Estimation theory in reflection seismology

Jon Claerbout, Stanford University. Prepared for the Optical Society of America. [1]

SYNOPSIS OF OIL INDUSTRY SEISMOLOGY

The international oil industry spends about \$4 billion/year acquiring reflection seismic data. A large survey collects a terabyte (10^{12}) of data which is computer processed to a pixel volume of $(10^3)^3 = 10^9$ bytes. At sea the energy source is usually an air gun while on land, the source is mostly buried dynamite, and sometimes, multiple trucks carrying sweep-frequency ground vibrators. At sea, a shot is fired every 10 seconds; echos are recorded along 6 km cables at about a thousand locations, each channel recording a signal of about 2000 floating point values. A typical marine survey contract whose result is shown in Figure 1, lasts a month or more and costs upward of \$10M.

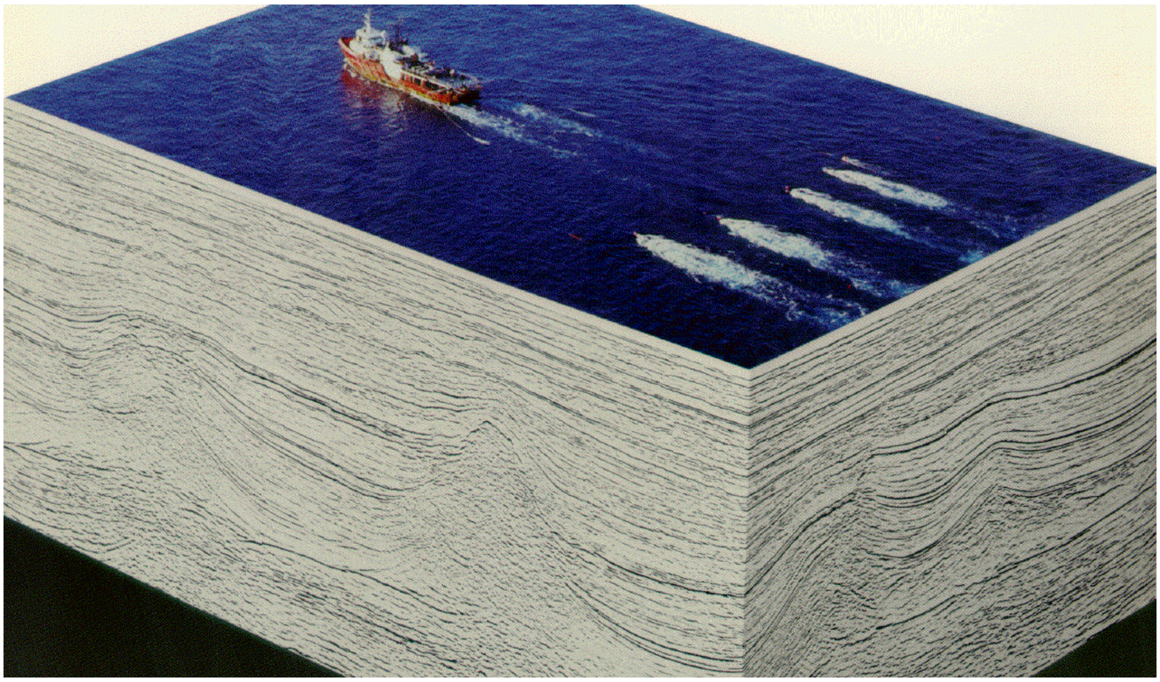

Figure 1: 10 km earth reflectivity cube (ship magnified 100×). (courtesy GSI)

Since the cost of solving a dense matrix grows as the cube of the number of unknowns and gets burdensome at about 10^3 unknowns, many approximations are necessary for the 10^9 unknowns in Figure 1.

[1] **email:** jon@geo.stanford.edu, http://sepwww.stanford.edu/sep/jon/, Geophysics Dept 2215, Stanford, CA 94305, phone 650-723-3717, FAX 650-723-0683. Please enjoy my five free textbooks at the web address.

Familiar formulas from inverse theory such as

$$\hat{\mathbf{m}} = (\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}'\mathbf{d}$$

or

$$\hat{\mathbf{m}} = \mathbf{A}'(\mathbf{A}'\mathbf{A})^{-1}\mathbf{d}$$

must have the inverse matrix approximated by an identity, a diagonal, or a band matrix (hence industrial allergy to the word "inversion"). Reality is that the images are created with weights and filters both before and after applying the adjoint, $\mathbf{A}'$. No one would ever express the basic operator $\mathbf{A}$ as a matrix, for it would have 10^{9+12} elements.

The most basic and widely used imaging techniques involve hyperbolas. An early stage of most data processing reduces the dimensionality of the data by stacking (summing over shot-receiver separation). The summing is done with time shifts (called Normal MoveOut (NMO) and Dip Moveout (DMO)) that attempt to mimic zero-offset signals from the nonzero-offset ones. (Offset is the shot-receiver separation.) This enhances signal-to-noise ratio and reduces the data dimensionality to that of the final image cube, $(10^3)^3$.

An impulse response in the exploding-reflector model (see Figure 2) is a point at

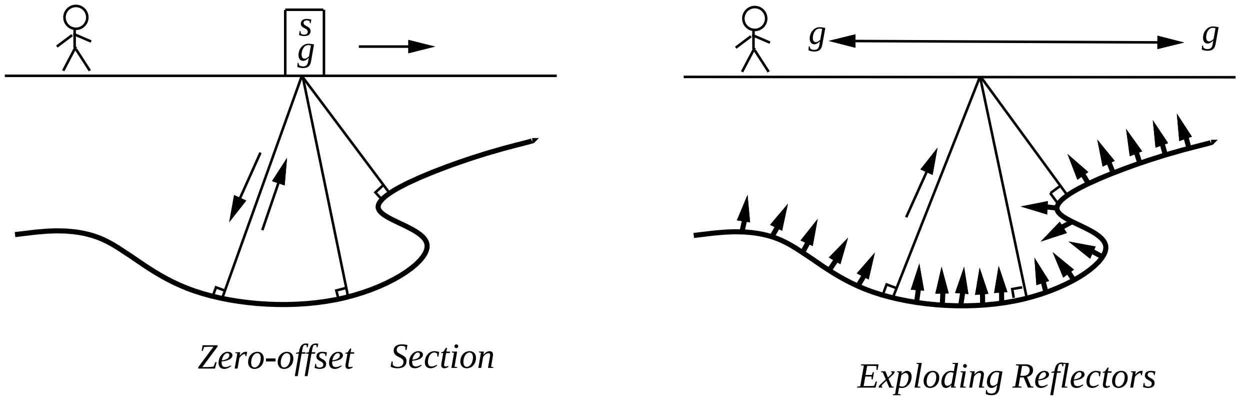

Figure 2: The exploding-reflector concept: Wrong, but a good start. Says a collection of echo soundings from many unconnected shots at all earth surface locations is equivalent to a single thought experiment, one where all reflectors explode at time t=0. Need to scale time by a factor of two.

(x_0, y_0, z_0) in the earth making a spherical wave

$$v^2t^2 = (z - z_0)^2 + (x - x_0)^2 + (y - y_0)^2$$

seen at the earth surface $z = 0$ which is a hyperbola of revolution around the t-axis. The impulse response, a delta function on this hyperboloid, is a column vector in the earth response matrix $\mathbf{A}$. Our approximation to an inverse is the transpose matrix, or adjoint $\mathbf{A}'$. Thus we sum all values on the hyperboloid to find each point in the earth at the location of the exploding reflector. Considering all points in the earth, this summing gives us the image of reflectivity in Figure 1.

Fourier transforms are useful over the time axis because the earth's velocity and reflectivity are time invariant. Unfortunately, the space axes are more troublesome because the velocity varies rapidly with depth and somewhat rapidly horizontally.

Before forming the reflectivity image cube of Figure 1, we need the earth velocity as a function of (x, y, z), since in reality we will need a traveltime function that is considerably more accurate than the constant velocity expression

$$v^2t^2 = z_0^2 + (x - x_0)^2 + (y - y_0)^2$$

. We often make the "Dix approximation" which (in 2-D) says that the observed data can be fit to the traveltime equation

$$t^2 = \tau^2 + h^2/v(\tau)^2$$

where h is the offset and where τ is the vertical traveltime depth. To do this data fitting, we sum the CMP gather over this trajectory for many constant values of v and then for each τ, select the maximum sum (actually, the maximum coherence). See Figure 3. The function $v(\tau)$ that emerges, estimates the Root Mean Square (RMS) integral of the earth's velocity from the surface to the reflector point beneath. Given these values of RMS velocity, differentiation gives us local velocity as a function of depth, and midpoint (horizontal location).

Most interpretors of reflection seismic data ignore the shear waves and handle the seismic data as though it were from a single scalar wave equation for the compressional waves. Thus you might expect that we would set up an inverse problem to seek the density and the velocity. This is not the case. We can find the reflectivity (impedance gradient) and velocity but not the density. Here is why: Our most unambiguous measurement is reflectivity between about 10-50 Hz. Our measurements of velocity do not measure velocity directly, but its integral from one significant reflector to a deeper one. See Figure 4 Where we are lucky, there are coherent reflectors almost everywhere, but the process of subtracting the integrated velocity from one layer to the next deeper one introduces so much error that our measurement of velocity must be regarded as having a scale significantly larger than the wavelengths of our 10-50 Hz waves. Expressing the velocity as a function of vertical travel time, the bandwidth of our velocity measurements is well below 10 Hz. Thus we measure reflectivity from 10-50 Hz and velocity below 10 Hz, so we cannot divide the velocity out of the impedance to find the density. Reflectivity gives us shapes of objects but tells us little about what they are made from. For this we study polarity and amplitude versus angle, but these are not especially reliable. Where possible we use logs of a not-too-distant well.

Some smaller problems that some geophysicists attack with inverse theory are: (1) suppress strong multiple reflections (2) estimate large lateral velocity changes in the shallow weathered layers. (3) interpolate spatially aliased data.

Our most exciting advance for the coming decade promises to be what we grandly call 4-D, what in reality is merely subtracting 3-D surveys done about a year apart. In several pilot studies, 4-D has given a reliable indication of where the fluid pressure has changed because of gas going in or out of solution with the liquid. It is the key to hydrocarbon reservoir delineation and management.

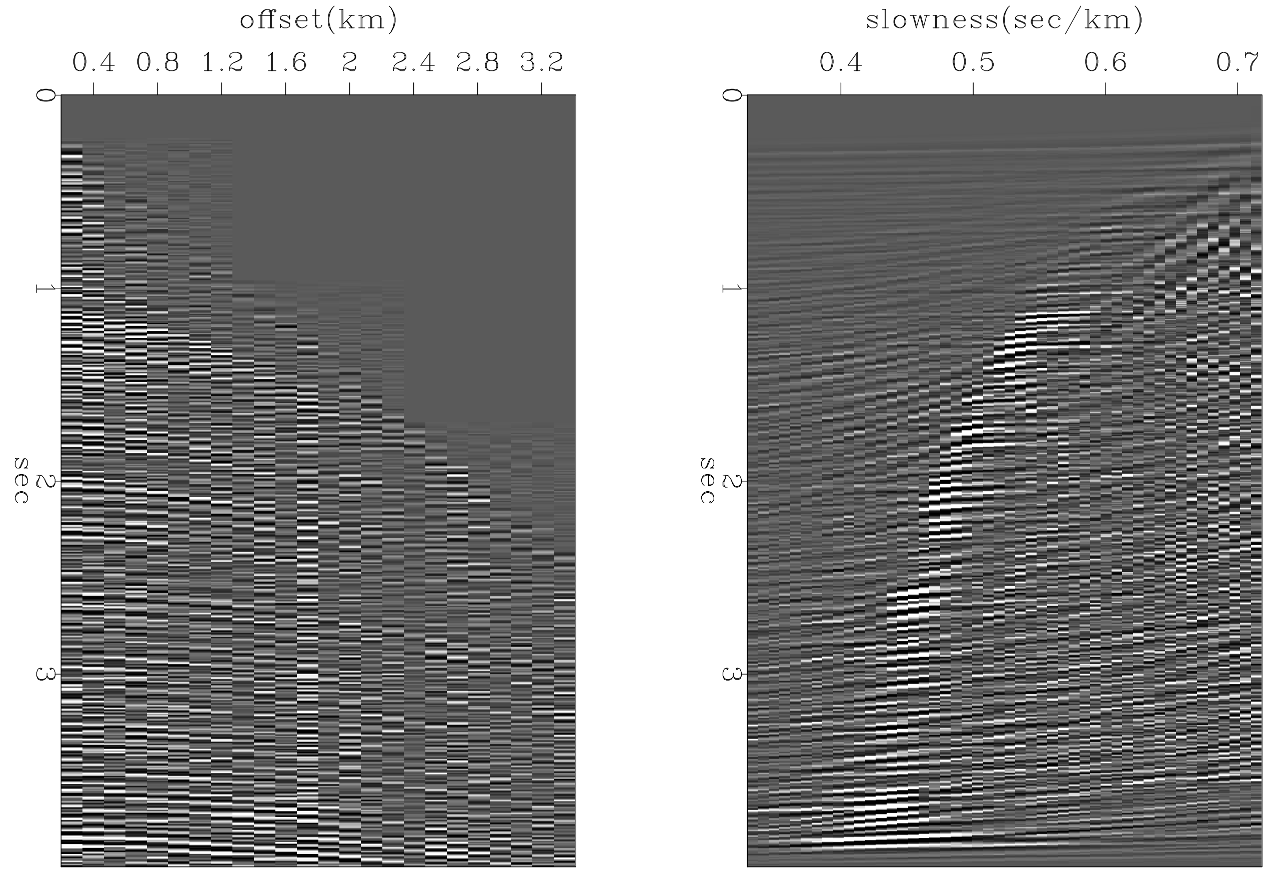

Figure 3: A useful simplification from which to begin velocity analysis is that the earth is nothing but horizontal layering. We do better than that, by gathering all the signals with a given Common MidPoint between the shot and the receiver (called a CMP gather) and observe how the CMP data depends on offset h. Left is a CMP gather (recorded signal as a function of (h, t)) and right is the velocity scan (sum over h at different values of velocity v). We see that earth velocity increases with depth.

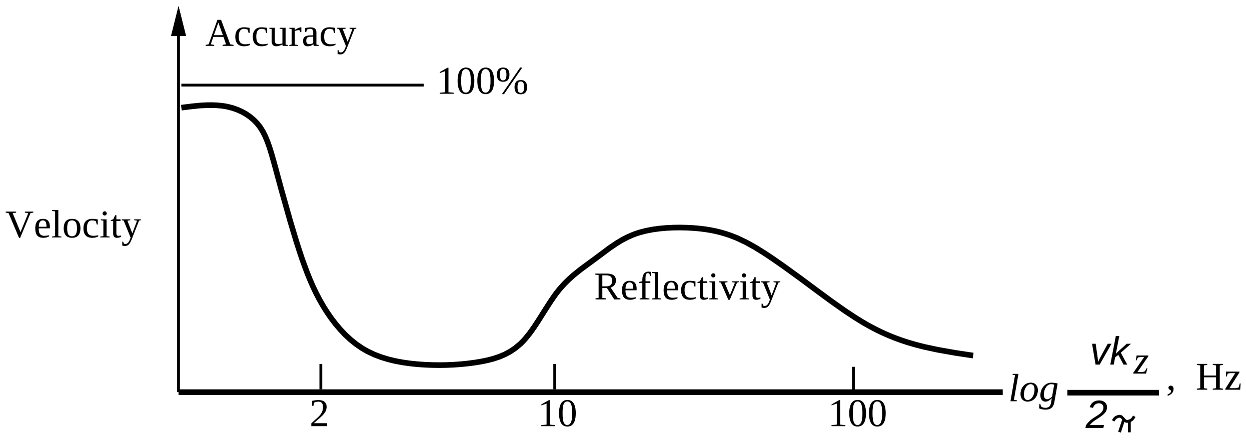

Figure 4: We measure velocity in a different spatial frequency band than we measure reflectivity (impedance) hence we cannot reliably determine density. Because the bandwidth for velocity is so much lower than for reflectivity, it is more realistic for statistical estimation theory.

MULTIDIMENSIONAL RECURSIVE FILTERS VIA A HELIX

Only a small nonstandardized portion of industrial codes in reflection seismology use inverse theory. In an effort to attack the N^3, $N = 10^3$ exhaustion limitation of textbook inverse theory, while seeking an all-purpose preconditioner, I recently happened upon a discovery of a very fundamental nature with many applications that my students have begun working on. Now we shift gears completely.

This section [2] introduces and expounds a basic principle that promises wide-ranging practical applications: A multiple-dimensional Cartesian space can very often be analyzed as a one-dimensional space. Many geophysical image and volumetric estimation problems appear to be multidimensional, but actually they can be handled much more simply. The idea is to use the helical coordinate system. This paper concentrates on explaining the helix idea and it concludes by pointing towards applications in 3-D migration, velocity estimation, and antialiasing.

To see the tip of the iceberg, consider this example: On a two-dimensional Cartesian mesh, the function $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

has the autocorrelation $\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$.

[2] jon@geo.stanford.edu. http://sepwww.stanford.edu/sep/jon/helix.ps Announced through NA Digest (electronic), February 1, 1998, Volume 98:04. Submitted to Geophysics, January, 1998. Presented as poster S32A-11 at the meeting of the American Geophysical Union in San Francisco on December 10, 1997. Previously presented at the August 3-7, 1997 summer research workshop of the Society of Exploration Geophysics in Vail, CO. Previously presented at the May 11-14, 1997 meeting of the sponsors of the Stanford Exploration Project.

Likewise, on a one-dimensional Cartesian mesh,

the function $\mathbf{b} = (1, 1, 0, 0, \cdots, 0, 1, 1)$

has the autocorrelation $\mathbf{r} = (1, 2, 1, 0, \cdots, 0, 2, 4, 2, 0, \cdots, 1, 2, 1)$.

The numbers in the one-dimensional world are identical with the numbers in the two-dimensional world! This correspondence is no accident.

The correspondence between 2-D and 1-D is not merely a mapping between the 2-D and 1-D spaces and autocorrelations in those spaces. It is also a correspondence for filtering. Figure 5 shows some two-dimensional shapes that are convolved together. The left panel shows an impulse-response function, the center shows some impulses, and the right shows the superposition of responses. A surprising, indeed amazing, fact is that Figure 5 was

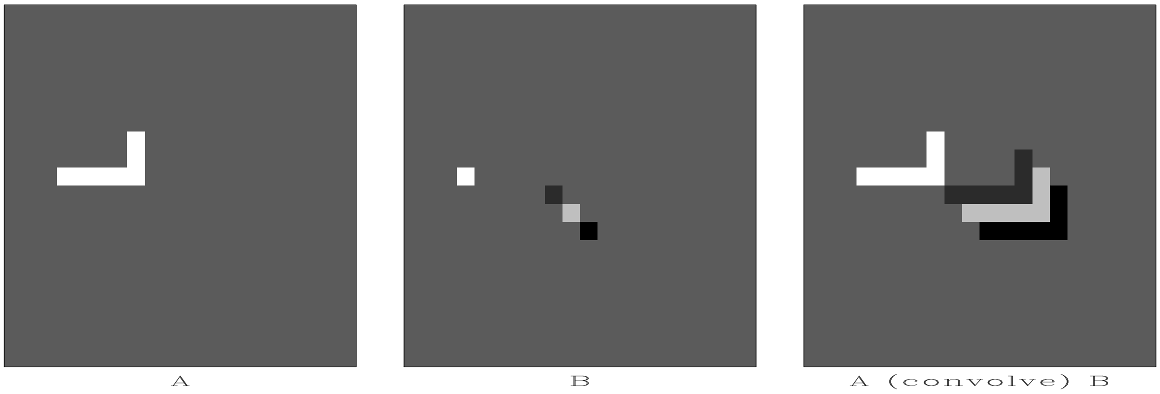

Figure 5: Two-dimensional convolution (as performed in one dimension).

not computed with a two-dimensional convolution program. It was computed with a one-dimensional computer program. It could have been done with anybody's one-dimensional convolution program, either in the time domain or in the Fourier domain. This magical trick is done with the helical coordinate system.

The correspondences above are not limited to two dimensions. They also link multi-dimensional space to one-dimensional space. Although we can perform multidimensional filtering and spectral operations in one dimension, the gain is only conceptual. The cost is the same.

Breakthroughs arise, however, in concept and in computation when we inspect deconvolution, known to engineers as feedback filtering or recursive filtering, known to numerical analysts as back substitution, and known to physicists as finite differences for partial-differential equations (in Cartesian coordinates). Recursion is a powerful operation but it can be dangerous. With recursion, stability always needs to be considered. Multidimensional recursion would be devoid of power without stability theory and the useful techniques that go with it. Fortunately, the correspondence between one and many dimensions gives us the needed stability theory and techniques.

The helix filtering idea

Computational work on a 2-D Cartesian mesh sometimes requires a Fourier transform on one of the axes. A byproduct is that the original plane becomes a cylinder—no real problem when the diameter of the cylinder is large enough. The helix idea involves a similar compromise. Take the Cartesian mesh to be a collection of columns. Instead of connecting the bottom of each column to its top getting a cylinder, the helix connects the bottom of one column to the top of the next. The columns all connect into a supercolumn, a one-dimensional space. The mesh can be parameterized either as a 1-D space or as a 2-D space. The first surprising thing is what the helix does to convolutions and partial differential equations.

A basic idea of filtering, be it in one dimension, two dimensions, or more, is that you have some filter coefficients (finite-differencing star) and some sampled data; you pass the filter over the data; at each location you find an output by crossmultiplying the filter coefficients times the underlying data and then summing the terms. Figure 6 shows a helical mesh for 2-D data on a cylinder. Darkened squares depict a 2-D filter shaped like the Laplacian

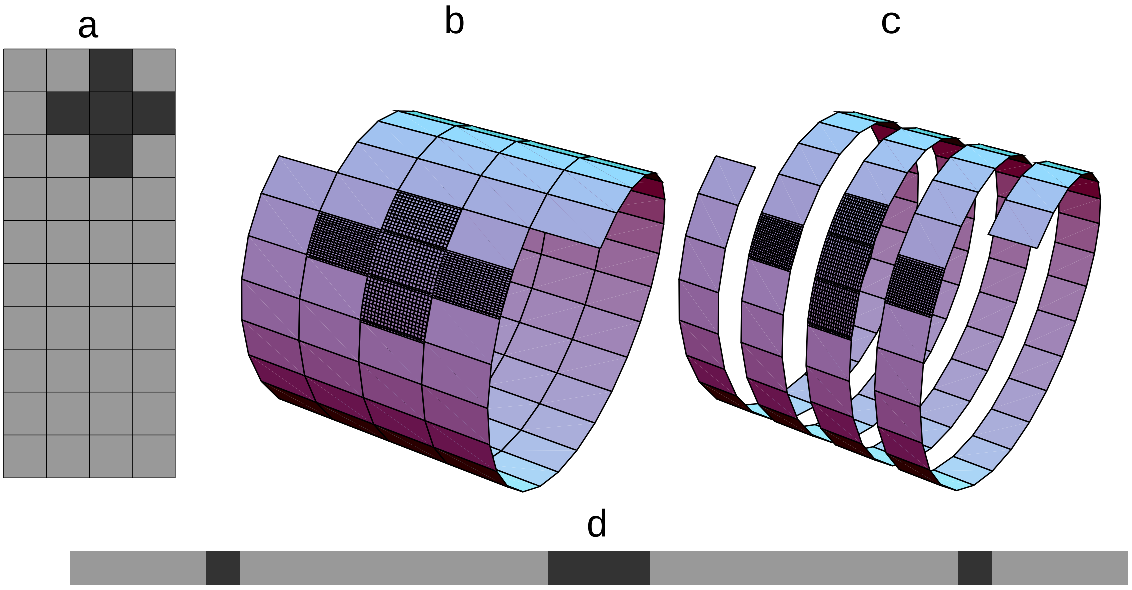

Figure 6: Filtering on a helix. The same filter coefficients overlay the same data values if the 2-D coils are unwound into 1-D strips.

operator. The input data, the filter, and the output data are all on helical meshes all of which could be unrolled into linear strips. A compact 2-D filter like a Laplacian, on a helix is a sparse 1-D filter having long empty gaps.

Since the values output from filtering can be computed in any order, we can slide the filter coil over the data coil in any direction. The order that you produce the outputs is irrelevant. You could compute the results in parallel. We could, however, slide the filter over the data in the screwing order that a nut passes over a bolt. The screw order is the same order that would be used if we were to unwind the coils into one-dimensional strips and convolve them across one another. The same filter coefficients overlay the same data values if the 2-D coils are unwound into 1-D strips. We can do 2-D convolution with a 1-D convolution program.

Convolution creates an output q_t from an input p_t and a filter b_τ with

$$q_t \quad = \quad p_t + \sum_{\tau>0} b_\tau p_{t-\tau} \tag{1}$$

(I use the convention that $b_0 = 1.0$). Deconvolution (polynomial division) can undo convolution (polynomial multiplication) by backsolving, by simply rearranging the terms

$$p_t \quad = \quad q_t - \sum_{\tau>0} b_\tau p_{t-\tau} \tag{2}$$

Deconvolution is recursive filtering. Recursive filter outputs cannot be computed in parallel, but must be computed sequentially as in one dimension, namely, in the order that the nut screws on the bolt (because an ordinary filter makes outputs from past *inputs* whereas a recursive filter makes them from past *outputs*).

Recursive filtering sometimes solves big problems with astonishing speed. It can propagate energy rapidly for long distances. Unfortunately, recursive filtering can also be unstable. The most interesting case, near resonance, is also near instability. There is an old textbook literature e.g. Claerbout (1976) or (1992) with extensive technology for recursive filtering in one dimension. The helix allows us to apply that technology to two (and more) dimensions. It is a wonderful insight. We could not previously do 2-D deconvolution because we had no stability theory for it. We cannot simply use polynomial division to undo the 2-D Laplacian operator for example, because the output will diverge. Poles and zeros tell us about 1-D stability but we don't have them for 2-D polynomials.

In 3-D we simply append one plane after another (like a 3-D Fortran array). It is easier to code than to explain or visualize a spool or torus wrapped with string, etc.

Examples of simple 2-d recursive filters

Figures 7 and 8 contain this 2-D filter

$$\begin{bmatrix} 0 & -1/4 \\ 1 & -1/4 \\ -1/4 & -1/4 \end{bmatrix} \tag{3}$$

Let us experiment using this 2-D filter as a recursive filter. In Figure 7 the input is shown on the left. This input contains two copies of the filter (3) near the top of the frame and some impulses near the bottom boundary. The second frame in Figure 7 is the result of deconvolution by the filter (3). Notice that deconvolution turns the filter itself into an impulse, while it turns the impulses into comet-like images. The use of a helix is evident by the comet images wrapping around the vertical axis.

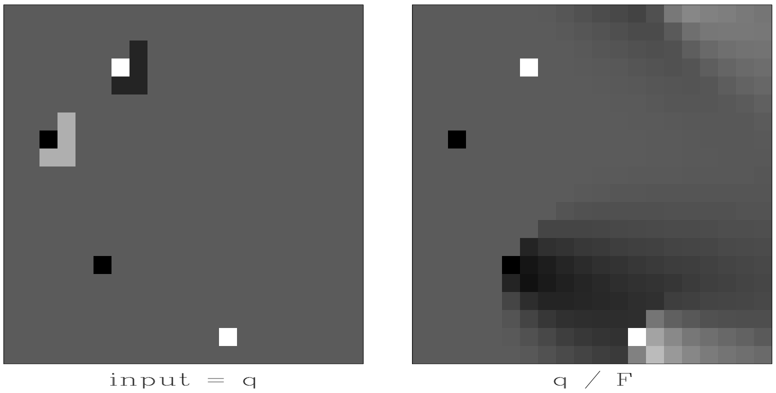

Figure 7: Illustration of 2-D deconvolution. Left is the input $\mathbf{q}$. Right is after deconvolution $\mathbf{q}/\mathbf{F}$ with the filter (3).

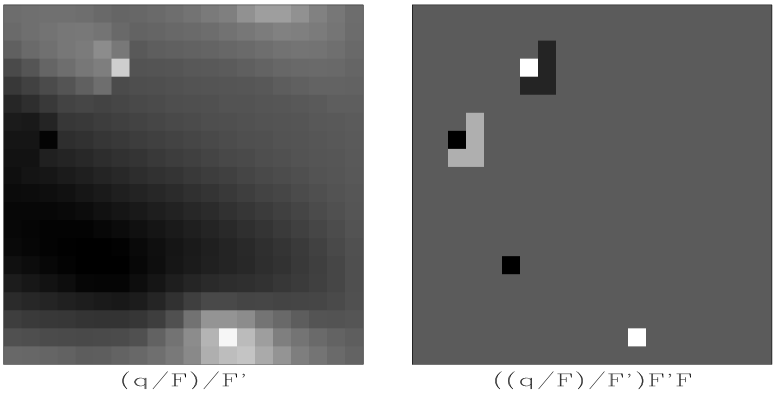

Figure 8: Left is $(\mathbf{q}/\mathbf{F})/\mathbf{F}'$ and right is $((\mathbf{q}/\mathbf{F})/\mathbf{F}')\mathbf{F}'\mathbf{F}$.

Using the result of Figure 7 as an input, the first frame in Figure 8 results from backwards recursion, filtering backwards along the helix, the adjoint operator $\mathbf{F}'$. The adjoint blows energy leftward. Filtering forward and backwards has implemented a symmetrical smoothing operator $(1/\mathbf{F})/\mathbf{F}'$. The last frame in Figure 8 shows that the two deconvolution smoothers can be inverted by convolution, convolving once with filter (3) and once with its adjoint (reverse on both axes). We see we are back where we started. No errors, no evidence remains of any of the boundaries where we have wrapped and truncated. Using the helix, we find that one-dimensional polynomial division is a rapidly invertible *multidimensional* transformation, one with many adjustable parameters.

Finite differences on a helix

The function

$$\mathbf{r} = (-1, 0, \cdots, 0, -1, 4, -1, 0, \cdots, 0, -1) \tag{4}$$

is an autocorrelation function. It is symmetrical about the "4" and its Fourier transform is positive for all frequencies. Digging out an old textbook (Claerbout, 1976), we discover how to compute a causal wavelet with this autocorrelation. I used the "Kolmogoroff spectral-factorization method" to find this wavelet $\mathbf{h}$:

$$\mathbf{h} \quad = \quad (1.791, -.651, -.044, -.024, \cdots, \cdots, -.044, -.087, -.200, -.558) \tag{5}$$

Wind the signal $\mathbf{r}$ around a vertical-axis helix to see its two-dimensional shape $\mathbf{R}$:

$$\mathbf{R} \quad = \quad \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} \tag{6}$$

This 2-D filter is is the negative of the finite-difference representation of the negative of the Laplacian operator, generally denoted $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. Now wind the signal $\mathbf{h}$ around the same helix to see its two-dimensional shape $\mathbf{H}$:

$$\mathbf{H} \quad = \quad \begin{bmatrix} & & & & 1.791 & -.651 & -.044 & -.024 & \cdots \\ \cdots & -.044 & -.087 & -.200 & -.558 & & & & \end{bmatrix} \tag{7}$$

In the 2-D representation (7) we see the coefficients diminishing rapidly away from maximum value 1.791. My claim is that the 2-D autocorrelation of (7) is (6). You verified this idea at the beginning of this paper where the numbers were all ones. You can check it again in a few moments if you drop the small values, say 0.2 and smaller.

Since the autocorrelation of $\mathbf{H}$ is $\mathbf{H'H} = \mathbf{R} = -\nabla^2$ is a *second* derivative, the operator $\mathbf{H}$ must be something like a *first* derivative. As a geophysicist, I found it natural to compare the operator $\frac{\partial}{\partial y}$ to $\mathbf{H}$ by applying them to a local topographic map. The result shown in Figure 9 is that $\mathbf{H}$ enhances drainage patterns whereas $\frac{\partial}{\partial y}$ enhances mountain ridges.

Our construction makes $\mathbf{H}$ have the energy spectrum $k_x^2 + k_y^2$, so the magnitude of its Fourier transform is $\sqrt{|k_x^2 + k_y^2|}$. This rotationally invariant function in the Fourier domain contrasts sharply with the nonrotationally invariant function shape in (x, y)-space. The difference must arise from the phase spectrum. The factorization (7) is nonunique in that causality associated with the helix mapping can be defined along either x or y axes; thus the operator (7) can be rotated or reflected.

The operator $\mathbf{H}$ has curious similarities and differences with the familiar gradient and divergence operators. In two-dimensional physical space, the gradient maps one field to *two* fields (north slope and east slope). The factorization of $-\nabla^2$ with the helix gives us the operator $\mathbf{H}$ that maps one field to *one* field. Being a one-to-one transformation (unlike gradient and divergence) the operator $\mathbf{H}$ is potentially invertible by deconvolution (recursive filtering).

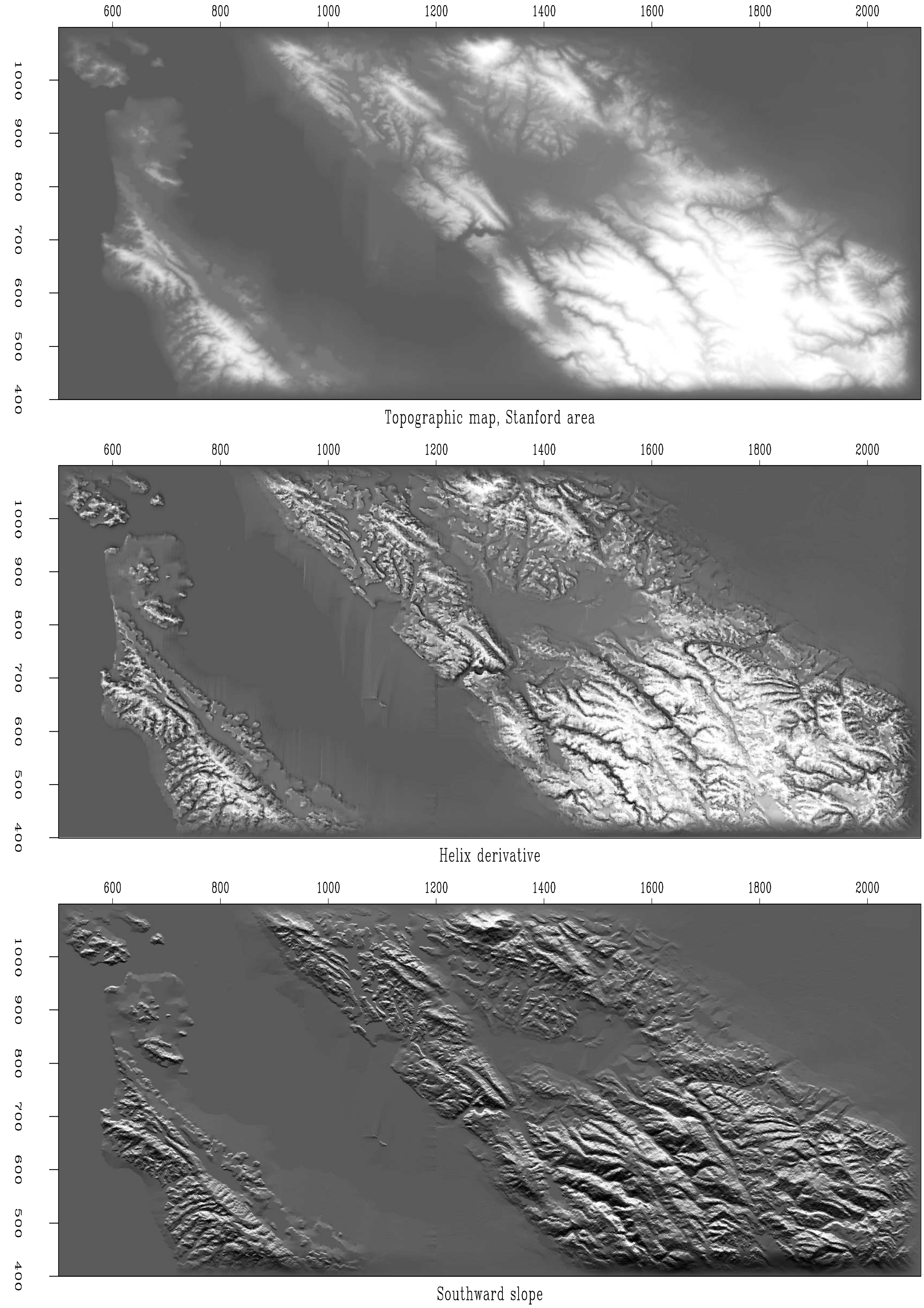

Figure 9: (a) Topography, (b) helical derivative, (c) slope south.

I have chosen the name[3] "helix derivative" or "helical derivative" for the operator $\mathbf{H}$. A telephone pole has a narrow shadow behind it. The helix integral (middle frame of Figure 10) and the helix derivative (left frame) show shadows with an angular bandwidth approaching 180°. Thus, $\mathbf{H}$ is much less directional than either $\frac{\partial}{\partial x}$ or $\frac{\partial}{\partial y}$.

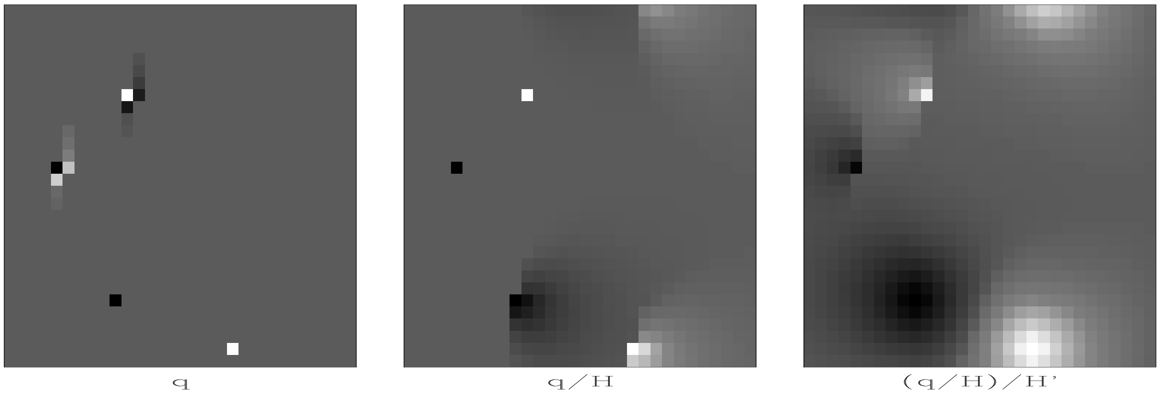

Figure 10: Deconvolution by a filter whose autocorrelation is the two-dimensional Laplacian operator. Amounts to solving the Poisson equation. Left is $\mathbf{q}$; Middle is $\mathbf{q}/\mathbf{H}$; Right is $(\mathbf{q}/\mathbf{H})/\mathbf{H}'$.

This is where the story all comes together. One-dimensional theory, such as the Kolmogoroff spectral factorization, produces not merely a causal wavelet with the required autocorrelation. It produces a wavelet that is stable in deconvolution. Using $\mathbf{H}$ in one-dimensional polynomial division, we can solve many formerly difficult problems very rapidly. Consider the Laplace equation with sources (Poisson's equation). Polynomial division and its reverse (adjoint) gives us $\mathbf{p} = (\mathbf{q}/\mathbf{H})/\mathbf{H}'$ which means that we have solved $\nabla^2 \mathbf{p} = -\mathbf{q}$ in Figure 10 by using polynomial division on a helix. Using the seven coefficients shown, the cost is fourteen multiplications (because we need to run both ways) per mesh point.

[3] Any fact this basic should be named in some earlier field of mathematics or theoretical physics. Admittedly, the concept exists on an infinite Cartesian plane without a helix, but all my codes in a finite space involve the helix, and the helix concept led me to it.

Matrix view of the helix

Finally, let us look at the helix from the view of matrices and numerical analysis. This is not easy because the matrices are so large. Discretize the (x, y)-plane to an $N \times M$ array and pack the array into a vector of $N \times M$ components. Likewise pack the Laplacian operator $\partial_{xx} + \partial_{yy}$ into a matrix. For a 4×3 plane, that matrix is shown in equation (8).

$$
-\nabla^2 \quad = \quad \left[\begin{array}{cccc|cccc|cccc}
4 & 1 & \cdot & \cdot & -1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-1 & 4 & -1 & \cdot & \cdot & -1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & -1 & 4 & -1 & \cdot & \cdot & -1 & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & -1 & 4 & h & \cdot & \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\
\hline
-1 & \cdot & \cdot & h & 4 & -1 & \cdot & \cdot & -1 & \cdot & \cdot & \cdot \\
\cdot & -1 & \cdot & \cdot & -1 & 4 & -1 & \cdot & \cdot & -1 & \cdot & \cdot \\
\cdot & \cdot & -1 & \cdot & \cdot & -1 & 4 & -1 & \cdot & \cdot & -1 & \cdot \\
\cdot & \cdot & \cdot & -1 & \cdot & \cdot & -1 & 4 & h & \cdot & \cdot & -1 \\
\hline
\cdot & \cdot & \cdot & \cdot & -1 & \cdot & \cdot & h & 4 & -1 & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & -1 & \cdot & \cdot & -1 & 4 & -1 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -1 & \cdot & \cdot & -1 & 4 & -1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -1 & \cdot & \cdot & -1 & 4
\end{array}\right] \tag{8}
$$

The two-dimensional matrix of coefficients for the Laplacian operator is shown in (8), where, on a Cartesian space, $h = 0$, and in the helix geometry, $h = -1$. (A similar partitioned matrix arises from packing a cylindrical surface into a 4×3 array.) Notice that the partitioning becomes transparent for the helix, $h = -1$. With the partitioning thus invisible, the matrix simply represents one-dimensional convolution and we have an alternative analytical approach, one-dimensional Fourier Transform. We often need to solve sets of simultaneous equations with a matrix similar to (8). The method we use is triangular factorization.

Although the autocorrelation $\mathbf{r}$ has mostly zero values, the factored autocorrelation $\mathbf{h}$ from (5) has a great number of nonzero terms, but fortunately they seem to be converging rapidly (in the middle) so truncation (of the middle coefficients) seems reasonable. I wish I could show you a larger matrix, but all I can do is to pack the signal $\mathbf{h}$ into shifted columns of a lower triangular matrix $\mathbf{H}$ like this:

$$
\mathbf{H} \;=\; \left[\begin{array}{cccccccccccc}
1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -.6 & -.2 & 0.0 & -.6 & 1.8
\end{array}\right] \tag{9}
$$

If you will allow me some truncation approximations, I now claim that the laplacian represented by the matrix in equation (8) is factored into two parts $-\nabla^2 = \mathbf{H}'\mathbf{H}$ which are upper and lower triangular matrices whose product forms the autocorrelation seen in (8). Recall that triangular matrices allow quick solutions of simultaneous equations by backsubstitution. That is what we do with our deconvolution program.

Iteratively fitting models to data

The basic formulation of a geophysical estimation problem consists of setting up *two* goals, one for data fitting, and the other for model shaping. With two goals, preconditioning is somewhat different. The two goals may be written as:

$$
\begin{aligned}
\mathbf{0} &\approx \mathbf{Fm} - \mathbf{d} \qquad (10)\\
\mathbf{0} &\approx \mathbf{Am} \qquad (11)
\end{aligned}
$$

which defines two residuals, a so-called "data residual" and a "model residual" that are usually minimized by conjugate-gradient, least-squares methods.

To fix ideas, let us examine a toy example. The data and the first three rows of the matrix below are random numbers truncated to integers. The model roughening operator $\mathbf{A}$ is a first differencing operator times 100.

```
d(m)     F(m,n)                                                iter     Norm
---     ----------------------------------------------         ---- -----------
 41.     -55. -90. -24. -13. -73.  61. -27. -19.  23. -55.        1 20.00396538
 33.       8. -86.  72.  87. -41.  -3. -29.  29. -66.  50.        2 12.14780140
-58.      84. -49.  80.  44. -52. -51.   8.  86.  77.  50.        3  8.94393635
  0.     100.   0.   0.   0.   0.   0.   0.   0.   0.   0.        4  6.04517126
  0.    -100. 100.   0.   0.   0.   0.   0.   0.   0.   0.        5  2.64737511
  0.       0.-100. 100.   0.   0.   0.   0.   0.   0.   0.        6  0.79238468
  0.       0.   0.-100. 100.   0.   0.   0.   0.   0.   0.        7  0.46083349
  0.       0.   0.   0.-100. 100.   0.   0.   0.   0.   0.        8  0.08301232
  0.       0.   0.   0.   0.-100. 100.   0.   0.   0.   0.        9  0.00542009
  0.       0.   0.   0.   0.   0.-100. 100.   0.   0.   0.       10  0.00000565
  0.       0.   0.   0.   0.   0.   0.-100. 100.   0.   0.       11  0.00000026
  0.       0.   0.   0.   0.   0.   0.   0.-100. 100.   0.       12  0.00000012
  0.       0.   0.   0.   0.   0.   0.   0.   0.-100. 100.       13  0.00000000
```

Notice at the tenth iteration, the residual suddenly plunges 4 significant digits. Since there are ten unknowns and the matrix is obviously full-rank, conjugate-gradient theory tells us to expect the exact solution at the tenth iteration. This is the first miracle of conjugate gradients.

The second miracle of conjugate gradients is exhibited below. The data and data fitting matrix are the same, but the model damping is simplified.

```
d(m)     F(m,n)                                                iter     Norm
---     ----------------------------------------------         ----  ----------
 41.     -55. -90. -24. -13. -73.  61. -27. -19.  23. -55.        1  3.64410686
 33.       8. -86.  72.  87. -41.  -3. -29.  29. -66.  50.        2  0.31269890
-58.      84. -49.  80.  44. -52. -51.   8.  86.  77.  50.        3 -0.00000021
  0.     100.   0.   0.   0.   0.   0.   0.   0.   0.   0.        4 -0.00000066
  0.       0. 100.   0.   0.   0.   0.   0.   0.   0.   0.        5 -0.00000080
  0.       0.   0. 100.   0.   0.   0.   0.   0.   0.   0.        6 -0.00000065
  0.       0.   0.   0. 100.   0.   0.   0.   0.   0.   0.        7 -0.00000088
  0.       0.   0.   0.   0. 100.   0.   0.   0.   0.   0.        8 -0.00000074
  0.       0.   0.   0.   0.   0. 100.   0.   0.   0.   0.        9 -0.00000035
  0.       0.   0.   0.   0.   0.   0. 100.   0.   0.   0.       10 -0.00000037
  0.       0.   0.   0.   0.   0.   0.   0. 100.   0.   0.       11 -0.00000018
  0.       0.   0.   0.   0.   0.   0.   0.   0. 100.   0.       12  0.00000000
  0.       0.   0.   0.   0.   0.   0.   0.   0.   0. 100.       13  0.00000000
```

Even though the matrix is full-rank, we see the residual drop about 6 decimal places after the third iteration!

Practitioners usually don't like the identity operator for model-shaping but they want the acceleration in convergence. These goals are reconciled by a change in variables also known as preconditioning. The new variables are $\mathbf{p} = \mathbf{A}\mathbf{m}$. The helix gives us a large family of invertable multidimensional roughening operators.

Geoestimation: empty bin example

The basic formulation of a geophysical estimation problem consists of setting up two goals, one for data fitting, and the other for model smoothing. We have data $\mathbf{d}$, a map or model $\mathbf{m}$, a transformation and a roughening filter $\mathbf{A}$, like the gradient ∇ or the helix derivative

H. The two goals may be written as:

$$\begin{aligned} \mathbf{0} &\approx \mathbf{r} = \mathbf{Lm} - \mathbf{d} \qquad (12)\\ \mathbf{0} &\approx \mathbf{p} = \mathbf{Am} \qquad (13)\end{aligned}$$

which defines two residuals, a data residual **r**, and a model residual **p**, which are usually minimized by iterative conjugate-gradient, least-squares methods. A common case is "minimum wiggliness" when **A** is taken to be the gradient operator. The gradient is not invertable so we cannot precondition with its inverse. On the other hand, since $-\nabla^2 = \mathbf{div}\cdot\mathbf{grad} = \mathbf{H}'\mathbf{H}$, taking **A** to be the helix derivative **H**, we can invert **A** and proceed with the preconditioning: We change the free variable in the fitting goals from **m** to **p** (by inverting (13)) with $\mathbf{m} = \mathbf{A}^{-1}\mathbf{p}$ and substituting into both goals getting new goals

$$\begin{aligned} \mathbf{0} &\approx \mathbf{LA}^{-1}\mathbf{p} - \mathbf{d} \qquad (14)\\ \mathbf{0} &\approx \mathbf{p} \qquad (15)\end{aligned}$$

In my experience, iterative solvers find convergence much more quickly when the free variable is the roughened map **p** rather than the map **m** itself.

Figure 11 (left) shows ocean depth measured by a Seabeam apparatus. Locations not surveyed are evident as the homogeneous gray area. Using a process akin to "blind deconvolution" a 2-D prediction error filter **A** is found. Then missing data values are estimated and shown on the right. Preconditioning with the helix speeded this estimation by a factor of about 30. The figure required a few seconds of calculation for about 10^5 unknowns.

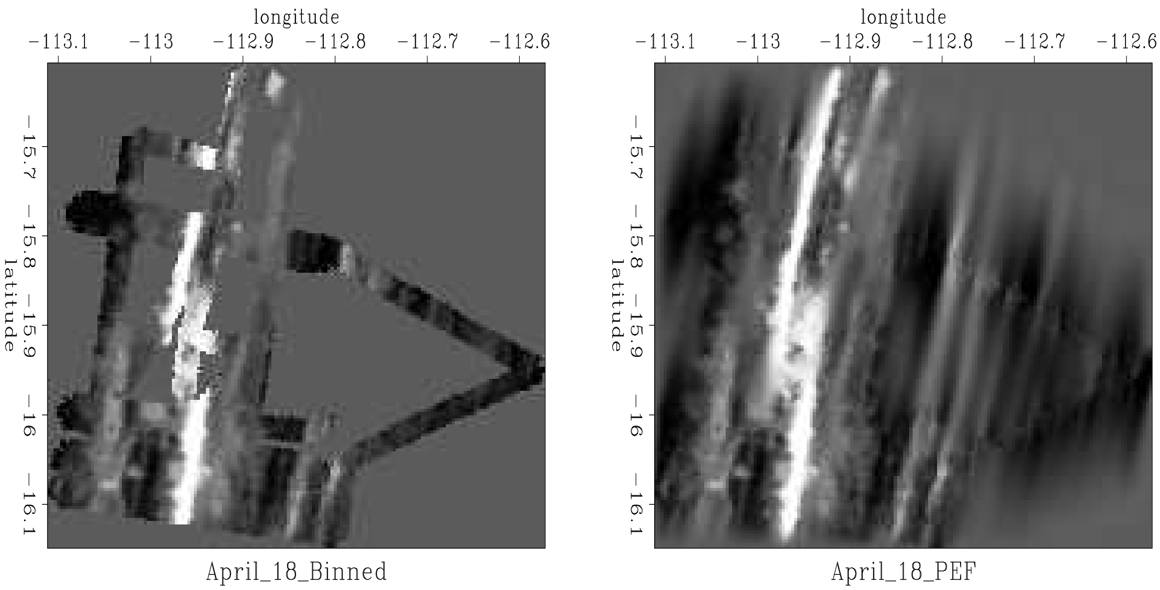

Figure 11: Filling empty bins with a prediction-error filter.

Further applications

I expect applications to rapidly outpace what I write here (that's been happening during the review process) but anyway I indicate a few new directions.

Velocity estimation resembles the empty-bin problem because data quality provides us with velocity information in some locations but not in others. Where we have no velocity information, we often have an idea of the orientation of bedding planes or we have other dip information that tells us how to interpolate or smooth velocity between measurement locations. Figure 12 shows how easy it is to use the helix to create directional smoothers. Clapp

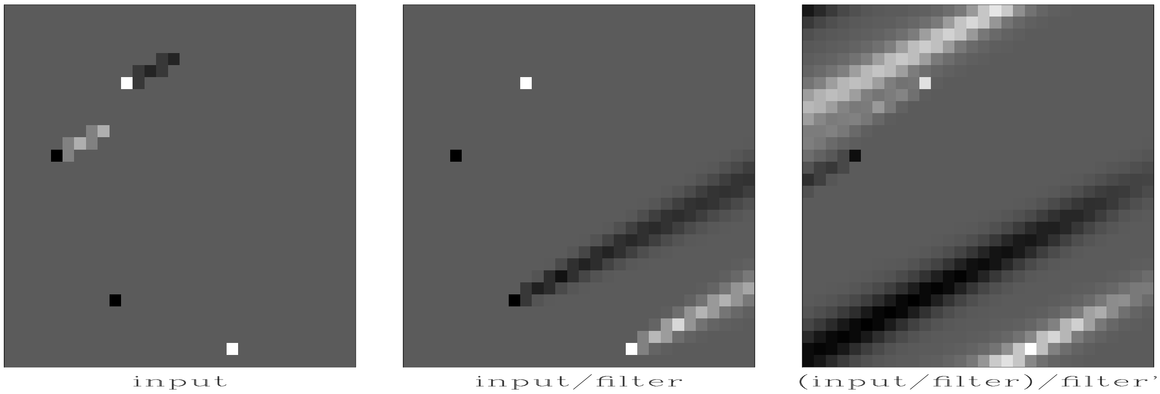

Figure 12: This filter is my guess at a simple low-order 2-D filter whose inverse times its inverse adjoint is approximately a dipping seismic arrival.

et.al (1998) demonstrate constructing velocity functions via the helix by the use of directional smoothers or "flags." The space-variable flag filter is introduced as the regularization for a conventional velocity estimation process.

Poststack 3-D wave-equation migration is another area ripe for the helix. When 3-D became important, the finite-difference method declined because we were unable to find rapid solutions to large scale algebraic equations with coefficient matrices like (8). A first question to be asked, however, is whether the helix does anything new that we could not do with Fourier transforms? Certainly, much of the above depends on Fourier transforms, and some could have been done with Fourier transforms. Recursive filtering is often faster than Fourier transforms, but the real contribution of the helix depends on a weakness of Fourier transformation. The frequency domain is great for differential equations with constant coefficients, stationary statistics, and regularly sampled data, but its utility diminishes when these conditions are not met. This theme can be developed further into 3-D poststack migration process that handles 3-D velocity variation in a wave-equation consistent manner (Rickett et al., 1998).

Spatial filters easily handle aliased data. Figure 13 shows a filter that destroys two waves, one at the border of aliasing, the other well beyond aliasing. The figure shows that the inverse of such a filter creates aliased plane waves. Claerbout (1992) illustrates synthetic data processing beyond aliasing. Recently Crawley (1998) began demonstrating stacking on field data that is aliased in offset at constant midpoint. Again, the helix should accelerate convergence.

REFERENCES

Claerbout, J. Fundamentals of Geophysical Data Processing:. http://sepwww.stanford.edu/sep/prof/, 1976.

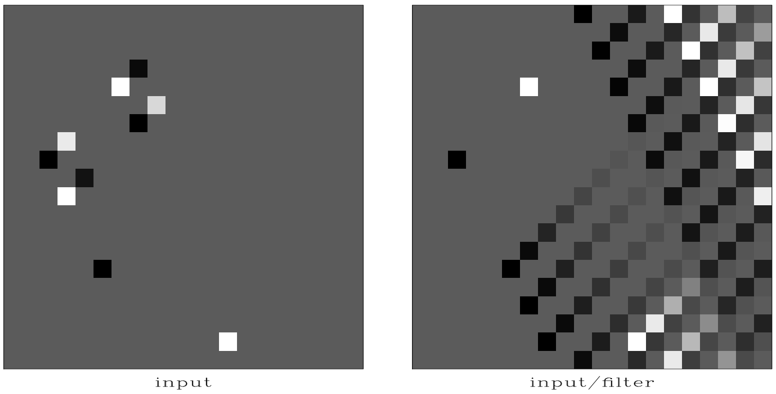

Figure 13: This filter is my guess at a simple low-order 2-D filter whose inverse contains plane waves of two different dips. One of them is spatially aliased.

Claerbout, J. F., 1992, Earth Soundings Analysis: Processing versus Inversion: Blackwell Scientific Publications.

Clapp, R., Biondi, B., Fomel, S., and Claerbout, J., 1998, Regularizing velocity estimation using geologic dip information: 68th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts.

Crawley, S., 1998, Shot interpolation for radon multiple suppression: 68th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts.

Rickett, J., Claerbout, J., and Fomel, S., 1998, Implicit 3-D depth migration by wavefield extrapolation with helical boundary conditions: 68th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts.