Tu P8 03

Fast Full Wavefield Inversion of Cross-hole Tomographic Data Using Machine Learning Methods

T. Holm-jensen* (Copenhagen University), T. Mejer Hansen (Copenhagen University)

Summary

Linear machine learning algorithms are used to perform a full-wavefield inversion of synthetic cross-hole tomographic data. A general method to invert geophysical data is proposed and tested on a specific problem of cross-hole tomography.

A linear mapping is learned between a tomographic image and the resulting wavefield, so an analytical inversion is possible to obtain the posterior on the tomographic image. This has the advantage that it is extremely fast.

The full wavefield is summarized by traveltimes and Principal Component Analysis (PCA) reduction. The linear mapping is learned using ridge regression. The tomographic images are generated to show a channel structure and the wavefields are generated using finite-difference forward modeling.

The method is shown to perform better than traditional linear ray inversion methods, both qualitatively in that the posterior means have a higher resolution and quantitatively in that there is a higher correlation coefficient between the posterior mean and the true value.

Introduction

First let us define a bit of **geophysical terminology**. If there is a relation g between a physical model $\mathbf{m}$ and some observed data $\mathbf{d}$ of the form:

$$\mathbf{d} = g(\mathbf{m}) \quad (1)$$

then the **forward problem** consists of computing $\mathbf{d}$ given knowledge of $\mathbf{m}$. It is a **linear** forward problem if it can be written as a matrix product $\mathbf{d} = \mathbf{G} \cdot \mathbf{m}$.

The **inverse problem** consists of computing $\mathbf{m}$ given knowledge of $\mathbf{d}$. This is normally the case when interpreting experimental data.

A **probabilistic inversion** is when we are interested in computing the posterior on $\mathbf{m}$

$$P(\mathbf{m}|\mathbf{d}) = K \cdot P(\mathbf{m})P(\mathbf{d}|\mathbf{m}) \quad (2)$$

(the reader might be more familiar with the notation $\sigma_m(\mathbf{m}) = k \cdot \rho_M(\mathbf{m}) L(\mathbf{m})$)

where K is a normalization constant, $P(\mathbf{m})$ is a prior on $\mathbf{m}$ and $P(\mathbf{d}|\mathbf{m})$ is the likelihood of the observed data given the model.

We can solve the probabilistic inverse problem **analytically** if we have (1) a Gaussian prior on $\mathbf{m}$, (2) a Gaussian error and (3) the forward problem is linear (A. (2005)).

It is *very* fast to compute the posterior this way, but the above 3 assumptions are rarely met in practice, especially the assumption of linearity.

This paper suggests approximating *any* g with a linear $\mathbf{G}$ and using this for the analytical inversion.

A **machine learning** algorithm is proposed to find $\mathbf{G}$, so a bit of machine learning terminology is necessary.

First there is the part of a data set called the **training set**, on which we fit n different algorithms to approximate g .

To pick the best algorithm we test these n algorithms on another set called the **validation set** and pick the algorithm with the highest performance. We *cannot* use this performance as a measure of the true performance because we have specifically picked the algorithm that performs the best on the validation set.

To test the quality of the chosen algorithm, we test it on a final set called the **testing set**.

The advantage of using a machine learning algorithm to learn a linear mapping between $\mathbf{m}$ and $\mathbf{d}$ is that we do not need any physical intuition to do so. We just need a lot of examples to learn from.

The main idea of this article is thus to take any g , approximate it by a linear $\mathbf{G}$ and use this to make a linear Gaussian probabilistic inversion.

Method

To put it very generally, I did the following steps:

1. Generate a large data set of examples $\mathbf{m}$ and $\mathbf{d}$.
2. Optional: Reduce the dimensionality of $\mathbf{m}$ to $\tilde{\mathbf{m}}$ using a linear method.
3. Optional: Reduce the dimensionality of $\mathbf{d}$ to $\tilde{\mathbf{d}}$. It does not have to be a linear method.
4. Use a **linear machine learning algorithm** to learn an operator $\mathbf{G}$ mapping $\tilde{\mathbf{m}}$ to $\tilde{\mathbf{d}}$ using the training and validation set.
5. Estimate a Gaussian model to describe the prior on $\mathbf{m}$ using $\mathbf{m}^{(\text{training})}$.
6. Estimate a Gaussian model to describe the modeling error using the validation set.
7. Calculate the posterior on $\mathbf{m}^{(\text{testing})}$ and compare with the true value.

The above method is *very* general. I tested it on the specific problem of interpreting data from Ground-Penetrating Radar cross tomography experiments to reconstruct tomographic images. Cross Hole Tomography is a popular method to explore near-surface structures.

The experiment is the same setup as the setup used in Hansen et al. (2013). An experiment is performed

by boring 2 holes (15 meters deep) in the ground (separated by 5 meters), lowering a source and receiver into each hole and sending an electromagnetic pulse (Ricker wavelet with frequency 0.1 GHz) through the earth. By changing the location of the source and receiver, one can cover the whole area between the holes. The setup has 702 pairs of source-receiver positions.

To generate the data $\mathbf{m}$ (tomographic images), I used the single normal equation simulation algorithm (SNESIM) (Strebelle (2002)) to create 10 000 examples of tomographic images with a channel structure. Examples of these can be seen in figure 1 (left column). The velocity inside a channel is 0.18 m/ns, otherwise 0.1 m/ns. I generated images of size 9 x 15 m (46 x 76 pixels with size 0.2 x 0.2 m). This gives a 2 m padding from the boundaries of the generated image to the source/receiver locations. I used 7 000 images for the training set, 2 000 images for validation set and 1 000 images for the testing set.

To generate the data $\mathbf{d}$ (wavefields), I numerically integrated the wave equation using the 2D leapfrog scheme. The equation integrated was the equation for the magnitude of the electric field of an electromagnetic wave without dissipation. I integrated from 0.0 ns to 80.0 ns using a time resolution of 0.1 ns. Each trace (what a single receiver observes) thus has length 800 and the wavefield has length $702 \cdot 800 = 561600$.

I did not want reflected waves, so I first (1) matched the boundary with an outgoing perpendicular wave and (2) I included a diffusion term near the boundaries of the image. The diffusion constant increased linearly from $0.0 \text{ m}^2/\text{ns}$ (further than 1.5 m from the boundary) to $0.5 \text{ m}^2/\text{ns}$ (at the boundary).

To get a stable integration, I linearly interpolated the tomographic image by a factor 4 in both the x and y -direction, reducing the spatial step size to 0.025 m in the integration.

I chose not **to reduce the dimensionality of $\mathbf{m}$** . A dimensionality of $46 \cdot 76 = 3496$ was not too much to deal with.

I did choose **to reduce the dimensionality of $\mathbf{d}$** , however. It took too much space on my hard disk to save the full wavefield of 10 000 models. I reduced each individual trace from length 800 to 23. This changed the dimensionality of the wavefield from $702 \cdot 800$ to $702 \cdot 23$. Let me denote the data as $\tilde{\mathbf{d}} = [\tilde{\mathbf{d}}_1, \tilde{\mathbf{d}}_2, \dots, \tilde{\mathbf{d}}_{22}, \tilde{\mathbf{d}}_{23}]$, where each $\tilde{\mathbf{d}}_i$ has length 702.

I reduced each trace in the following 3 steps:

1. I found the first arrival of the signal. Let me call this $\tilde{d}_1$. This was done by using the cross-correlation between the first part of the Ricker wavelet and the trace. Then I picked the first time the cross-correlation exceeded 10^{-5} .
2. I cut out 15 ns of the trace directly following the first arrival. This is about the length of the Ricker wavelet used. I used Principal Components Analysis (PCA) (Bishop (2006)) to reduce the dimensionality of this to 7 components. Let me call these $\tilde{d}_2, \tilde{d}_3, \dots, \tilde{d}_8$.
3. The *whole* trace was reduced using PCA to 15 components.

In order to train the PCA schemes, I generated a small extra dataset of only 100 examples of $\mathbf{m}$ and $\mathbf{d}$ to provide the training data.

To learn the forward operator, I used the machine learning algorithm called **ridge regression**.

The method finds a matrix $\mathbf{G}_{\text{ML}}$ that minimizes the following

$$L(\mathbf{G}, \alpha) = \alpha \cdot \|\mathbf{G}\|^2 + \sum_{i=1}^N \|\tilde{\mathbf{d}} - \mathbf{G} \cdot \mathbf{m}\|^2 \quad (3)$$

where the second term is a misfit term (how well it fits the training data) and the first term is a regularization term to avoid overfitting. α is a regularization parameter that must be set before fitting the method. All norms are l^2 -norms.

To determine the value of α , I trained 10 methods on the training set with differing values of α . I used the following values of $\alpha = [10^{-4}, 10^{-3}, \dots, 10^4, 10^5]$ and picked the value of α that gave the best performance (R^2 -score) on the validation set. I fitted one $\tilde{\mathbf{d}}_i$ at a time for a total of 23 fits.

To estimate the Gaussian prior on $\mathbf{m}$, I took the empirical covariance and empirical mean of the training data $\mathbf{m}^{(\text{training})}$. This amounts to retaining all pair-wise correlations but setting all higher order dependencies to the maximum entropy.

To estimate the Gaussian error on $\mathbf{d}$, I made predictions using $\mathbf{G}_{\text{ML}}$, subtracted a systematic error and used the empirical covariance. I assumed that the error on different $\mathbf{d}_i$ were uncorrelated. The empirical errors had a larger tail than a Gaussian, so I added some uncorrelated noise along the diagonal of the estimated covariance matrix equal to the mean of the trace.

Reference Methods

The last thing I did was to compare my method to a couple of reference methods.

Both methods are **straight ray** linear inversion schemes. This assumes that the signal travels in a straight ray from source to receiver and can thus be used to calculate the first arrival $\tilde{\mathbf{d}}_1$.

The first reference method assumes straight rays and a non-correlated error with a standard deviation of 0.8ns. I call this method **Straight Ray**.

The second reference method assumes straight rays but I used the empirical error from the validation set to estimate a Gaussian model of the error. I call this method **Straight Ray (New)**.

Both reference methods use the same prior model as the new method.

Results

The posterior mean (slowness) of a few different models can be seen in figure 1.

Since the reconstructed slowness fields are continuous but the original model is discrete, I created a 1D histogram for the reconstructed velocities inside the channels and a 1D histogram for outside the channels. I then used kernel density estimation (Bishop (2006)) to estimate the probability densities of those 2 distributions. For a new value of velocity, I found the probability of that velocity belonging to a channel. This is plotted in figure 2.

A couple more quantitative measure of the success of the different methods are shown in table 1.

Method	Correlation coefficient	ROC area
Straight Ray	0.49	0.83
Straight Ray (New)	0.66	0.92
Full Wavefield	0.81	0.97

Table 1 Comparison of posterior means of $\mathbf{m}^{(\text{testing})}$ with true value of $\mathbf{m}$. ROC area is the area under a ROC curve (Receiver Operating Characteristic), a higher value is better (MathWorks (2018)).

Conclusions

In this paper I propose a linear full-wavefield inversion scheme to reconstruct tomographic images. Some examples of reconstructed images are shown in figure 1 as well as the pixel-wise probability of the posterior mean to belong to a channel in figure 2.

A much better resolution is seen when using my method. This is also supported by the better measures of quality in table 1 when comparing the methods.

References

- A., T. [2005] *Inverse Problem Theory*. Siam, 3600 University City Science Center, Philadelphia, PA 19104 2688.
- Bishop, C.M. [2006] *Pattern Recognition and Machine Learning*. Springer, 233 Spring Street, New York, NY 10013, USA.
- Hansen, T., Cordua, K., Looms, M. and Mosegaard, K. [2013] SIPPI: a Matlab toolbox for sampling the solution to inverse problems with complex prior information: Part 2, Application to cross hole GPR tomography. *Computers & Geosciences*, **1**(52), 481-492.
- MathWorks [2018] Detector Performance Analysis Using ROC Curves.

Figure 1 Posterior means of $\mathbf{m}$. Far left: True model. Center left: Straight Ray inversion. Center right: Straight Ray (New) inversion. Far right: Full linear wavefield inversion.

Figure 2 Posterior mean probability of pixel belonging to a channel. Far left: True model. Center left: Straight Ray. Center right: Straight Ray (New). Far right: Full linear wavefield.

Strebelle, S. [2002] Conditional simulation of complex geological structures using multiple-point statistics. *Mathematical Geology*, **34**(1), 1-21.