one hand we need to update the arrays taking into account the values of the neighbouring points, on the other hand we are doing a portion of each of the arrays we are working on, so we wonder what to do with the points on the boundaries of our data set; how to update them. The answer is given by PUL-RD: each of the processes in the grid will exchange the boundaries of the data which they hold with their neighbours, in order to compute the values of the data elements at the edges of their data block. It is therefore necessary to perform a boundary swap for each complete computation step. PUL-RD provides functions to perform both blocking and non-blocking boundary swaps, in our case the non-blocking communication interface will be useful since it allows us to update the internal region of the block and once the swapping is done, we can update the boundaries.

3.4.3 Boundary conditions

Conditions for the source There is one specific requirement in our system, if we want to use parallel computing and partition our grids in different blocks to compute on different processors, we have to take into account it's not possible for our source to be near a boundary. The reason is because if we did so, we would have to update the boundaries immediatly (as mentioned in section 3.2 points which surround the source will be updated in a special manner) and this won't happen in our simulation until the arrays get swapped.

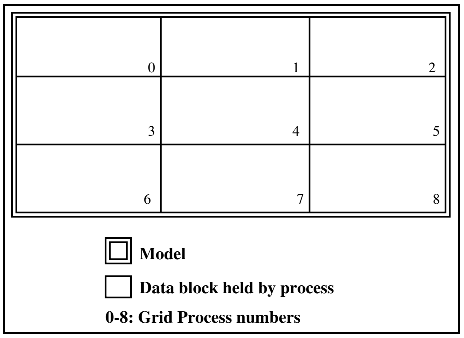


Figure 5: Model edges update

Model edges update As mentioned in section 2.1.2 the receiver position relative to the boundary of the finite-difference model will be chosen in such a manner that the waves reflected from this boundary will arrive at times sufficient for the separation of the direct wave. This means we'll have to care about these edges in our model. Therefore we'll have to study whether part of the real boundary of the model is inside the block of the regular decomposition we are working on. For instance: if we have such a decomposition as shown in figure 5. In order to update the nine blocks we are working on we'll have to follow different strategies: we'll have to take into account that the boundaries of our model are distributed on block numbers 0, 1, 2, 3, 5, 6, 7, 8. On the other hand, we'll be able to update all the points in block number 4 in the same way.

We have also mentioned that we will specify a rigid boundary, i.e. for every time level we will put all the three displacement components equal to zero at the boundary points. This rigid boundary update will be done at the end of all the rest of updating computations in order to avoid wrong interactions.

3.5 Problems encountered

3.5.1 Understand the existing code

This project is based on a seismic modelling code given to us by the British Geological Survey (B.G.S.). This code simulates seismic waves propagating through the Earth. But we have been only able to take advantage of some functions and data structures of this code because of:

- It was designed to solve a different problem: basically an Earthquake
- It was designed to run on a PC

I'd like to mention I have encountered different problems to understand this existing code, some of them because of the geophysics concepts and computations which are involved, but also because of it's poorly written, designed (use of unnecessary global variables, not using abstract data types...), etc. But in any case, since one of the goals of this project was to parallelize this code I had to adapt it. I think the final result would have been better if I had another basis to work from.

3.5.2 PUL-RD bug

It should be mentioned, I found a bug in the PUL-RD utility library which delayed my work for at least ten days. This is probably the most important reason for delaying some of the planned goals at the beginning of this work.

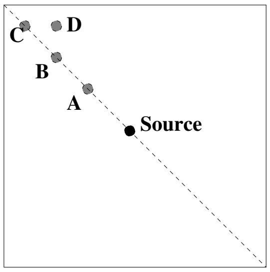


Figure 6: Distribution of source and receivers in the example

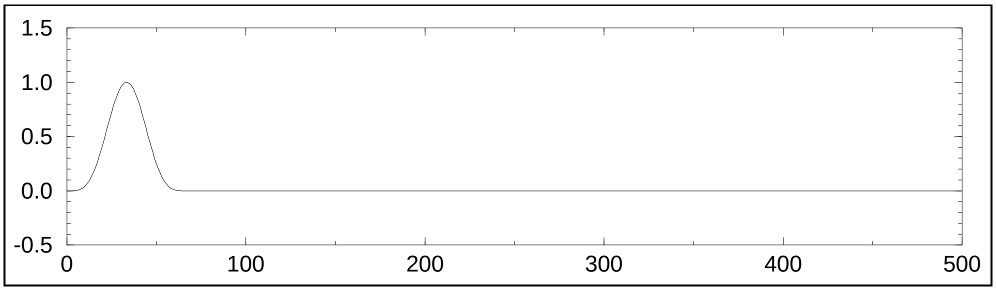


Figure 7: Source behaviour

3.6 Results obtained

3.6.1 Testing from extensions

The first and most important goal was to parallelize the existing code given to us by B.G.S. and check that the results obtained in both parallel and sequential versions were the same. This goal has been achieved. In order to get the same results we've had to use the same data types in both sequential and parallel versions. This means we are not getting more accurate results in the parallel application, but we could easily take advantage of more accurate computations on parallel architectures.

3.7 Running in other architectures

The project has been run on a Sun cluster platform. In any case, it could be easily moved to faster architectures like Transputers, the CS-2, or even the Cray T3D, because the code is completely portable.

3.8 Running a simulation

Let's analyse the results obtained in a simulation. In this case figure 6 shows the location of the source and the four receivers considered. The source will be a single dipole one and the seismograph of the Y-component of the source is shown in figure 7. As we can see the source has shaken one pulse and the stays quiet.

We are able to see the results obtained during the simulation in figure 3.8. The first, second and third rows of the figure show the displacements of each point of our model in the X, Y and Z directions respectively. Each picture in each row shows the configuration after 0, one hundred, two hundred, three hundred... iterations. Finally for five hundred iterations we can see reflecting effects in both the X and Z directions.

Seismographes for the four receivers we are working with, are shown in figure 9. We can see one of the lines (the thicker) corresponding to the D receiver point, has behaviour which is not exactly the same as the rest of the receiver points. This is because the location of D is not right on the diagonal of the model, but is quite close.

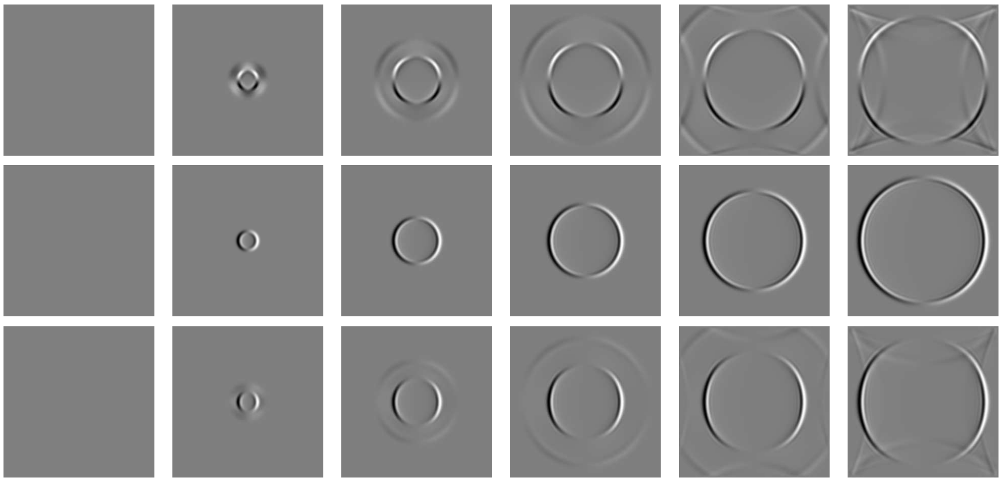


Figure 8: Simulation

4 Work for the future

- Change the characteristics of the source

  One possible extension could be to change the location and other physical characteristics of the source in order to interact with the medium in different ways. The characteristics of sources we could work with are as varied as those discussed in section 2.1.1. Once the mathematician model for the source is built, this adaptation should not be too difficult to achieve in our system.

- Optimize data structures

  There are many things which could be done in this area. We could think about the application of the object oriented programming paradigm, which would give a better abstract way of working than of the current one. For instance, we could think about the source as one object which has several characteristics (location in the space, stresses it applies, stress directions...).

  Another idea is to work to get more flexible data structures, with dynamic allocation which would let us work with arbitrarily large models.

- Thinking about 3-D

  As mentioned in section 3.1 we are actually working on two-dimensional arrays, but it could be very interesting to work in 3-D with the source and the different receiver points.

- Improve X-interface and visualization

  It would be friendlier for the user to have an X-window interface and a visualization of the results. This is one of the areas I wasn't able to attack properly during my project because of the PUL-RD bug mentioned before. In any case, my intention was to have an animation of the whole set of receiver points through time which would give the user a real impression of what is going on during each time step in our simulation. Once we have done all the computation for the receivers, this animation wouldn't be difficult to realize with the appropriated software.

5 Conclusions

This project is an attempt to paralllelise a finite-difference aproximation to a not well known area in seismology: the near field properties of seismic waves in an homogenous anisotropic medium. Although more results can be obtained, we hope this project will be an useful tool for geophysical computations.

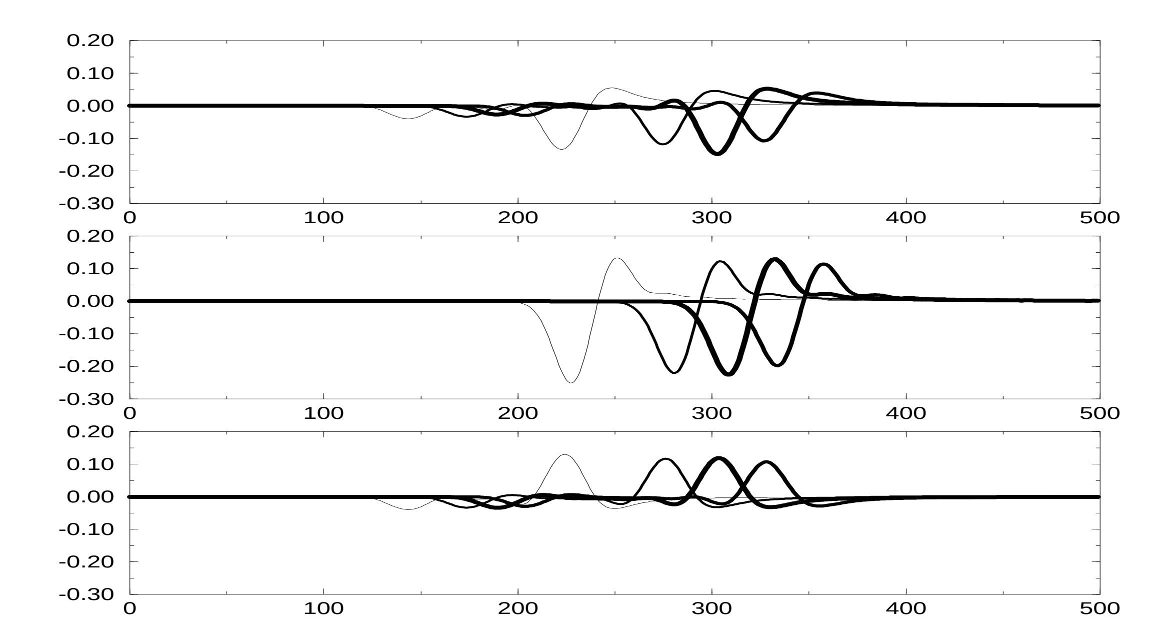


Figure 9: Seismographes for the receiver points

As mentioned above there are many things we could envisage as possible extensions and we hope the results of this project will provide a useful starting point for further developments.

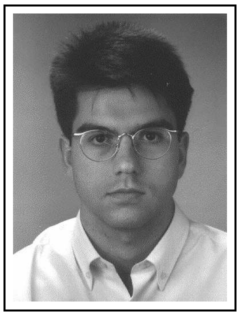
References

[Aki and G., 1980a] K. Aki and Richaards. P. G. *Quantitative Seismology, Theory and Methods*, volume I. W.H. Freeman, 1980.

[Aki and G., 1980b] K. Aki and Richaards. P. G. *Quantitative Seismology, Theory and Methods*, volume II. W.H. Freeman, 1980.

[Alterman and Loewenthal, 1970] Z. Alterman and D. Loewenthal. *Computer generated seismograms, in Methods in Computational Physics*, volume 12. Academic Press, New York, 1970.

[Mitchell, 1969] A. R. Mitchell. *Computational Methods in Partial Differential Equations*. John Wiley and Sons, New York, 1969.



Sergio Damas Arroyo is a 4th year student of the 5-year Computer Science programme degree at the University of Granada (Spain). After graduation in 1995 he hopes to pursue his doctoral studies in Computer Science.

This project was supervised by Patrick Surry (EPCC) and Colin MacBeth (British Geological Survey)