Inadequacy of inverse theory for images

Jon Claerbout and Antoine Guitton

ABSTRACT

Prior information generally enters inverse theory as regularization. In large scale problems such as image estimation where iteration does not continue to completion, an additional way to introduce a prior model is as the starting model. This lesson, hard won at Galilee, is widely applicable.

INTRODUCTION

Reflection seismology is a powerful tool in petroleum prospecting. It works so well it is often pushed beyond reasonable limits. That's when we mortals (mere data processors) get pushed into huge null spaces.

Inverse theory along with least-squares solving technology seduces us into thinking a good solution is at hand as soon we have a data fitting regression along with suitable model and data covariances. A drawn-out, humiliating experience taught us more is needed. The very size of our problems leads to a pitfall that is surely widespread.

Our most valuable datasets bring us into spaces of such high dimensionality we cannot know whether we have iterated long enough. A simple data-fitting problem where we know which answers are plausible and which impossible eventually taught us that what we generally ignore is what we very often need. The years we spent with this simple problem gives us fear that many colleagues, with their more difficult problems, produce solutions that are often wrong!

The convexity of least squares methods along with clever precondioning schemes trick us into feeling our final solution hardly depends on the starting guess. Actually, truth may often be the opposite.

THE CHALLENGING SETTING

An example of the simplest null space is one data value d to be divided into two models, m_1 and m_2. Should 10 be divided into $5+5$ or into $9+1$? This arbitrariness becomes obscure when $\mathbf{m}_1$ and $\mathbf{m}_2$ are families of complicated models competing to grab what's left (if anything) in multivariate data $\mathbf{d}$.

Less comprehensible scenarios in seismology arise when: (1) We gave them a great map of reflectivity, now they want density. (2) We gave them a fine map of velocity,

now they want anisotropy. And a grand challenge, (3) There is a giant the null space between anisotropy and inhomogeneity. How should we characterize it?

The clear academic example examined here points to analogous, but much deeper, industrial examples. We take up a lake survey with a depth sounder. Data is the travel time from the water surface to the water bottom measured at many locations, possibly along survey lines that are somewhat organized. Starting from the presumption the lake surface is perfectly flat, unchanging during the survey, we find apparent survey tracks in our derived image (model) of the water bottom. This astonishes us and forces us to imagine a supplemental model looking like water surface level fluctuating during the surveying.

For 20 years we did not adequately solve the lake problem. The heart of it is that we cannot learn a better model without fully stating our prior model. Model covariances (even if we knew them!) are not enough.

The lake here is known as the "Sea of Galilee." A survey there gave data with complexities in many forms. Here we limit details to those central to the story. Looking back, why did it take us so long? When two different things look the same, they tend to have the same name, even though they differ. That and because theory and practice are worlds apart.

Prior model

There is a boat with a depth sounder, thirty year old navigation gear, and a recorder. Over the course of a season or several, the boat crosses the lake hundreds of times, along somewhat regular tracks, obtaining 131,514 triples (x_i, y_i, z_i), instances of observed depth $\mathbf{d} = (z_i)$ at surface (x_i, y_i) locations. The lake level fluctuates for many reasons (rain and drain?). We have been told (but cannot be certain) the given values of $\mathbf{d} = (z_i)$ have been properly corrected for lake level.

At boat locations (x_i, y_i) we have measured data depth z_i. A model is values of z on a uniform grid in (x, y) space. Linear interpolation (or nearest neighbor extraction) finds modeled depth z anywhere, hopefully a good approximation to z_i at each (x_i, y_i). We express data as $\mathbf{d}_{\text{modeled}} = \mathbf{Gm}$ where $\mathbf{m}$ is water depths on a regular 2-D mesh. The operator (matrix) $\mathbf{G}$ contains mainly information about navigation and interpolation. The navigation values (x_i, y_i) are not treated as data having noise (although we later suspect they should have been).

A number of data fitting issues need not concern us here, issues such as frequent spikes in the depth, zeros in the depth, gaps in the areal coverage, erratic spikes and a few surges in the navigation (x_i, y_i). What you do need to know is that brightness and darkness in Figure 1 do not directly represent depth itself; they represent a roughening of it, like its gradient (actually, after a 2-D operator like a gradient, the helix derivative). This roughening is needed because water bottom features are such small features on the overall trend of water depth.

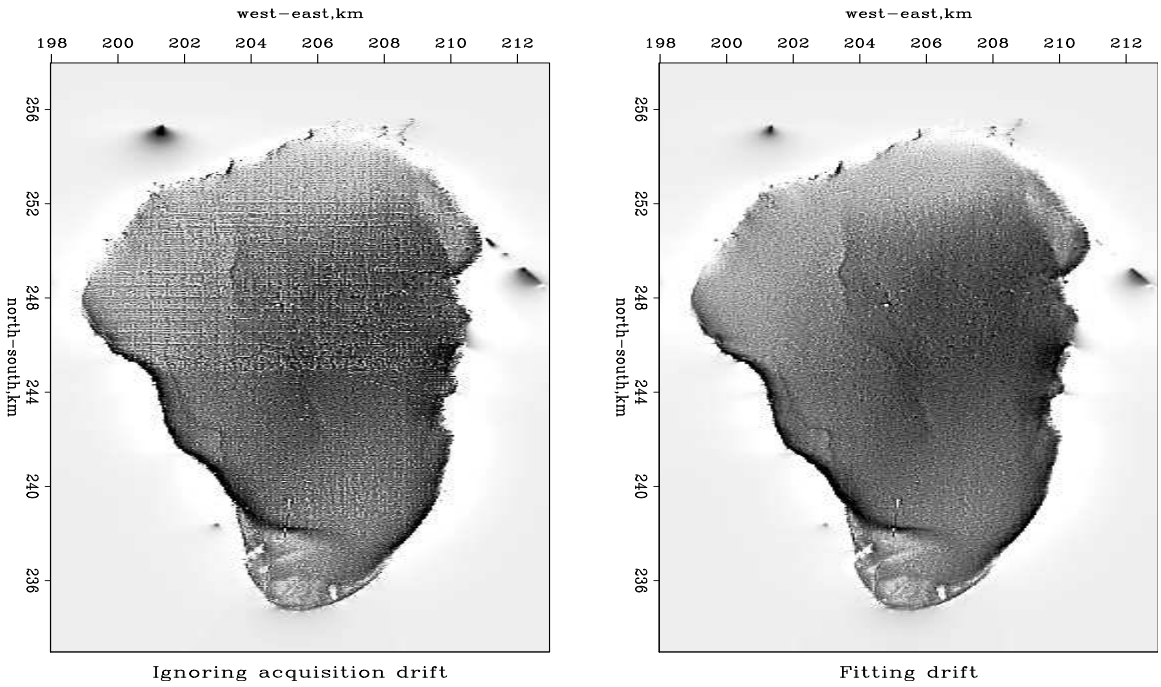

Figure 1: Left, roughened water depth **p** given a flat water surface. Right, roughened water depth **p** after compensating for apparent surface elevation variation.

Correction for apparent water level

Our first idea of water level fluctuation with time (including date) was simply "rain and drain." Since surveying is inactive at night and on holidays we first imagined surface elevation function an assemblage of step functions. We didn't have a ready way of modeling those, so we tried a slowly-variable continuous function of measurement time.

Usually an erroneous low frequency in data is easy to handle by minimizing low-cut filtered residuals. Attempts to do this failed because of the erratic presence of spikes, zeros, and surges throughout the data. So, low-frequency measurement drift had to be modeled instead. Then spikes were easily managed by using an ℓ_1 styled data fitting procedure (hyperbolic penalty).

Modeling the water surface as a function of time led to beautiful track-free images resembling the right side of Figure 1. These images are delightful, but there was an underlying problem to be taken seriously. The water surface elevation **e** crossing the lake looked as sketched in Figure 2. At low frequencies the surface elevation **e** mimicked the water bottom **b**. The data (travel time) was distributing itself between the water bottom **b** and the surface top **e**. The water bulge in the middle of the lake came out unreasonably large. To get the tracks out of the water bottom image, a couple meters of water bulge was required!

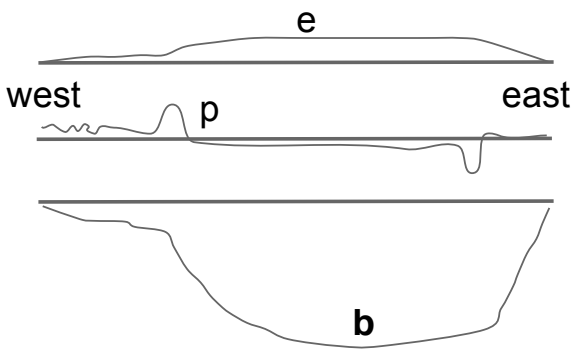

Figure 2: Cartoon of estimated lake surface elevation **e**, roughened bottom **p**, and estimated bottom **b**.

Algebraic foundation

To "debug" the analysis we need to examine the regressions. Physical functions are smooth, both the water bottom map $b(x,y) = \mathbf{b}$ and the water surface elevation $e(t) = \mathbf{e}$. For regularization, $\mathbf{b}$ is roughened with the 2-D operator $\mathbf{A}$, typically a helix derivative (square-root of the FT of a Laplacian) and $\mathbf{e}$ is roughened with a low-cut filter, typically $\mathbf{L}^{-1}$, where $\mathbf{L}$ is leaky integration. The data fitting and two regularizing regressions are:

$$\mathbf{0} \approx_h \mathbf{Gb} + \mathbf{e} - \mathbf{d} \tag{1}$$
$$\mathbf{0} \approx_2 \mathbf{Ab} \tag{2}$$
$$\mathbf{0} \approx_2 \mathbf{L}^{-1}\mathbf{e} \tag{3}$$

The subscript 2 on $\approx_2$ means least squares, while the subscript h on $\approx_h$ means hyperbolic penalty (to soften noise bursts in the data).

A basic notion of statistics is that the regularizations should lead to residuals that tend to be roughly 2-D white (flat spectrum) in the (x,y) plane. Thus, $\mathbf{A}$ should be chosen so that $\mathbf{Ab}$ is roughly white. In practice a new variable $\mathbf{Ab} = \mathbf{p}$ called the preconditioner is introduced. Besides fulfilling theoretical desiderata, the variable $\mathbf{p}$ has interpretive use. It is the variable shown in Figure 1. The second regularization is that the elevation $\mathbf{e}$ be smooth along measurement time. Smoothed white noise $\mathbf{Ln}$ should give us a signal that looks like our preconceived water surface $\mathbf{e}$, so the associated regularization is $\mathbf{0} \approx \mathbf{n} = \mathbf{L}^{-1}\mathbf{e}$.

Changing the formulation from the physical variables $(\mathbf{b}, \mathbf{e})$ to the computational/statistical variables $(\mathbf{p}, \mathbf{n})$, is called "preconditioning." Preconditioning speeds iterative solutions, and it handles matters as statistical theory instructs us. The preconditioned regressions are:

$$\mathbf{0} \approx_h \mathbf{GA}^{-1}\mathbf{p} + \lambda\mathbf{Ln} - \mathbf{d} \tag{4}$$
$$\mathbf{0} \approx_2 \epsilon_b \mathbf{p} \tag{5}$$
$$\mathbf{0} \approx_2 \epsilon_e \mathbf{n} \tag{6}$$

From a purely mathematical point of view, ϵ_b and ϵ_e are infinitesimals. Iteration would then resolve the data fitting before starting on the regularizations. Model space size is roughly $400^2 = 160,000$ while the iteration count is likely under 50. With so few iterations the regularizations seem hardly to come into play, except that they were earlier embedded by the preconditioning.

It might seem the ratio of the two unknown epsilons determines how much of the null space will end out on the water bottom and how much on the top. And, it might seem the parameter lambda λ is merely a scaling factor in the lowpass filter $\mathbf{L}$. But, λ strongly affects the balance of $\mathbf{p}$ and $\mathbf{n}$ in the gradient. The limited iteration count leads to the epsilon ratio being far less significant than the size of λ. Although we can easily include regularizations (5) and (6), at SEP they are generally ignored. What really matters is the data fitting (4). Here Antoine tried keeping the regularizations and found it made no difference.

Originally, we felt leaky integration $\mathbf{L}$ contained the only parameter needed to adjust the spectrum of the elevation, but we soon realized it always had too much short wavelength energy because of the sharp onset of the damped exponential in $\mathbf{L}$. So we switched to its autocorrelation $\mathbf{L}^{\mathrm{T}}\mathbf{L}$ in the low pass relation $\mathbf{e} = \mathbf{L}^{\mathrm{T}}\mathbf{L}\mathbf{n}$. Thus, in practice, our code is iteratively working this lone regression:

$$\mathbf{0} \;\approx_h\; \mathbf{G}\mathbf{A}^{-1}\mathbf{p} \;+\; \lambda\mathbf{L}^{\mathrm{T}}\mathbf{L}\mathbf{n} \;-\; \mathbf{d} \tag{7}$$

Finding the worst source of null space

We have a convex regression in data space. We have two regularizations in model space that we have taken into account. This should not fail, but it does. How does it fail? We can play with λ, and we did. We could find values of λ that were big enough to suppress the tracks in the image, but those values of λ still created giant bulges on the water surface. In other words, the water surface elevation $\mathbf{e}$ visually correlates with the data, both the observed data and the modeled data $\mathbf{G}\mathbf{b}$. In the middle of the lake the estimated elevation is a couple meters above that at the shoreline. Obviously wrong.

Why should we care that it is wrong, and why is it coming out wrong? We are seeing correlation between data made from $\mathbf{b}$ and data made from $\mathbf{e}$, If we cannot prevent apparent correlation within an estimated model $\mathbf{m} = (\mathbf{b}, \mathbf{e})$ where in real life there is none, how can we hope to know when such correlation is real? Imagine a map of seismic velocity correlating with a map of anisotropy. Are the two correlated in geology, or is the correlation a data processing artifact?

Regularization is not the only way to manage a null space. Choosing your starting solution carefully can make a difference—a huge difference. Textbook theory tells us with convex optimization (such as least squares) final solutions are independent of the starting model, but we learn otherwise from nonlinear problems, and we learn otherwise from linear problems that are too large for us to iterate to completion.

Any null space produces no perturbation in the modeled data so it cannot improve the fit to the observed data. Consequently, whatever null space may exist in the starting model will remain there throughout the iterative fitting process. When we included $\mathbf{e}$ in the data fitting, we introduced as many unknown model parameters as we have data values in $\mathbf{d}$, so we certainly know we now have a giant null space.

The starting model is not the Bayes' prior model. The starting model is simply one of many places to start the iterative solver. But, putting our prior model into our starting model assures us whatever null space it may contain will remain in our final solution. Bingo! The ultimately found elevation is shown in Figure 3, hardly an assemblage of step functions suggested by our original rain and drain ideas!

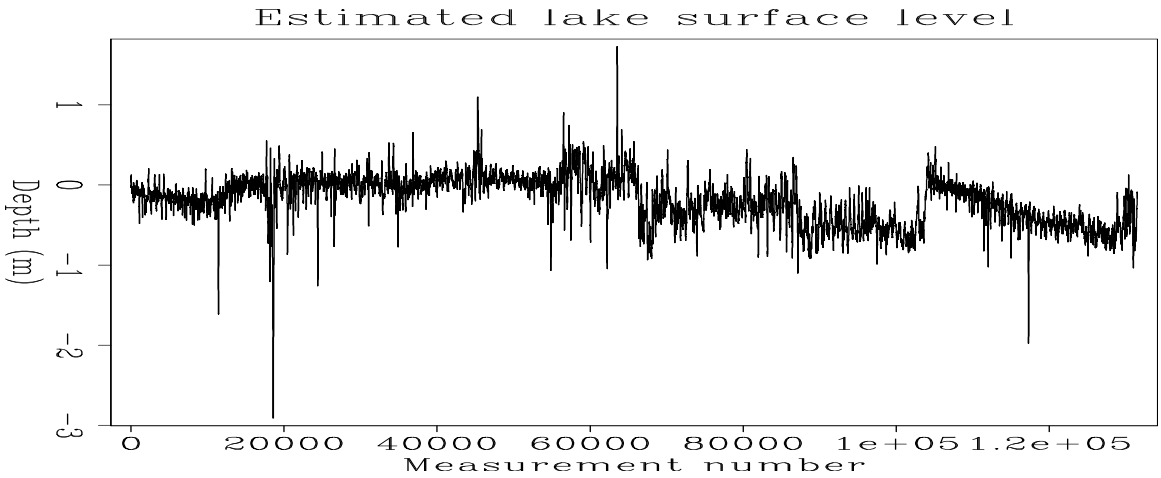

Figure 3: Apparent water surface elevation during the entire Galilee survey (an unknown number of months). Elevation ranges over almost a meter, but is no longer correlated with water depth. The 10cm fuzz corresponds to 10cm measurement z_i discretization. I'm left feeling (x_i, y_i) require a time-variable model, such as systematic navigation errors. We are astonished to see the apparent elevation containing big spikes. This might be explained by surges of navigation error.

Problem solved

So, after 20 years, how did we finally fix the Galilee fitting? We solved two problems, one after the other. Setting $\lambda = 0$ in regression (7)

$$\mathbf{0} \quad \approx_h \quad \mathbf{G}\mathbf{A}^{-1}\mathbf{p} \;+\; \lambda\mathbf{L}^{\mathrm{T}}\mathbf{L}\mathbf{n} \;-\; \mathbf{d}$$

amounts to freezing a flat surface elevation $\mathbf{e} = \mathbf{L}\mathbf{n} = \mathbf{0}$. So, starting from $(\mathbf{p}, \mathbf{n}) = (\mathbf{0}, \mathbf{0})$ with $\lambda = 0$, we ran it getting our first solution for $\mathbf{p}$. This bottom image shows the ship tracks. We use that $\mathbf{p}$ to define $\mathbf{p}_0$ for the second pass. Iterating $(\mathbf{p}, \mathbf{n})$ starting from $(\mathbf{p}_0, (\mathbf{n}_0 = \mathbf{0}))$ gives the final bottom and top $(\mathbf{b}, \mathbf{e})$ estimates. We are free now to experiment with λ without developing the unholy correlation between $\mathbf{b}$ and $\mathbf{e}$.

The various choices of λ represent subjective guesses how to divide the data between bottom $\mathbf{b}$ and top $\mathbf{e}$. There's an opportunity here. Perhaps upon investigating the various λ choices, we might some day find a favorite, then find a reason for the favorite, then understand that reason is suggesting something lacking in the present analytic framework. Maybe we can sniff out an opportunity for building a model that better describes this data. That's real science. Inverse theory is merely a guide to parameterizing known models.

CONCLUSION

Books tell us we should specify prior information in the regularizations. Here we learned that we should specify it in the starting model too, particularly if we cannot iterate to completion. Finding a suitable prior model may, in many cases, be an easier and better way of regularizing. That's what Galilee teaches me. This experience also suggests that high resolution models should generally be derived from lower resolution models (the assumption of scale invariance).

The process that led us to select regularizations ($\mathbf{A}$ and $\mathbf{L}^{-1}$) is highly subjective, sloppy even. We can do better simply by looking at the spectrum of $\mathbf{n}$ and $\mathbf{p}$. If they are not white, then adjusting the preconditioners to achieve it. This should have been done, but was not. If it had, would the overall story change? We don't know.

Our personal opinion is that our fellow image estimators get wrong answers much of the time, and they don't know it because they are not working on easy problems like Galilee, where "wrong answers" are easier to recognize.

What about velocity anisotropy?

Have we learned something from Galilee that we can carry over to estimating anisotropy? We think so, but have neither audacity nor time to explore details. You would be right to guess that we would start by solving for isotropic inhomogeneous material getting $\mathbf{m}_0$. We would use that $\mathbf{m}_0$ as the starting model in iterative fitting simultaneously for inhomogeneity and anisotropy. The regularization in the second problem would be minimizing filtered $\mathbf{m} - \mathbf{m}_0$.

Reference

This article is an adaptation of section 6.6 of the 2014 textbook "Geophysical Image Estimation by Example" by Jon Claerbout with Sergey Fomel. That book chapter is devoted to optimization with the hyperbolic penalty function using the Galilee dataset as an example. Many more details of the example are found there. The book is freely available at the website of Jon Claerbout along with his previous books.

ACKNOWLEDGEMENT

We thank Zvi Avraham for supplying the data. Jon wrote the words in this article, and prepared a video. Antoine cheerfully and enthusiastically persevered through the many versions of the computations.