Using Expert Knowledge in Solving the Seismic Inverse Problem [⋆]

Matthew G. Averill, Kate C. Miller, G. Randy Keller, Vladik Kreinovich, Roberto Araiza, and Scott A. Starks

Pan-American Center for Earth and Environmental Studies
University of Texas at El Paso, El Paso, TX 79968, USA

Abstract

For many practical applications, it it important to solve the seismic inverse problem, i.e., to measure seismic travel times and reconstruct velocities at different depths from this data. The existing algorithms for solving the seismic inverse problem often take too long and/or produce un-physical results – because they do not take into account the knowledge of geophysicist experts. In this paper, we analyze how expert knowledge can be used in solving the seismic inverse problem.

Key words: seismic inverse problem, expert knowledge

1 Seismic Inverse Problem: Brief Introduction

In evaluations of natural resources and in the search for natural resources, it is very important to determine Earth structure. Our civilization greatly depends on the things we extract from the Earth, such as fossil fuels (oil, coal, natural gas), minerals, and water. Our need for these commodities is constantly growing, and because of this growth, they are being exhausted. Even under the best conservation policies, there is (and there will be) a constant need to find new sources of minerals, fuels, and water.

[⋆] This work was supported in part by NASA under cooperative agreement NCC5-209 and by NSF grants EAR-0112968, EAR-0225670, and EIA-0321328. The authors are very thankful to Dimitar Filev, Guest Editor of this issue, for his encouragement.

Email addresses: `averill@geo.utep.edu` (Matthew G. Averill), `miller@geo.utep.edu` (Kate C. Miller), `keller@utep.edu` (G. Randy Keller), `vladik@utep.edu` (Vladik Kreinovich), `raraiza@utep.edu` (Roberto Araiza), `sstarks@utep.edu` (Scott A. Starks).

The only sure-proof way to guarantee that there are resources such as minerals at a certain location is to actually drill a borehole and analyze the materials extracted. However, exploration for natural resources using indirect means began in earnest during the first half of the 20th century. The result was the discovery of many large relatively easy to locate resources such as the oil in the Middle East.

However, nowadays, most easy-to-access mineral resources have already been discovered. For example, new oil fields are mainly discovered either at large depths, or under water, or in very remote areas – in short, in the areas where drilling is very expensive. It is therefore desirable to predict the presence of resources as accurately as possible before we invest in drilling.

From previous exploration experiences, we usually have a good idea of what type of structures are symptomatic for a particular region. For example, oil and gas tend to concentrate near the top of natural underground domal structures. So, to be able to distinguish between more promising and less promising locations, it is desirable to determine the structure of the Earth at these locations. To be more precise, we want to know the structure at different depths z at different locations (x, y).

Determination of Earth structure is also very important for assessing earthquake risk. Another vitally important application where the knowledge of the Earth structure is crucial is the assessment of earth hazards. Earthquakes can be very destructive, so it is important to be able to estimate the probability of an earthquake, where one is most likely to occur, and what will be the magnitude of the expected earthquake. Geophysicists have shown that earthquakes result from accumulation of mechanical stress; so if we know the detailed structure of the corresponding Earth locations, we can get a good idea of the corresponding stresses and faults present and the potential for occurrence of an earthquake. From this viewpoint, it is also very important to determine the structure of the Earth.

Data that we can use to determine the Earth structure. In general, to determine the Earth structure, we can use different measurement results that can be obtained without actually drilling the boreholes: e.g., gravity and magnetic measurements, analyzing the travel-times and paths of seismic ways as they propagate through the earth, etc.

Forward problem. The relation between the Earth structure and the related measurable quantities is usually known. So, when we know the exact structure at a given Earth location, we can predict, with reasonable accuracy, the corresponding values of the measured quantities – we can predict the local value of the gravity field, the time that a seismic signal needs to travel from its origin to the sensor, etc.

For example, once we know the density $\rho(\vec{x}) = \rho(x, y, z)$ at different 3-D points (x, y, z), we can determine the resulting ground level ($z = 0$) gravity value $g(\vec{X}) = g(X, Y, 0)$ at a location (X, Y), by using the formula dating back to the original works of Newton:

$$g(\vec{X}) = G \cdot \int \frac{\rho(\vec{x}) \cdot (\vec{x} - \vec{X})}{|\vec{x} - \vec{X}|^3} \, d\vec{x},$$

where G is the universal gravity constant.

In general, corresponding formulas come from problems which geophysicists usually solve: what is the gravity field generated by given mass distribution, what is the magnetic field generated by a given distribution of magnetic materials, what is the travel-time of a seismic signal between two given locations in the given medium, etc. For example, when we want to know how the signal propagates, we can start at the source of the signal and go forward step-by-step simulating the way the signal actually travels. Such problems are therefore usually called *forward* problems.

Inverse problems. Forward problems enable us, given a model of the Earth, to predict the values of different signals. What we need in the above geophysical applications is the opposite: given the measured values of different signals, we need to reconstruct the structure of the Earth at the location where the measurements have been made. Such problems are therefore called *inverse problems.*

Seismic measurements are usually the most informative. Because of the importance and difficulty of the inverse problem, geophysicists would like to use all possible measurement results: gravity, magnetic, seismic data, etc. In this paper, we will concentrate on the measurements which carry the largest amount of information about the Earth structure and are, therefore, most important for solving inverse problems.

Some measurements – like gravity and magnetic measurements – describe the overall effect of a large area. These measurements can help us determine the average mass density in the area, or the average concentration of magnetic materials in the area, but they often do not determine the detailed structure of this area. This detailed structure can be determined only from measurements which are narrowly focused on small sub-areas of interest.

The most important of these measurements are usually *seismic measurements*. Seismic measurements involve the recording of vibrations caused by distant earthquakes, explosions, or mechanical devices. For example, these records are what seismographic stations all over the world still use to detect earthquakes. However, the signal coming from an earthquake carries not only information about the earthquake itself, it also carries the information about the materials along the path from an earthquake to the station: e.g., by measuring the travel-time of a seismic wave, checking how fast the signal came, we can determine the velocity of sound v in these materials. Usually, the velocity of sound increases with increasing density, so, by knowing the velocity of sound at different 3-D points, we will be able to determine the density of materials at different locations and different depths.

The main problem with the analysis of earthquake data (i.e., *passive* seismic data) is that earthquakes are rare events, and they mainly occur in a few seismically active belts. Thus, we have a very uneven distribution of sources and receivers that results in a "fuzzy" image of earth structure in many areas.

To get a better understanding of the Earth structure, we must therefore rely on *active* seismic data – in other words, we must make artificial explosions, place sensors around them, and measure how the resulting seismic waves propagate. The most important information about the seismic wave is the *travel-time* t_i, i.e., the time that it takes for the wave to travel from its source to the sensor. to determine the geophysical structure of a region, we measure seismic travel times and reconstruct velocities at different depths from these data. The problem of reconstructing this structure is called the *seismic inverse problem.*

2 Known Algorithms for Solving the Seismic Inverse Problem: Description, Successes, Limitations

Towards a precise mathematical formulation of the seismic inverse problem. Let us first describe what we want to determine. Our objective is to find the values of the velocity $v(\vec{x})$ at different 3-D points $\vec{x}$. To exactly describe a generic function, we need an infinite number of parameters: e.g., values $v(\vec{x})$ at all possible 3-D points $\vec{x}$.

In reality, based on the finite number of measurements, we can only reconstruct a finite number of parameters. So, at best, we can only reconstruct the values of velocity at a finite number of points $\vec{x}$. A natural way is to take a rectangular grid and to reconstruct the velocities v_j at different grid points.

Once we have found these values, we need to be able to reconstruct the values of velocity at other points, i.e., *interpolate* the dependence $v(\vec{x})$ from the grid points to the entire zone. In many practical situations, we know that the dependence is smooth; in such situations, it is reasonable to use a smoothing interpolation: e.g., piece-wise linear, piece-wise quadratic, etc. However, inside the Earth, there are often abrupt boundaries between different layers, so the dependence is often not smooth. As a result, in geophysics, a more reasonable interpolation is to take, as $v(\vec{x})$, the velocity at the closest grid point (i.e., use piece-wise constant interpolation).

Points which are closest to one point on a rectangular grid form a rectangular cell around this point. Thus, what we are doing is, in effect, dividing the 3-D zone of interest into a grid of rectangular cells, and, for simplicity, assuming that the velocity is constant throughout each cell. In the following text, we will denote the velocity in j-th cell by v_j.

Algorithm for the forward problem: brief description. Once we know the velocities v_j in each cell j, we can then determine the paths which seismic waves take. Seismic waves travel along the shortest path – shortest in terms of time. It can be easily determined that for such paths, within each cell, the path is a straight line, and on the border between the two cells with velocities v and v', the direction of the path changes in accordance with Snell's law

$$\frac{\sin(\varphi)}{v} = \frac{\sin(\varphi')}{v'},$$

where φ and φ' are the angles between the paths and the line orthogonal to the border between the cells. (If this formula requires $\sin(\varphi') > 1$, this means that this wave cannot penetrate into the neighboring cell at all; instead, it bounces back into the original cell with the same angle φ.)

In particular, we can thus determine the paths from the source to each sensor. The travel-time t_i along i-th path can then be determined as the sum of travel-times in different cells j through which this path passes: $t_i = \sum\limits_j \ell_{ij} v_j$, where ℓ_{ij} denotes the length of the part of i-th path within cell j.

This formula becomes linear if we replace the original unknowns – velocities v_j – by their inverses $s_j \stackrel{\text{def}}{=} \dfrac{1}{v_j}$, called *slownesses*. In terms of slownesses, the formula for the travel-time takes the simpler form $t_i = \sum\limits_j \ell_{ij} \cdot s_j$.

Algorithm for the inverse problem: general description. There are several algorithms for solving this inverse problem; see, e.g., [7,11,16]. The most widely used is the following iterative algorithm proposed by John Hole [7].

At each stage of this algorithm, we have some approximation to the desired slownesses. We start with some reasonable initial slownesses, and we hope that after several iterations, we will be able to get slownesses which are much closer to the actual values.

At each iteration, we first use the currently known slownesses s_j to find the corresponding paths from the source to each sensor. Based on these paths, we compute the predicted values $t_i = \sum\limits_j \ell_{ij} \cdot s_j$ of travel-times.

Since the currently known slownesses s_j are only approximately correct, the travel-times t_i (which are predicted based on these slownesses) are approximately equal to the measured travel-times $\widetilde{t}_i$; there is, in general, a discrepancy $\Delta t_i \stackrel{\text{def}}{=} \widetilde{t}_i - t_i \neq 0$. It is therefore necessary to use these discrepancies to update the current values of slownesses, i.e., replace the current values s_j with corrected values $s_j + \Delta s_j$. The objective of this correction is eliminate (or at least decrease) the discrepancies $\Delta t_i \neq 0$. In other words, the objective is to make sure that for the corrected values of the slowness, the predicted travel-times are closer to $\widetilde{t}_i$.

Of course, once we have changed the slownesses, the shortest paths will also change; however, if the current values of slownesses are reasonable, the differences in slowness are not large, and thus, paths will not change much. Thus, in the first approximation, we can assume that the paths are the same, i.e., that for each i and j, the length ℓ_{ij} remains the same. In this approximation, the new travel-times are equal to $\sum \ell_{ij} \cdot (s_j + \Delta s_j)$. The desired condition is then $\sum \ell_{ij} \cdot (s_j + \Delta s_j) = \widetilde{t}_i$. Subtracting the formula $t_i = \sum\limits_j \ell_{ij} v_j$ from this expression, we conclude that the corrections Δs_j must satisfy the following system of (approximate) linear equations: $\sum \ell_{ij} \cdot \Delta s_j \approx \Delta t_i$.

Solving this system of linear equations is not an easy task, because we have many observations and many cell values and thus, many unknowns, and for a system of linear equations, computation time required to solve it grows as a cube n^3 of the number of variables n. So, instead of the standard methods for solving a system of linear equations, researchers use special faster geophysics-motivated techniques (described below) for solving the corresponding systems. These methods are described, in detail, in the next subsection.

Once we solve the corresponding system of linear equations, we compute the updated values Δs_j, compute the new (corrected) slownesses $s_j + \Delta s_j$, and repeat the procedure again. We stop when the discrepancies become small; usually, we stop when the mean square error $\dfrac{1}{n} \sum\limits_{i=1}^{n} (\Delta t_i)^2$ no longer exceeds a given threshold. This threshold is normally set up to be equal to the measurement noise level, so that we stop iterations when the discrepancy between the model and the observations falls below the noise level – i.e., when, for all practical purposes, the model is adequate.

Algorithm for the inverse problem: details. Let us describe, in more detail, how the corresponding linear system of equations is usually solved. In other words, for a given cell j, how do we find the correction Δs_j to the current value of slowness s_j in this cell?

Let us first consider the simplified case when there is only path, and this path is going through the j-th cell. In this case, cells through which this path does not go does not need any correction. To find the corrections Δs_j for all the cells j through which this path goes, we only have one equation $\sum\limits_j \ell_{ij} \cdot \Delta s_j = \Delta t_i$. The resulting system of linear equations is clearly under-determined: we have a single equation to find the values of several variables Δs_j. Since the system is under-determined, we have a infinite number of possible solutions. Our objective is to select the most geophysical reasonable of these solutions.

For that, we can use the following idea. Our single observation involves several cells; we cannot distinguish between the effects of slownesses in different cells, we only observe the overall effect. Therefore, there is no reason to assume that the value Δs_j in one of these cells is different from the values in other cells. It is thus reasonable to assume that all these values are equal to each other: $\Delta s_j = \Delta s_k = \ldots = \Delta s$. Substituting these equal values into the equation $\sum\limits_j \ell_{ij} \cdot \Delta s_j = \Delta t_i$, we conclude that $L_i \cdot \Delta s = \Delta t_i$, where $L_i = \sum\limits_j \ell_{ij}$ is the overall length of i-th path. Thus, in the simplified case in which there is only one path, to the slowness of each cell j along this path, we add the same value $\Delta s = \dfrac{\Delta t_i}{L_i}$.

Let us now consider the realistic case in which there are many paths, and moreover, for many cells j, there are many paths i which go through the corresponding cell. For a given cell j, based on each path i passing through this cell, we can estimate the correction Δs_j by the corresponding value $\Delta s_{ij} \stackrel{\text{def}}{=} \dfrac{\Delta t_i}{L_i}$. When there are several (n_j) paths P_i going through the j-th cell C_j ($P_i \cap C_j \neq \emptyset$), we have, in general, several different estimates $\Delta s_j \approx \Delta s_{ij}$.

Now, we have an over-determined system of equations: we have a single variable Δs_j and as many equations (n_j) as there are paths i going through this cell. Ideally, we would like all the differences $e_i \stackrel{\text{def}}{=} \Delta s_j - \Delta s_{ij}$ to be equal to 0, i.e., we would like the corresponding difference vector $e = (e_i, e_{i'}, \ldots)$ to be equal to 0. In general, the estimates Δs_{ij} are different, so we cannot find a single correction Δs_j which is equal to all of them – and thus, we cannot have all the components of the difference vector e to be equal to 0. Since we cannot make e *exactly* equal to 0, it is reasonable to find the value Δs_j for which the corresponding vector e is as *close* to 0 as possible, i.e., for which the distance $\sqrt{s_i^2 + s_{i'}^2 + \ldots}$ between e and $0 = (0, 0, \ldots)$ is the smallest possible.

Since the square root is a monotonic function, the square root of the expression is the smallest if and only if the expression itself is the smallest. Thus, minimizing the distance is equivalent to minimizing the sum $s_i^2 + s_{i'}^2 + \ldots$ Thus, we arrive at the Least Squares method for solving a system of over-determined equations. For the system $\Delta s_j \approx \Delta s_{ij}$ the solution is well known: the arithmetic average of different estimates:

$$\Delta s_j = \frac{1}{n_j} \sum_{P_i \cap C_j \neq \emptyset} \frac{\Delta t_i}{L_i}.$$

This is the formula used in Hole's algorithm.

Comment. The above formula treats all the paths equally. In reality, some paths may barely touch the cell, while other paths really traverse the entire cell. The paths which barely touch the cell should not have the same influence on the cell as the others. How can we achieve this objective?

In the expression $\Delta t_i = \sum\limits_j \ell_{ij} \cdot \Delta s_j$, the value Δs_j occurs with a coefficient ℓ_{ij}. Thus, for the same accuracy in travel-times Δt_i, the resulting accuracy with which we can determine Δs_j from i-th path is proportional to $1/\ell_{ij}$. In other words, the accuracy σ_i with which $\Delta s_j \approx \Delta s_{ij}$ is proportional to $1/\ell_{ij}$. By applying the Least Square Method to this new system of equations, we conclude that $\sum\limits_i \frac{e_i^2}{\sigma_i^2} \to \min$, i.e., that $\sum\limits_i (\Delta s_j - \Delta s_{ij})^2 \cdot \ell_{ij}^2 \to \min$. Differentiating over Δs_j, we conclude that

$$\Delta s_j = \frac{\sum\limits_j \Delta s_{ij} \cdot \ell_{ij}^2}{\sum\limits_j \ell_{ij}^2},$$

i.e., that instead of a *simple* average, we take a *weighted* average of estimates coming from different paths i, with weights proportional to ℓ_{ij}^2. This method has also been used in solving the inverse problem; it uses slightly more computations on each iteration but, since iterations are somewhat more reasonable, requires fewer iterations.

Successes of the known algorithms. The known algorithms have been actively used to reconstruct the slownesses, and, in many practical situations, they have led to reasonable geophysical models.

Limitations of the known algorithms. As we have mentioned, the inverse problem is often under-determined: we have fewer observations than unknowns. As a result, based on the same measurement results, we may have many different velocity models $v_j = 1/s_j$ that are all consistent with the same measurement results.

The above algorithms selects one of these velocity models; often, however, the velocity model that is returned by the existing algorithm is not geophysically meaningful: e.g., it predicts velocities outside of the range of reasonable velocities at this depth. To get a geophysically meaningful model, a geophysicist tries to adjust the initial approximation – so as to avoid this discrepancy between the actual distribution and the geophysical knowledge.

This adjustment usually requires several iterations. It is a very time-consuming process, because there is no algorithmic way of adjusting the initial data, only heuristic recipes, and as a result, each adjustment requires many time-consuming trial-and-error steps. Moreover, because of the non-algorithmic character of adjustment, it requires special difficult-to-learn skills; as a result, the existing tools for solving the seismic inverse problem are not as widely used as they could be.

What we do in this paper. To enhance the use of the seismic data, it is imperative to make the corresponding tools more accessible and their handling more algorithmic. To achieve this goal, we must incorporate the expert knowledge into the algorithm for solving the inverse problem.

In this paper, we describe how to do it.

3 How to Use Explicit Expert Knowledge: Interval and Fuzzy Information

Main idea. As we have mentioned, one of the reasons why the mathematically valid solution is not geophysically meaningful is that at some points, the velocity is outside the interval of values which are possible at this depth for this particular geological region.

Additional information provided by experts: case of interval information. To take this expert knowledge into consideration, it is reasonable to explicitly solicit, from the experts, the intervals of possible values – and then modify the inverse algorithms in such a way that the velocities are always within these intervals.

To be more precise, for each cell j, a geophysicist provides us with the smallest and largest possible value of slowness for this cell. In other words, for each cell j, the expert provides us with an interval $[\underline{s}_j, \overline{s}_j]$ that is guaranteed to contain the actual (unknown) value of slowness s_j.

We select the initial model $s_j^{(0)}$ that is consistent with this information, i.e., for which $s_j^{(0)} \in [\underline{s}_j, \overline{s}_j]$ for all j. Then, we modify the algorithm form solving the inverse problem in such a way that on all iterations, slownesses always stay within the corresponding intervals.

Comment. For some cells – e.g., in some cells which are close to the surface and for which the geophysical structure is well known – we may even know the *exact* values s_j of slowness. It is worth mentioning that this information can also be expressed in interval terms – by saying that for each of these cells, the geophysicist provides us with a *degenerate* interval $[s_j, s_j]$.

How to use interval information: first idea. To explain our first idea, let us reformulate the problem. Suppose that we start with the slowness values $s_j^{(0)}$ which are within the given intervals: $s_j^{(0)} \in [\underline{s}_j, \overline{s}_j]$. On the next iteration, however, we may get values $s_j^{(0)} + \Delta s_j$ which are outside the corresponding intervals.

For example, we may know the exact values of slownesses in the upper layers (closer to the surface), and these are exactly the values which we select as the initial approximation for the corresponding cells. Since we know the exact slownesses for the surface cells, for paths which only go through these surface cells, the predicted travel-times are close to the measured ones – so no correction is needed. However, for paths which do go through deeper cells as well, we will, in general, need a correction. The existing algorithms evenly spread the resulting correction Δs to all the cells along the path. So, if we follow these algorithms, then, in addition to the deeper cells, we also update slownesses in the upper layer cells. This update changes the slowness values and thus, leads to values outside the given (degenerate) interval $[s_j, s_j]$.

To remedy this situation, we can do the following. For each cell j, after an iteration of, say, Hole's algorithm, we have a corrected value of the slowness $s_j^{(k)}$ which approximates the actual (unknown) slowness s_j: $s_j \approx s_j^{(k)}$. We also know that s_j should be located in the interval $[\underline{s}_j, \overline{s}_j]$. Similar to our previous analysis, it is therefore reasonable to use the Least Squares Method to combine these two piece of information: i.e., we look for the value $s_j \in [\underline{s}_j, \overline{s}_j]$ for which the square $(s_j - s_j^{(k)})^2$ is the smallest possible. In geometric terms, we look for the value within the given interval $[\underline{s}_j, \overline{s}_j]$ which is the closest to $s_j^{(k)}$. Thus:

- If the value $s_j^{(k)}$ is already within the interval, we keep it intact.
- If the value $s_j^{(k)}$ is to the left of the interval, i.e., if $s_j^{(k)} < \underline{s}_j$, then the closest point from the interval is its left endpoint $\underline{s}_j$.
- Similarly, if the value $s_j^{(k)}$ is to the right of the interval, i.e., if $s_j^{(k)} > \overline{s}_j$, then the closest point from the interval is its right endpoint $\overline{s}_j$.

Algorithm resulting from the first idea. We thus arrive at the following modification of the existing iterative algorithms for solving the seismic inverse problem. This modification takes into account that for each cell j, we know an interval $[\underline{s}_j, \overline{s}_j]$ which is guarantee to contain the actual (unknown) value of slowness s_j.

We start with an initial approximation $s_j^{(0)}$ which is consistent with this information, i.e., for which $s_j^{(0)} \in [\underline{s}_j, \overline{s}_j]$ for all j. On each iteration, we first apply the iteration step from the existing algorithms, and then update the resulting values $s_j^{(k)}$ as follows:

- if $s_j^{(k)} < \underline{s}_j$, we replace the value $s_j^{(k)}$ with $\underline{s}_j$;
- if $s_j^{(k)} > \overline{s}_j$, we replace the value $s_j^{(k)}$ with $\overline{s}_j$;
- if $\underline{s}_j \le s_j^{(k)} \le \overline{s}_j$, we keep the value $s_j^{(k)}$.

After this additional step, we perform the next iteration, etc.

Advantage of the first idea. The main advantage of this approach is that on every iteration, we get slownesses which are within the given intervals. Thus, in contrast to the existing algorithms, we always remain within the given intervals and thus, avoid non-physical solutions.

First idea: main problem. The main problem with this approach is that it is still too slow. Let us explain why it may be even slower than the traditional algorithms. We will illustrate this on the same example on which we explained, above, why the un-modified Hole's algorithm can lead us outside the desired interval for s_j. In this example, Hole's algorithm equally distributes the discrepancy Δt_i between all the cells along the i-th path: both surface cells and the deeper cells, as $\Delta s = \Delta t_i / L_i$. After the slowness adjustment of all these cells, we change the original predicted travel-time value of $t_i = \sum\limits_j \ell_{ij} \cdot s_j$ to the new value $t_i + \sum\limits_j \ell_{ij} \cdot \Delta s_j = t_i + \Delta s \cdot L_i = t_i + \Delta t_i$, i.e., to $\widetilde{t}_i$. Thus, on the next iteration, the discrepancy along this path will be much smaller (or even disappear completely).

In the new algorithm, after producing the same correction $\Delta s = \Delta_i / L_i$, we, in effect, dismiss it for all the upper-layer cells – because for these cells, adding this correction will bring us outside the given (degenerate) interval. As a result, after the correction, the resulting addition to the travel-time comes only from the deeper cells, and is is thus equal to $\ell_i \cdot \Delta s$, where ℓ_i is the overall length of all deep portions of the i-th path. Since the path also covers several upper-layer cells, we have $\ell_i < L_i$, thus $\ell_i \cdot \Delta s < L_i \cdot \Delta s = \Delta t_i$; so, a large portion of discrepancy re-appears in the next iteration.

Second idea. To speed up computations, it is desirable not just to dismiss the slowness corrections that lead us outside the given intervals $[\underline{s}_j, \overline{s}_j]$, but also to re-distribute the travel-time discrepancy that remains uncovered as a result of this dismissal. In other words, instead of simply *repeating* each iteration step from Hole's algorithm, we *modify* these steps so as to make computations faster and still remain within given slowness intervals.

On each iteration of the new procedure, we start with the slowness values $s_j^{(k-1)}$ which are within given intervals $[\underline{s}_j, \overline{s}_j]$, and we want to produce corrected values $s_j^{(k-1)} + \Delta s_j$ within these same intervals. The condition $s_j^{(k-1)} + \Delta s_j \in [\underline{s}_j, \overline{s}_j]$ is equivalent to $\Delta s_j \in [\underline{\Delta}_j, \overline{\Delta}_j]$, where $\underline{\Delta}_j \stackrel{\text{def}}{=} \underline{s}_j - s_j^{(k)}$, and $\overline{\Delta}_j \stackrel{\text{def}}{=} \overline{s}_j - s_j^{(k)}$.

If the values Δs_{ij} corresponding to each path i are within the desired interval $[\underline{\Delta}_j, \overline{\Delta}_j]$, then their average is also inside this same interval – and, if we want to use the weighted average instead (as mentioned above), this weighted average is also guaranteed to be within the given interval. Thus, to guarantee that the resulting average corrections are within the given intervals, it is sufficient, for each path i, to produce the distribution of slowness corrections Δs_{ij} which are within the corresponding intervals $[\underline{\Delta}_j, \overline{\Delta}_j]$.

Let us therefore concentrate on a single path i. To better describe the idea, let us assume that $\Delta t_i > 0$ (the case $\Delta t_i < 0$ is similar). In this case, our idea is as follows:

- First, we distribute the time discrepancy Δt_i by the overall length L_i of i-th path, and get the correction value $\Delta s_{ij} = \Delta t_i / L_i$.
- If for all the cells j along the path i, we have $\Delta s_{ij} \le \overline{\Delta}_j$, then we are done.
- Otherwise, if there are some cells for which the correction Δ_i / L_i exceeds the allowed maximum correction $\overline{\Delta}_i$, then for these cells, we use the maximum correction instead.
- These slowness corrections result in the travel-time change of $t_i^{(1)} \stackrel{\text{def}}{=} \sum\limits_j \ell_{ij} \cdot \overline{\Delta}_j$, where the sum is taken only over the cells for which the slowness correction is fixed at the value $\overline{\Delta}_j$; the set of such cells will be denoted by $I^{(1)}$. The remaining discrepancy $\Delta t_i^{(1)} \stackrel{\text{def}}{=} \Delta t_i - t_i^{(1)}$ needs to be distributed

between the remaining cells along the i-th path.

- Then, we divide the remaining discrepancy $\Delta t_i^{(1)}$ between the remaining cells, producing the value $\Delta s_{ij}^{(1)} = \Delta t_i^{(1)}/L_i^{(1)}$, where $L_i^{(1)} = \sum\limits_{j \notin I^{(1)}} \ell_{ij}$ is the total path within the remaining cells.
- If for all the remaining cells j, we have $\Delta s_{ij}^{(1)} \le \overline{\Delta}_j$, then we are done.
- Otherwise, if there are some cells for which the correction $\Delta t_i^{(1)}/L_i^{(1)}$ exceeds the allowed maximum correction $\overline{\Delta}_i$, for these cells, we use the maximum correction instead; let us denote the set of such cells by $I^{(2)}$. These slowness corrections result in the travel-time change of $t_i^{(2)} \stackrel{\text{def}}{=} \sum\limits_{j \in I^{(2)}} \ell_{ij} \cdot \overline{\Delta}_j$.
- The remaining discrepancy $\Delta t_i^{(2)} \stackrel{\text{def}}{=} \Delta t_i^{(1)} - t_i^{(2)}$ is distributed between the remaining cells along the i-th path, etc.

Comment. In general, the above algorithm leads to the assignment of slowness corrections which stay within the desired intervals – and, at the same time, still cover the discrepancy Δt_i. Sometimes, however, even after picking each slowness correction at its highest allowed level $\overline{\Delta}_j$, we will still not cover the observed time discrepancy Δt_i: i.e., $\sum\limits_j \ell_{ij} \cdot \overline{\Delta}_j < \Delta t_i$. This means that the observed travel-times are *inconsistent* with the intervals $[\underline{s}_j, \overline{s}_j]$. This information should be reported back to the experts, so that the experts will be able to adjust their bounds for s_j in such a way that the new bounds will be consistent with the observations.

Advantage of the second idea. The main problem with the *first* idea was that it while it guaranteed the slownesses within the given intervals, this idea required, in general, more iterations than the original algorithm – because:

- in contrast to the original algorithm in which the discrepancy at the next iteration is much smaller than in the previous one,
- in the first idea, the discrepancies may decrease only slightly.

When we use the *second* idea, then on each iteration, we decrease all the discrepancies Δt_i as much as possible. As a result, on the next iteration, the discrepancy (if any) is much smaller than the original one. Thus, the number of iterations is about the same as for the original algorithm – and at the same time, we always get slownesses within the given interval.

So, in terms of the number of iterations, the algorithm based on the second idea is faster than the algorithm based on the first idea – and comparable with the number of iterations in the traditional algorithm for solving seismic inverse problems.

Drawbacks of the second idea. The main drawback of the second idea is that while the *number* of iterations goes down, the number of necessary computations *within* each iteration drastically increases.

Indeed, for each path i, in the original Hole's algorithm, all we need to do is compute the overall length L_i along the i-th path, divide Δ_i by L_i, and then add the resulting slowness correction Δs to all the slowness values. Computing the length L_i – by adding the lengths ℓ_{ij} of the segments within different cells j – requires as many arithmetic operations as there are such cells; let us denote this number by c. Division is 1 operation, and adding Δs to all c slownesses also requires c arithmetic operations. Thus, overall, we need $c + 1 + c = O(c)$ operations.

For the algorithm based on the second idea, we still need $O(c)$ steps in the first round, but we may then need second, third, etc., rounds. On each round, at least one of the new slowness correction is assigned to one of the cells; thus, the number of rounds cannot exceed the overall number of cells c along the path. On the other hand, it is possible that we will need as many rounds as there are cells. So, in the worst case, we need c rounds with $O(c)$ operations in each – to the total of $O(c^2)$ arithmetic operations.

In other words, each iteration may require $O(c^2)$ steps instead of $O(s)$. Paths can be long, up to 100-200 km, and cells can be of 1 km size, so we can have c in hundreds. For such paths, a c times increase (= hundreds times increase) can drastically increase the computation time of each iteration and thus, drastically increase the overall computation time. It is thus desirable to come up with a faster method of reallocating the travel-time discrepancy. Let us describe such a method.

How to use interval information: third idea. We want to find the corrections $\Delta s_j \in [\underline{\Delta}_j, \overline{\Delta}_j]$ for which $\sum\limits_j \ell_{ij} \cdot \Delta s_j = \Delta t_i$. Similarly to the above derivation of the original Hole's algorithm, the resulting system is clearly under-determined: we have a single equation to find the values of several variables Δs_j. Since the system is under-determined, we have an infinite number of possible solutions, so we must select the most geophysical reasonable of these solutions.

Similarly to the above derivation of Hole's algorithm, the idea is that there is no reason to assume that the value Δs_j in one of these cells is different from the values in other cells, so we assume that $\Delta s_j \approx \Delta s_{j'}$ for all j and j'. In Hole's algorithm, we simply assumed that all these approximate equalities are actual equalities, so all the values Δs_j along the path were equal, but that assumption leads to slownesses outside given interval ranges. So, in general, some of these equalities can only be approximately true. As we mentioned in our derivation of Hole's code, this situation can be naturally handled by using the Least Squares Method, i.e., in this case, by selecting, from all the solutions of the above system of equations and inequalities, the values Δs_j for which the objective function $\sum\limits_{j\neq j'}(\Delta s_j - \Delta s_{j'})^2$ takes the smallest possible value.

For simplicity, we can add values $j = j'$ in the sum without changing the value of the objective function – since, for $j = j'$, the corresponding differences $\Delta s_j - \Delta s_{j'}$ are 0s anyway. This objective function can be further simplified if we introduce the notations $E \stackrel{\text{def}}{=} \frac{1}{c}\sum\limits_j \Delta s_j$ and $\delta_j \stackrel{\text{def}}{=} \Delta s_j - E$, so that $\sum\limits_j \delta_j = 0$. In these notations, $\Delta s_j - \Delta s_{j'} = \delta_j - \delta_{j'}$, so

$$\sum_{j,j'}(\delta_j - \delta_{j'})^2 = \sum_j \delta_j^2 + \sum_{j'} \delta_{j'}^2 - 2 \cdot \sum_j \sum_{j'} \delta_j \cdot \delta_{j'}.$$

The last term is the product of the two 0 sums $\left(\sum\limits_j \delta_j\right) \cdot \left(\sum\limits_{j'} \delta_{j'}\right)$, and the first two terms contain $2c^2$ expressions δ_j^2 – these two terms cover each of c squares δ_j^2 the same number of times (namely, $2c^2/c = 2c$ times). Thus, the objective function is equal to $2c \cdot \sum\limits_j \delta_j^2$. So, minimizing the objective function is the same as minimizing the population variance $V = \frac{1}{c} \cdot \sum\limits_j \delta_j^2$ of the population Δs_j.

In case of no linear constraints, there exist efficient algorithms for minimizing variance under interval uncertainty in time $O(c \cdot \log(c))$ (which is faster than $O(c^2)$); see, e.g., [6,10]. Let us show that these algorithms can be modified to the case when have a linear constraint.

Third idea: towards a faster algorithm. Let us consider the case $\Delta t_i > 0$. The case $\Delta t_i < 0$ can be treated similarly.

If we have $\Delta s_j \in [\underline{\Delta}_j, \overline{\Delta}_j)$ for two different values j and j' for which $\Delta s_j \neq \Delta s_{j'}$, then we can replace both values Δs_j and $\Delta s_{j'}$ by their weighted average $\dfrac{\ell_{ij} \cdot \Delta s_j + \ell_{ij'} \cdot \Delta s_{j'}}{\Delta s_j + \Delta s_{j'}}$ without changing Δt_i, and the variance will only decrease. Similarly, if for some j, we have $\Delta s_j < \overline{\Delta}_j$, and for some other $j' \neq j$, we have $\Delta s_{j'} = \overline{\Delta}_{j'} > \Delta s_j$, then we can replace both values by their weighted average and thus, decrease the variance.

Thus, when the variance attains its minimum, all the values Δs_j which are not "maxed out" to $\overline{\Delta}_j$ are equal to each other, and all the maxed-out values do not exceed the common non-maxed-out value. Let p denote the overall number of the values Δs_j which are maxed out when the variance is minimized. So, if we sort the bounds $\overline{\Delta}_1, \ldots, \overline{\Delta}_c$ into a non-decreasing sequence

$$\overline{\Delta}_{(1)} \le \overline{\Delta}_{(2)} \le \ldots \le \overline{\Delta}_{(c)},$$

then the smallest value of the variance is attained when $\Delta s_{(1)} = \overline{\Delta}_{(1)}$, $\Delta s_{(2)} = \overline{\Delta}_{(2)}$, $\ldots$, $\Delta s_{(p)} = \overline{\Delta}_{(p)}$, and $\Delta s_{(p+1)} = \ldots = \Delta s_{(c)} = \delta$. The common non-maxed-out value δ can be found from the condition that $\sum\limits_j \ell_{ij} \cdot \Delta s_j = \Delta t_i$, i.e., that $A_p + \mathcal{L}_p \cdot \delta = \Delta t_i$, where

$$A_p \stackrel{\text{def}}{=} \sum_{j=1}^{p} \ell_{i(j)} \cdot \overline{\Delta}_{(j)} \text{ and } \mathcal{L}_p \stackrel{\text{def}}{=} \sum_{j=p+1}^{c} \ell_{i(j)}.$$

Therefore, $\delta = \dfrac{\Delta t_i - A_p}{\mathcal{L}_p}$. Thus, once we know the value p, we can easily compute the corresponding slowness corrections Δs_j. The only remaining problem is how to find the value p.

To find p, we must use the fact that since only p values Δs_j are maxed out, the $(p+1)$-st value $\Delta s_{(p+1)}$ is not maxed out, so $\delta = \Delta s_{(p+1)} < \overline{\Delta}_{(p+1)}$. Hence, from the above equation for Δt_i, we conclude that $\Delta t_i < S_p \stackrel{\text{def}}{=} A_p + \mathcal{L}_p \cdot \overline{\Delta}_{(p+1)}$. (For $p = c$, a similar inequality holds if we take $\overline{\Delta}_{c+1)} = \infty$.)

Similarly, from the fact that the non-maxed-out value δ should be larger than all the maxed-out ones, we conclude that $\delta \ge \overline{\Delta}_{(p)}$, hence $\Delta t_i \ge A_p + \mathcal{L}_p \cdot \overline{\Delta}_{(p)}$. Let us simplify this condition. By definition,

$$A_p + \mathcal{L}_p \cdot \overline{\Delta}_{(p)} = \sum_{j=1}^{p} \ell_{i(j)} \cdot \overline{\Delta}_{(j)} + \sum_{j=p+1}^{c} \ell_{i(j)} \cdot \overline{\Delta}_{(p)}.$$

By moving the term proportional to $\overline{\Delta}_{(p)}$ from the first into the second sum, we conclude that

$$A_p + \mathcal{L}_p \cdot \overline{\Delta}_{(p)} = \sum_{j=1}^{p-1} \ell_{i(j)} \cdot \overline{\Delta}_{(j)} + \sum_{j=p}^{c} \ell_{i(j)} \cdot \overline{\Delta}_{(p)} = A_{p-1} + \mathcal{L}_{p-1} \cdot \overline{\Delta}_{(p)} = S_{p-1}.$$

So, the two conditions on Δt_i take the form $S_{p-1} \le \Delta t_i < S_p$.

It is easy to check that for every p, when we go from S_{p-1} to S_p, then for every j, the coefficient at $\ell_{i(j)}$ can only increase, so $S_{p-1} \le S_p$. Thus, the sequence $S_0, \ldots, S_c$ is non-decreasing: $S_0 \le S_1 \le S_2 \le \ldots \le S_c$, and p can be found as the only value for which Δ_i belongs to the corresponding subinterval $[S_{p-1}, S_p)$. So, we arrive at the following algorithm for computing slowness corrections for each path.

Third idea: algorithm. We start with the initial slowness values $s_j^{(0)}$ which are within the given intervals $[\underline{s}_j, \overline{s}_j]$.

On each iteration of the new procedure, we start with the slowness values $s_j^{(k-1)}$ which are within given intervals $[\underline{s}_j, \overline{s}_j]$. We then compute, for each cell j, the values $\underline{\Delta}_j = \underline{s}_j - s_j^{(k-1)}$ and $\overline{\Delta}_j = \overline{s}_j - s_j^{(k-1)}$.

Based on these slownesses, we find the paths from the sources to the sensors, compute the predicted travel-times t_i along each path, and the discrepancies $\Delta t_i = \widetilde{t}_i - t_i$.

Let us describe how we compute the correction Δs_j along the i-th path. Once we have computed these corrections for all the paths, then for each cell j, we take the average (or weighted average) of all the corrections coming from all the paths which pass through this cell.

We will consider the case when $\Delta t_i > 0$; the case when $\Delta t_i < 0$ is treated similarly. In this case, we first sort all c values $\overline{\Delta}_j$ along the i-th path into a non-decreasing sequence

$$\overline{\Delta}_{(1)} \le \overline{\Delta}_{(2)} \le \ldots \le \overline{\Delta}_{(c)}.$$

Then, for every p from 0 to c, we compute the values A_p and $\mathcal{L}_p$ as follows:

$$A_0 = 0, \quad \mathcal{L}_0 = L_i, \quad A_p = A_{p-1} + \ell_{i(p)} \cdot \overline{\Delta}_{(p)}, \quad \mathcal{L}_p = \mathcal{L}_{p-1} - \ell_{i(p)}.$$

After that, for each p, we compute $S_p = A_p + \mathcal{L}_p \cdot \Delta_{(p+1)}$, and we find p for which $S_{p-1} \le \Delta t_i < S_p$. Once this p is found, we take $\Delta s_{(j)} = \overline{\Delta}_j$ for $j \le p$, and for $j > p$, we take $\Delta s_{(j)} = \dfrac{\Delta t_i - A_p}{\mathcal{L}_p}$.

When $\Delta t_i < 0$, we similarly sort the values $\underline{\Delta}_j$ into a decreasing sequence, and find p so that the first p corrections are "maxed out" to $\underline{\Delta}_j$, and the rest $c - p$ corrections are determined from the condition $\Delta s_{(j)} = \dfrac{\Delta t_i - A_p}{\mathcal{L}_p}$.

Computational complexity of the new algorithm. Let us show that this new algorithm is indeed faster.

- Sorting c values requires time $O(c \cdot \log(c))$.
- Computing each of c values A_p, $\mathcal{L}_p$, and S_p requires a finite number ($O(1)$) of elementary arithmetic operations, so the overall computation time for this step is $O(c)$.
- Finally, once p is found, we need a finite number of step to compute each of c slowness corrections Δs_j, so we also need $O(c)$ steps.

Overall, we thus need $O(c \cdot \log(c)) + O(c) + O(c) = O(c \cdot \log(c))$ steps.

Case of fuzzy uncertainty. Experts are often not 100% sure about the corresponding intervals. They can usually produce a wider interval $[\underline{s}_j, \overline{s}_j]$ of which they are practically 100% certain, but in addition to that, they can also produce narrower intervals about which their degree of certainty is smaller. As a result, instead of a single interval, we have a nested family of intervals corresponding to different levels of uncertainty – i.e., in effect, a fuzzy interval (of which different intervals are α-cuts).

So, instead of simply saying that a given solution to the seismic inverse problem is satisfying or not, we provide a *degree* to which the given solution is satisfying – as the largest α for which the velocity at every point is within the corresponding α-cut intervals.

How we can use fuzzy uncertrainty. How can we incorporate the fuzzy information into the inverse method? A natural way to do it is as follows: instead of simply getting a solution in which all the slownesses belong to the guaranteed (wide) intervals corresponding to $\alpha = 0$, we try to find the largest possible value α for which all the slownesses belong to the corresponding (narrower) α-cuts.

We can find such α, e.g., by simply trying $\alpha = 0$, $\alpha = 0.1$, $\alpha = 0.2$, etc., until we reach such a value of α that the process no longer converges. Then, the solution corresponding to the previous value α – i.e., to the largest value α for which the process converged – is returned as the desired solution to the seismic inverse problem.

Comment. What we have just described is the basic straightforward way to take fuzzy-valued expert knowledge into consideration. Several researchers have provided other ideas for successfully using fuzzy expert knowledge in geophysical problems; see, e.g., [2,4,12] and references therein. We plan to add some of these ideas to our modified algorithms.

4 How to Use Implicit Expert Knowledge

Implicit knowledge: case of interval uncertainty. In other cases, for each 3-D point, the reconstructed velocity is within the corresponding interval, but the geophysical structure is still not reproduced right. In such cases, it is difficult to explicitly describe, to a computer system, what exactly is wrong, but often, we can describe it implicitly. Namely, the seismic inverse algorithm – like many other algorithms for solving the inverse problem – is based on the assumption that the measured errors are independent and normally distributed. As a result, as a criterion of how well the velocity model fits the measurement results, these algorithms use of mean square error

$$E \stackrel{\text{def}}{=} \sum_{i=1}^{N} (x_i - \widetilde{x}_i)^2,$$

where N is the overall number of measured travel-times, x_i is the i-th travel-time according to the model, and $\widetilde{x}_i$ is the measured travel-time.

For geophysically adequate reconstructions, this mean square error is indeed reasonably small, and the individual differences $x_i - \widetilde{x}_i$ are indeed more or less normally distributed. On the other hand, for geophysically meaningless models, while the mean square error E is also small, several individual differences $|x_i - \widetilde{x}_i|$ are very large in comparison with the others – so that the resulting empirical distribution of these differences is far from normal.

To avoid this problem, it is desirable to require not only that the mean square error be small, but that all individual differences $|x_i - \widetilde{x}_i|$ be small as well. Ideally, we should have an exact upper bound Δ on such a difference, and dismiss a solution as non-physical if at least one of the differences exceeds Δ.

How we can use interval uncertrainty. How can we guarantee that we only get solutions which are physical in the above sense?

- Traditionally, once the mean square error is small, we stop iterations.
- Instead, we propose to continue iterations until all the differences are under Δ.

If this does not happen, then we need to do what traditional algorithms do if we do not get a convergence – restart computations with a different starting velocity model.

Implicit expert knowledge: fuzzy uncertainty. Experts cannot always provide us with exact upper bounds Δ; instead, based on the expert's experience of solving inverse problems, we can have different bounds with different degrees of certainty – i.e., again, in effect, a fuzzy number as an upper bound.

How we can use fuzzy uncertainty. How can we use this fuzzy information? A natural idea is – like in the use of explicit expert knowledge – to find the largest α for which we can decrease all the differences $x_i - \widetilde{x}_i$ into the corresponding α-cut intervals. Similar to the case of explicit knowledge, we can do it, e.g., by trying $\alpha = 0$, $\alpha = 0.1$, etc.

Future work: need for data fusion. Yet another way to detect velocity models that are not geophysically reasonable is to take into consideration other geophysical and geological data, such as the gravity and geological maps, etc. – in other words, fuse several different types of data. Preliminary results of such fusion are indeed very encouraging; see, e.g., [1].

References

[1] R. Aldouri, G. R. Keller, A. Gates, J. Rasillo, L. Salayandia, V. Kreinovich, J. Seeley, P. Taylor, and S. Holloway, "GEON: Geophysical data add the 3rd dimension in geospatial studies", *Proceedings of the ESRI International User Conference 2004*, San Diego, California, August 9–13, 2004, Paper 1898.

[2] G. Bardossy and J. Fodor, *Evaluation of Uncertainties and Risks in Geology*, Springer Verlag, Berlin, 2004.

[3] S. D. Cabrera, K. Iyer, G. Xiang, and V. Kreinovich, "On Inverse Halftoning: Computational Complexity and Interval Computations", *Proceedings of the 39th Conference on Information Sciences and Systems CISS'2005*, John Hopkins University, March 16–18, 2005, Paper 164.

[4] R. Demicco and G. Klir (eds.), *Fuzzy Logic in Geology*, Academic Press, 2003.

[5] D. I. Doser, K. D. Crain, M. R. Baker, V. Kreinovich, and M. C. Gerstenberger "Estimating uncertainties for geophysical tomography", *Reliable Computing*, 1998, Vol. 4, No. 3, pp. 241–268.

[6] L. Granvilliers, V. Kreinovich, and N. Müller, "Novel Approaches to Numerical Software with Result Verification", In: R. Alt, A. Frommer, R. B. Kearfott, and W. Luther (eds.), *Numerical Software with Result Verification*, Springer Lectures Notes in Computer Science, 2004, Vol. 2991, pp. 274–305.

[7] J. A. Hole, "Nonlinear High-Resolution Three-Dimensional Seismic Travel Time Tomography", *J. Geophysical Research*, 1992, Vol. 97, No. B5, pp. 6553–6562.

[8] R. B. Kearfott and V. Kreinovich, "Beyond Convex? Global Optimization Is Feasible Only for Convex Objective Functions: A Theorem", *Journal of Global Optimization* (to appear).

[9] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational complexity and feasibility of data processing and interval computations*, Kluwer, Dordrecht, 1997.

[10] V. Kreinovich, G. Xiang, S. A. Starks, L. Longpre, M. Ceberio, R. Araiza, J. Beck, R. Kandathi, A. Nayak, R. Torres, and J. Hajagos, "Towards combining probabilistic and interval uncertainty in engineering calculations: algorithms for computing statistics under interval uncertainty, and their computational complexity", *Reliable Computing* (to appear).

[11] R. L. Parker, *Geophysical Inverse Theory*, Princeton University Press, Princeton, New Jersey, 1994.

[12] M. Nikravesh, "Soft computing-based computational intelligence for reservoir characterization", *Expert Syst. Appl.*, 2004, Vol. 26, No. 1, pp. 19–38.

[13] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover Publications, Inc., Mineola, New York, 1998.

[14] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*, W. H. Whinston & Sons, Washington, D.C., 1977.

[15] S. A. Vavasis, *Nonlinear Optimization: Complexity Issues*, Oxford University Press, New York, 1991.

[16] C. A. Zelt and P. J. Barton, "Three-dimensional seismic refraction tomography: A comparison of two methods applied to data from the Faeroe Basin", *J. Geophysical Research*, 1998, Vol. 103, No. B4, pp. 7187–7210.

Appendix: Computational Complexity of the Seismic Inverse Problem and Why Traditional Methods of Solving Inverse Problems Do Not Work Well in the Seismic Case

Most inverse problems in science and engineering are ill-posed. The above ill-posedness of the seismic problem is a common feature in applications: most inverse problems in science and engineering are ill-posed; see, e.g., [14].

Smoothness: traditional approach to solving ill-posed inverse problems. A typical way to solve an inverse problem is to find a natural physically meaningful property of actual solution, and use this *a priori* information to select a single most physically meaningful solution among many mathematically possible ones. This process is called *regularization.*

Typically, in inverse problems, this natural property is smoothness. Smoothness can be naturally described in precise mathematical terms. For example, when we reconstruct a 1-D signal $x(t)$, then the degree of smoothness can be defined as follows. At a given moment of time t, the larger the absolute value $|x'(t)|$ of the derivative $x'(t)$, the less smooth the signal is. Thus, at a given time t, the value $|x'(t)|$ is a natural degree of the signal's non-smoothness.

Overall, a natural degree of non-smoothness can be defined as a mean square of these degrees corresponding to different moments t, i.e., as $J \stackrel{\text{def}}{=} \int (x'(t))^2\,dt$.

Most regularization techniques try to find, among many signals that are consistent with given observations, the smoothest signal, i.e., the signal with the smallest possible value of the degree of non-smoothness J.

Smoothness: discrete case. In real life, we only have the values

$$x(t_1), x(t_2), \ldots,$$

of the signal $x(t)$ at discrete moment of time

$$t_1, t_2 = t_1 + \Delta t, \ldots, t_{i+1} = t_i + \Delta t, \ldots$$

Based on this discrete data, we can approximate the derivative $x'(t)$ as a difference

$$\frac{x(t_{i+1}) - x(t_i)}{\Delta t},$$

so minimizing the integral J is equivalent to minimizing the corresponding integral sum

$$J_{\text{discr}} \stackrel{\text{def}}{=} \sum_i (x(t_{i+1}) - x(t_i))^2.$$

Smoothness: 2-D case. For a 2-D velocity distribution $f(n_1, n_2)$, similarly, a natural assumption is that this distribution is smooth. Similarly to the 1-D case, a natural way to describe the degree of smoothness of a given distribution is to use the integral sum

$$J \stackrel{\text{def}}{=} \sum_{n_1, n_2} s(n_1, n_2),$$

where

$$s(n_1, n_2) \stackrel{\text{def}}{=}$$
$$(f(n_1 + 1, n_2) - f(n_1, n_2))^2 + (f(n_1, n_2 + 1) - f(n_1, n_2))^2.$$

Alternatively, we can describe this criterion as the sum of the squares of the differences in intensity between all possible pairs (p, p') of neighboring pixels $p = (n_1, n_2)$ and $p' = (n_1', n_2')$:

$$J = \sum_{p,p' \text{ are neighbors}} (f(p) - f(p'))^2.$$

Smoothness makes problems computationally solvable. A practically useful property of the above degrees of non-smoothness J is that the expression J is a convex function of the signal $x(t_i)$ or $f(n_1, n_2)$. Thus, if the conditions describing the fact that the unknown velocity distributions is consistent with the observations is also described by linear or, more generally, smooth inequalities, then the problem of finding the regularized solution can be reformulated as a problem of minimizing a convex function J on the convex set.

Similarly, if we fix the degree of non-smoothness and look, among all the solutions with a given degree of non-smoothness, for the one that is the closest to the original approximate solution, we also have a problem of minimizing a convex function (distance) on the convex set (of all functions that are consistent with the observations and have the desired degree of smoothness).

It is known that, in general, the problems of minimizing convex functions over convex domains are algorithmically solvable (see, e.g., [15]), and smoothness-based regularization has indeed been efficiently implemented; see, e.g., [14].

For the seismic inverse problem, we only have piecewise smoothness. In geophysics, we have clear layers of different types of rocks, with sharp difference between different layers, so we face an inverse problem with only piecewise smoothness; see, e.g., [11].

Traditional smoothness measures are not adequate for piecewise smoothness. In the piecewise smooth case, the above measure of non-smoothness is not applicable, because it would include neighboring pixels on the different sides of the border between the two layers.

Appropriate smoothness measures for piecewise smoothness case. To avoid the above problem, we need to only take into account the pairs of neighboring pixels that belong to the same zone (layer), i.e., consider the sum

$$J(Z) = \sum_{p,p' \text{ are neighbors in the same zone}} (f(p) - f(p'))^2,$$

where Z denotes the information about the zones. This measure makes computational sense only if we know beforehand where the zones are – i.e., where is the border between the two zones.

However, in real life, finding the border is a part of the problem. In this case, we can use the same smoothness criterion not only to reconstruct the original velocity distribution, but also to find the border location. Specifically, we want to look for the zone distribution *and* for the zone location for which the above criterion J takes the smallest possible value.

In other words, we fix the number of zones, and we characterize the non-smoothness of an velocity distribution by a criterion

$$J^* = \min_{\text{all possible divisions } Z \text{ into zones}} J(Z).$$

The resulting problem is no longer convex. The resulting functional is no longer convex, because the division into zones is a discrete problem. It is known that non-convex problems are, in general, more computationally difficult than the corresponding convex ones (see, e.g., [8]), and adding discrete variables makes the problems even more computationally difficult; see, e.g., [13].

Complexity of piecewise smooth inverse problems. In the following sections, we show that in general, the inverse problem for piecewise smooth case is computationally intractable (NP-hard) even when the relation expressing the consistency between the measured results and the desired velocity distribution is linear.

This proof will follow the proof of NP-hardness of different signal processing problems described, e.g., in [3,9].

Let us prove that in general, the inverse problem for piecewise smooth case is computationally intractable (NP-hard).

Main idea of the proof: reduction to a subset problem. To prove NP-hardness of our problem, we will reduce a known NP-hard problem to the problem whose NP-hardness we try to prove: namely, to the inverse problem for piecewise smooth velocity distributions.

Specifically, we will reduce, to our problem, the following *subset sum* problem [9,13] that is known to be NP-hard:

- Given:
  - m positive integers $s_1, \ldots, s_m$ and
  - an integer $s > 0$,
- check whether it is possible to find a subset of this set of integers whose sum is equal to exactly s.

For each i, we can take $x_i = 0$ if we do not include the i-th integer in the subset, and $x_i = 1$ if we do. Then the subset problem takes the following form: check whether there exist values $x_i \in \{0, 1\}$ for which

$$\sum s_i \cdot x_i = s.$$

We will reduce each instance of this problem to the corresponding piecewise smooth inverse problem.

Reduction to a subset problem: details. Let us consider the following problem. We want to reconstruct an $m \times m$ velocity distribution $f(n_1, n_2)$. Let $d = \lfloor m/2 \rfloor$. We want a piecewise smooth velocity distribution $f(n_1, n_2)$ that consists of two zones.

The following linear constraints describe the consistency between the observations and the desired velocity distribution:

- $f(n_1, n_2) = 1$ for $n_2 > d$;
- $\sum\limits_{i=1}^{m} s_i \cdot f(i, d) = s$; and
- $f(n_1, n_2) = 0$ for $n_2 < d$.

The problem that we consider is to find the solution with the smallest possible value of smoothness J^* among all the velocity distributions that satisfy these linear constraints.

Let us show that the minimum of J^* is 0 if and only if the original instance of the subset problem has a solution.

Indeed, if J^* is 0, this means that all the values within each zone must be the same. Since we have values 1 for $n_2 > d$ and values 0 for $n_2 < d$, we must therefore have every value to be equal either to 0 or to 1. Thus, if we have such a solution, the corresponding values $f(i, d) \in \{0, 1\}$ provide the solution to the original subset problem $\sum s_i \cdot x_i = s$.

Vice versa, if the selected instance of the original subset problem has a solution x_i, then we can take $f(i, d) = x_i$ and get the solution of the inverse problem for which the degree of non-smoothness is exactly 0.

So, if we can solve the inverse problem for piecewise smooth velocity distributions, we will thus be able to solve the subset sum problem.

This reduction proves that the inverse problem for piecewise smooth velocity distributions is indeed NP-hard.