Stanford Exploration Project, Report 73, November 18, 1997, pages 435–448

Stanford Exploration Project, Report 73, November 18, 1997, pages 435–448

Information from smiles: mono-plane-annihilator weighted regression

Jon F. Claerbout[1]

keywords: *deconvolution, modeling, amplitudes, statics*

ABSTRACT

An interpreter looking at a migrated section containing two dips in the same place knows that something is wrong. To minimize the presence of multiple dipping events in the same place, we can design a rejection filter to remove the best fitting local plane. This is called a LOMOPLAN (LOcal MOno PLane ANnihilator). The output of a LOMOPLAN contains only other dips so minimizing that output should enable us to improve estimation of model parameters and missing data. Although the LOMOPLAN concept applies to models, not data, experience shows that processing field data with a LOMOPLAN quickly identifies data quality problems. A LOMOPLAN for adequately sampled 3-D data is $(\partial_x + p_x\partial_\tau,\ \partial_y + p_y\partial_\tau)$ which has two outputs, both of which should be minimized.

INTRODUCTION

The main way we estimate parameters in reflection seismology is that we maximize the coherence of theoretically redundant measurements. Thus, we estimate velocity and statics shifts by maximizing something like the power in the stacked data. Here I propose another optimization criterion for estimating model parameters and missing data. An interpreter looking at a migrated section containing two dips in the same place knows that something is wrong. To minimize the presence of multiple dipping events in the same place, we should use the mono plane annihilator (MOPLAN) filter as the weighting operator in any regression. Since the filter is intended for use on images or migrated data, not on data directly, I call it a *plane* annihilator, not a *planewave* annihilator. (A time-migration or merely a stack, however, might qualify as an image.) I must be wary of using the word "wavefront" because waves readily satisfy the superposition principle, whereas images do not, and it is this aspect of images that I advocate and formulate as "prior information."

An example of a MOPLAN in two dimensions, $(\partial_x + p_x\partial_\tau)$, is explored in chapter 4

[1]**email:** not available

of PVI (Claerbout, 1992) where the main goal is to estimate the (τ, x)-variation of p_x. Another family of MOPLANs arise from multidimensional prediction-error filtering as described in PVI chapter 8. These have two additional interesting features: (1) they work on aliased information, and (2) they are a factor of the inverse covariance matrix that can be helpful in any regression.

Here I hypothesize that a MOPLAN may be a valuable weighting function for many estimation problems in seismology. Perhaps we can estimate statics, interval velocity, and missing data using the principle of minimizing the power out of a LOcal MOno PLane ANnihilator (LOMOPLAN) on a migrated section. Thus, those embarrassing semicircles that we have seen for years on our migrated sections may hold one of the keys for unlocking the secrets of statics and lateral velocity variation. I do not claim this concept is as powerful as our traditional methods. I merely claim that we have not yet exploited this concept in a systematic way and that it might prove useful where traditional methods break.

> For an image model of nonoverlapping curved planes a suitable choice of weighting function for regression (model covariance) is the local filter that destroys the best fitting local plane.

Statics

Some years ago, Shuki Ronen and I proposed a formalization of the statics corrections process as "maximize the power in the stack". With MOPLAN filters, a new formalization of the statics problem is possible, "minimize the power in the monoplane-annihilated stack". Unlike the statics concept with Ronen, this new MOPLAN concept is an instance of "geophysical inverse theory" where the model covariance matrix is identified as a MOPLAN. Ronen and I discovered, when we dug inside the statics programs, that our principle implied techniques that were practically the same as the traditional ones. It will be interesting to see what is implied by MOPLAN statics. A data type that might validate MOPLAN statics would be where near-surface corrections and variable dip are important, but curvature is not (so migration is not also involved). Later we will see that an important MOPLAN simultaneously incorporates deconvolution.

DISCLAIMERS

It is not strictly true that a properly migrated data volume should have only one plane at a given location. This is merely a model that for its validity requires many conditions including that (1) a local planar model suffices for curved interfaces, (2) the data is mainly primary reflections, and (3) faults and unconformities are a second order issue.

The idea that the weighting function for regression must be determined along with the regression itself is a nonlinear concept, accompanied by the usual difficulties. The most elementary linearization is to presume the desired events are the strongest ones and that superposing semicircles and other events are weaker. In reality, we typically start from such an oversimplified model and then hope that by iteration we can escape its limitations.

MONO-PLANE DECONVOLUTION

Consider two-dimensional filters. The dip and deconvolution character of such filters is described in considerable detail in chapter 8 of PVI. The coefficients of these two-dimensional convolution filters are chosen to produce minimum power out of a given input. Here we limit attention to the filter coefficient pattern below:

$$\begin{array}{cc} c & d \\ b & e \\ a & f \\ 1 & g \\ \cdot & h \\ \cdot & i \\ \cdot & j \end{array} \tag{1}$$

The vertical axis is time. The horizontal axis is space. The "$\cdot$"s are zeros. The filter can be lengthened in time but not in space. The choice of exactly two columns is a choice to have an analytic form that can exactly destroy a single plane, but cannot destroy two. (The code shown can be used for more columns and more planes.) With two signals that are statistically independent, the filter reduces to the well-known prediction-error filter in the left column and zeros in the right column. If the filter coefficients were extended in both directions on t and to the right on x, it would flatten the two-dimensional spectrum of the input. (An appendix proves these whitening properties). Imagine all the coefficients in (1) vanished but $d = -1$ and the given 1. Such a filter would annihilate an appropriately sloping plane wave.

A subroutine for two-dimensional filtering is `cinlof()` on this page from PVI. That subroutine handles any number of columns so we would limit it to two columns by making the choice `a2=2`. This routine is overly general for the task at hand, but it is a well tested starting place. With `cinlof()` on the current page the filter stops before it runs off the end of the data so there are no edge transients and as a necessary byproduct, the output space is somewhat smaller than the input space. Also, the output aligns with the input under the coefficient that is constrained (in autoregression) to unity.

```
# CINLOF --- Convolution INternal with Lags.  Output is conjugate with FILTER.
#
subroutine cinlof( conj, add, lag1,lag2,  n1,n2,xx,  nb1,nb2,bb,  yy)
integer            conj, add, lag1,lag2,  n1,n2,     nb1,nb2
```

```
real                                        xx(n1,n2), bb(nb1,nb2), yy(n1,n2)
integer y1,y2, x1,x2, b1, b2
call conjnull( conj, add,                               bb,nb1*nb2,  yy,n1*n2)
if( conj == 0 )
        do b2=1,nb2 { do y2= 1+nb2-lag2, n2-lag2+1 {  x2= y2 - b2 + lag2
        do b1=1,nb1 { do y1= 1+nb1-lag1, n1-lag1+1 {  x1= y1 - b1 + lag1
                                yy(y1,y2) = yy(y1,y2) + bb(b1,b2) * xx(x1,x2)
                                }} }}
else
        do b2=1,nb2 { do y2= 1+nb2-lag2, n2-lag2+1 {  x2= y2 - b2 + lag2
        do b1=1,nb1 { do y1= 1+nb1-lag1, n1-lag1+1 {  x1= y1 - b1 + lag1
                                bb(b1,b2) = bb(b1,b2) + yy(y1,y2) * xx(x1,x2)
                                }} }}
return; end
```

Local windows

The earth dip changes rapidly with location. In a small region there is a local dip and dip bandwidth which determining the best LOMOPLAN (LOcal MOPLAN). To see how to cope with the edge effects of filtering in a small region, and to see how to patch together these small regions, see my article (Claerbout, 1992b).

Figure 1 shows a synthetic model that illustrates local variation in bedding. Notice dipping bedding, curved bedding, unconformity between them, and a fault in the curved bedding. Also, notice the image has its amplitude tapered to zero on the left and right sides. After local monoplane annihilation (LOMOPLAN) the continuous bedding is essentially gone. The fault and unconformity remain.

> The local spatial prediction-error filters contain the essence of a factored form of the inverse covariance matrix of the model.

The local spatial prediction-error filters contain the essence of a factored form of the inverse covariance matrix of the model. A covariance matrix has two aspects: a spectral aspect in the prediction-error filter and an amplitude aspect as in gain leveling. Thus the final step in local whitening is to divide the filters by the amplitude of the filter output. This is like applying automatic gain adjustment to the residuals in Figure 1. Doing this gives Figure 2.

Notice the model weakens in amplitude along the sides. (The designer of this model (me) evidently planned to use the model for diffraction and migration.) These damping amplitudes are strong on the LOMOPLAN because a damping plane wave has angular bandwidth greater than zero. Two ways to go beyond the monoplane model are (1) to bend the plane, or (2) to allow amplitude variation along the plane. A one point filter will predict perfectly an exponentially growing or decaying amplitude along a plane wave. But I prefer a bad prediction of the damping plane as an indication that the planar model is breaking down. (Here I took a hint from one-dimensional filter analysis. Recall Burg's old "forward and backward" residual

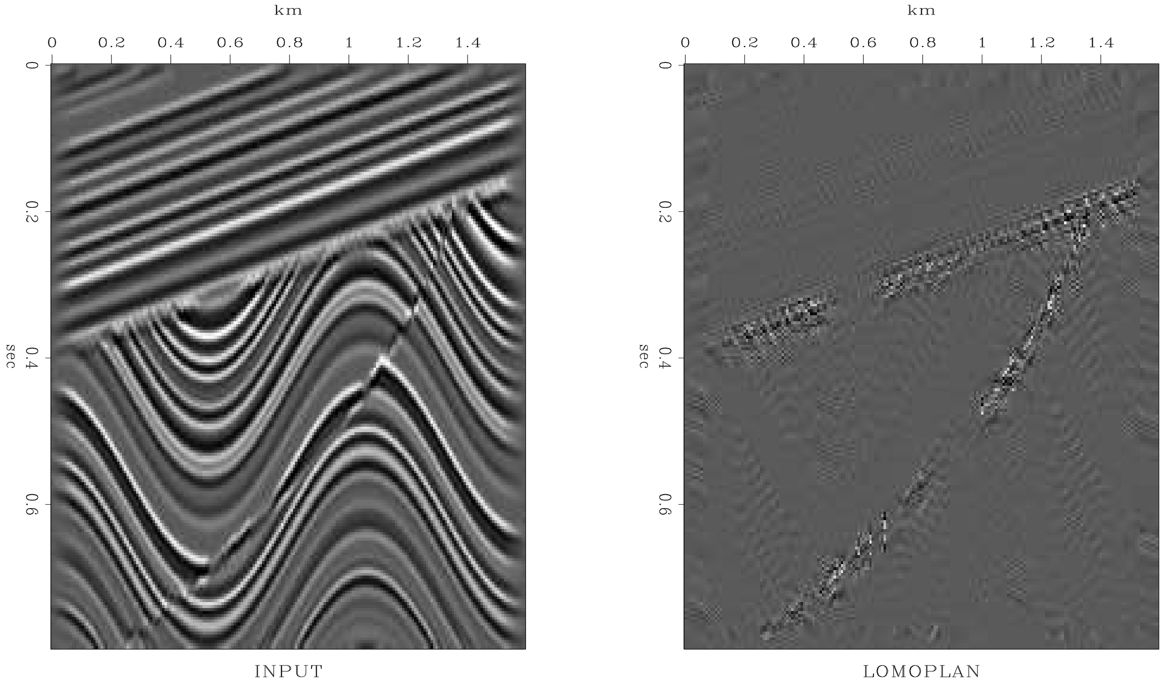

Figure 1: Left is a synthetic reflectivity model. Right is the result of local monoplane annihilation. jon6.moplan-sigmoid0 [ER]

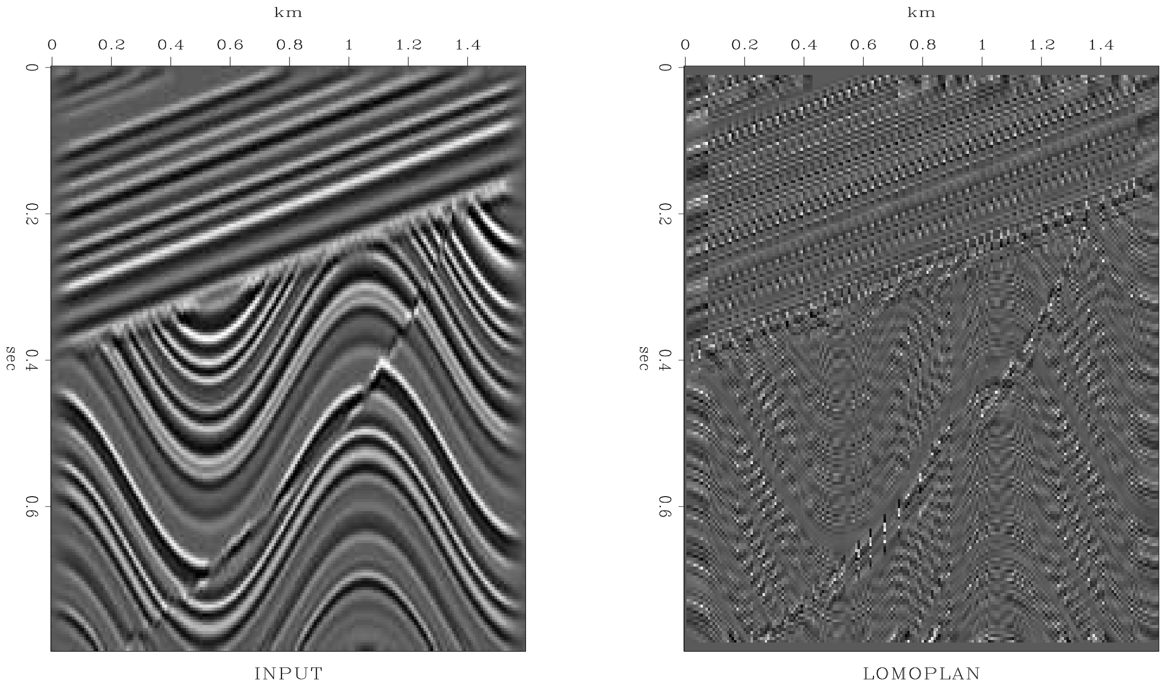

Figure 2: Left is a synthetic reflectivity model. Right is the result of local monoplane annihilation with inverse amplitudes. jon6.moplan-sigmoid1 [ER]

trick (FGDP, p. 134). By forcing a 1-D two-term filter to predict both backward and forward, Burg avoids perfect predictability on growing exponentials.) To achieve imperfect prediction of exponentially growing planes, I require one filter to extinguish two copies of the data, the original copy and a second copy with both τ and x reversed. Note that dips on the reversed copy are the same as on the original but amplitude growth on one is decay on the other. To give these ideas concrete form, I prepared subroutine `burg2()` which applies a convolution to both forward and reversed data.

```
# Burg2 --- Burg 2-D conv with (a1,a2) filter (monoplane annihilator if a2=2)
#       output residual partitioned into normal and backward parts.
#       output conjugate to FILTER.
#       output residual(,) aligns with data(,) at filter coef aa(lag1,lag2)
#
subroutine burg2( conj,add, lag1,lag2, data,n1,n2,  aa,a1,a2,  residual)
integer i1,i2,    conj,add, lag1,lag2,      n1,n2,     a1,a2
real                                  data(n1,n2) ,aa(a1,a2), residual(n1,2*n2)
temporary real                        back(n1,n2)
call conjnull(    conj,add,                          aa,a1*a2,  residual,n1*n2*2)
do i2= 1, n2 {
do i1= 1, n1 {  back( n1-i1+1, n2-i2+1) = data( i1, i2)
                }}
call cinlof( conj, 1, lag1,lag2, n1,n2,data, a1,a2,aa, residual        )
call cinlof( conj, 1, lag1,lag2, n1,n2,back, a1,a2,aa, residual(1,n2+1))
return; end
```

To annihilate the best fitting plane, it is a simple matter of using subroutine `burg2()` in the conjugate-gradient context, as in PVI. This is done in subroutine `moplan()`. Notice the backward residual in `residual( 1, n2+1...2*n2)`.

```
# mono plane annihilator and its residual
#
subroutine moplan( agc, eps, data,n1,n2, niter, lag1,lag2, gap, aa,a1,a2, rr)
integer            agc,           n1,n2, niter, lag1,lag2, gap,    a1,a2
real    dot, lam,       eps, data(n1,n2),            aa(a1,a2),rr(n1*n2*2+a1*a2)
integer i1, a12, r12, iter
temporary real da( a1,a2),  dr( n1*n2*2+a1*a2)
temporary real sa( a1,a2),  sr( n1*n2*2+a1*a2)
a12 = a1 * a2;          r12 = n1 * n2 * 2
call null( rr, r12+a12);        call null( aa, a12);    aa( lag1, lag2 ) = 1.
call null( dr, r12+a12);        call null( da, a12)
call        burg2 ( 0, 0, lag1,lag2, data,n1,n2, aa,a1,a2, rr       )
                          lam =  eps * sqrt( dot( r12, rr, rr))
call        ident ( 0, 0, lam,     a12,          aa,       rr(r12+1))
call        scale (    -1.,    r12+a12,                    rr       )
do iter= 0, niter {
        call burg2 ( 1, 0, lag1,lag2, data,n1,n2, da,a1,a2, rr       )
        call ident ( 1, 1, lam,     a12,          da,       rr(r12+1))

        do i1= 1, min0( a1, lag1+gap-1) {         da(i1,lag2)= 0.}
```

```
        call burg2 ( 0, 0, lag1,lag2, data,n1,n2, da,a1,a2, dr       )
        call ident ( 0, 0, lam,     a12,          da,      dr(r12+1))
        call cgstep( iter,  a12 , aa,da,sa,
                        a12+r12 , rr,dr,sr )
        }
if( agc>0)   call scale( 1./sqrt(dot(r12,rr,rr)), a1*a2,aa)
call         burg2 ( 0, 0, lag1,lag2, data,n1,n2, aa,a1,a2, rr)
return; end
```

As an afterthought, `moplan()` also offers the `agc` option, inverse scaling by the RMS value.

Patch utilities

The patch utilities (see page **??**) allow us to operate subroutines like the new deconvolution subroutine `moplan()` on the facing page in many small patches (or windows) of data. The utilities arrange for a smooth overlap of patches. Subroutine `lomoplan()` on the current page below does the same job as `moplan()` on the facing page above, except that `lomoplan()` on the current page does it in an array of overlapping patches by invoking the patch utility subroutines `cinloiwt()` on page 420, `mkwallwt()` on page 417, and `patch()` on page 416.

Subroutine `lomoplan()` produces a forward residual and also a backward residual that I could not find any reason to display. Also, the display does not explicitly show the patch size. My default values were 20 points on time and 6 channels. The number of patches was designed for a 50% overlap. The time axes are sampled at 4 ms. I also incorporated a gain leveling (`agc`) option though in principle, gain leveling is a separate process whose natural patch size might differ from that of the MOPLANs.

```
# lomoplan  ---- LOcal MOno PLane ANnihilate
#
subroutine lomoplan( agc, eps,  w1,w2,data,n1,n2, resid, gap,  aa,a1,a2,k1,k2)
integer i1,i2,lag1,lag2,agc,w1,w2,        n1,n2,        gap,     a1,a2,k1,k2
real dot,                 eps,       data(n1,n2),resid(n1,n2),aa(a1,a2,k1,k2)
integer niter
temporary real pdata(w1,w2), dpres (w1*w2*2+a1*a2), windwt(w1,w2), wallwt(n1,n2)
call null(                                        resid, n1*n2)
niter= a1 + a1/2;     lag1 = (a1+1)/2;  lag2 = 1
call cinloiwt( a1,a2, lag1,lag2, windwt, w1,w2)
call mkwallwt( k1,k2,           windwt, w1,w2, wallwt, n1,n2)
do i2= 1, k2 {
do i1= 1, k1 {
        call patch( 0, 0, i1,i2, k1,k2, data,     n1,n2, pdata, w1,w2)
        call moplan( agc, eps,         pdata,     w1,w2,
             niter, lag1,lag2, gap, aa(1,1,i1,i2),a1,a2, dpres       )
        call diag( 0, 0, windwt, w1*w2, dpres,           dpres       )
        call patch( 1, 1, i1,i2, k1,k2, resid,    n1,n2, dpres, w1,w2)
        }}
call diag  ( 0, 0, wallwt, n1*n2,      resid, resid )
return; end
```

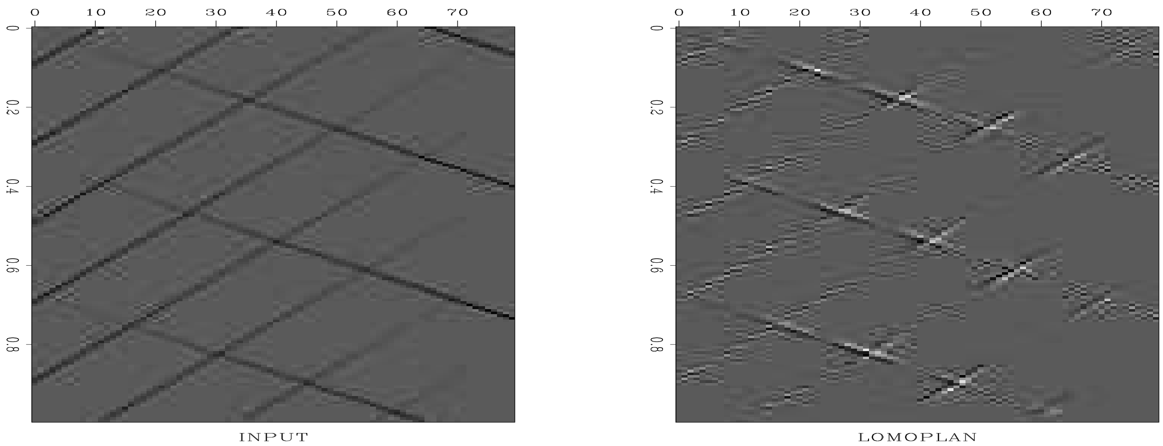

Figure 3: Conflicting dips before and after application of a local monoplane annihilator. Press button for movie. The movie sequence is: 1: data, 2: data after LOMOPLAN, 3: like previous but windows not overlapping, 4: predicted data jon6.moplan-conflict [ER]

Crossing dips

Figure 3 deserves careful study. The input frame is dipping events with amplitudes slowly changing as they cross the frame. The dip of the events is not commensurate with the mesh so linear interpolation is used which accounts for the irregularity along an event. The output panel tends to be small where there is only a single dip present. Where two dips cross, they tend to be equal in magnitude. Studying the output more carefully, we notice that of the two dips, the one that is strongest on the input becomes irregular and noisy on the output whereas the other dip tends to remain phase coherent.

I believe I could rebuild Figure 3 to do a better job of suppressing monodip areas if I passed the image through a lowpass filter, and then designed a gapped deconvolution operator. Instead, I prefered to show you high frequency noise in the place of an attenuated wavefront.

Post filtering

The residual of prediction-error deconvolution tends to have a white spectrum in time. This aspect of deconvolution is somewhat irritating and in practice it requires us to postfilter for display to regain continuity of signals. As is well known (PVI, for example), an alternative to postfiltering is to put a gap in the filter. This should work with 2-D filters too, but it is too early to describe how experimenters will ultimately choose to arrange gaps, if any, in 2-D filters. There are some interesting possibilities. (Inserting a gap also reduces the required number of CG iterations.)

FIELD DATA TESTS

Although the LOMOPLAN concept was developed for geophysical *models*, not raw *data*, initial experience showed that the LOMOPLAN program is very effective for quality testing data.

Some field data examples are in Figures 4 and 5. These results are not surprising. A dominent local plane is removed, leaving noise or the second from strongest local plane. These data sets fit the local plane model so well that subtracting the residual noise from the data made little improvement. These Figures are clearer on a video screen. To facilitate examination of the residual on Figure 4 on paper, I recolored the white residual with a short triangle filter on the time axis.

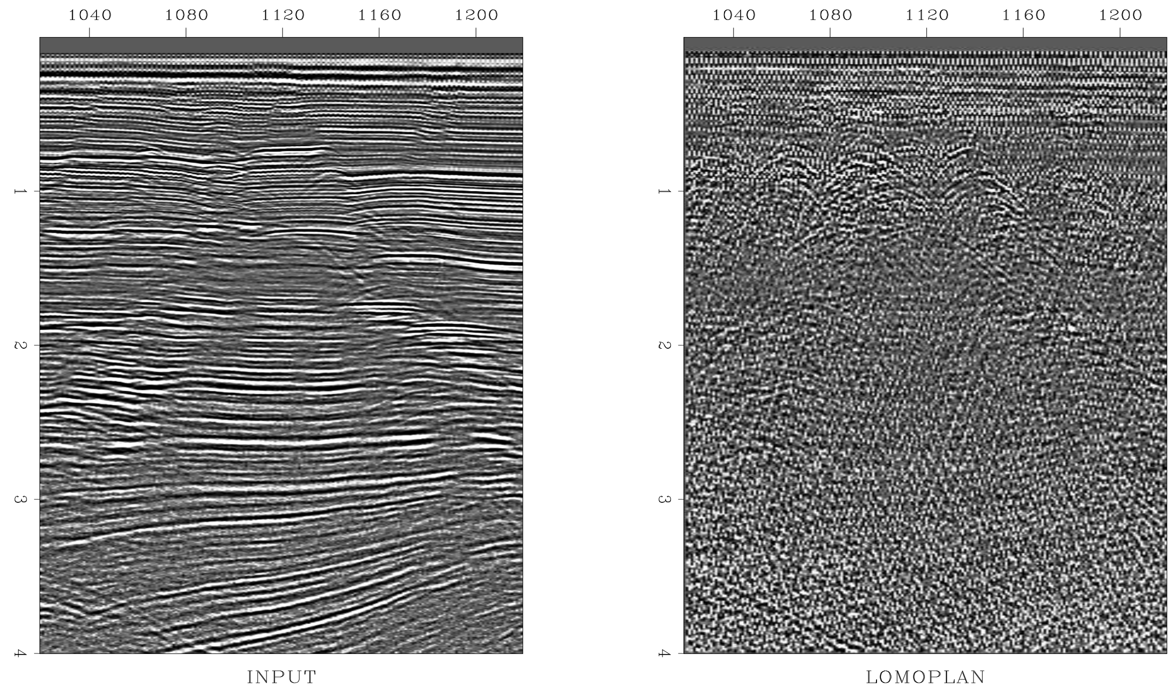

Figure 4: Data section from the Gulf of Mexico (left) and after LOMOPLAN (right) Press button for movie. jon6.moplan-dgulf [ER]

The residual in Figure 5 is large at the dead trace and wherever the data contains crossing events. Also, closer examination showed that the strong residual trace near 1.1 km offset is apparently slightly time shifted, almost certainly a cable problem, perhaps resulting from a combination of the stepout and a few dead pickups. Overall, the local-plane-residual shows a low frequency water velocity wave seeming to originate from the ship.

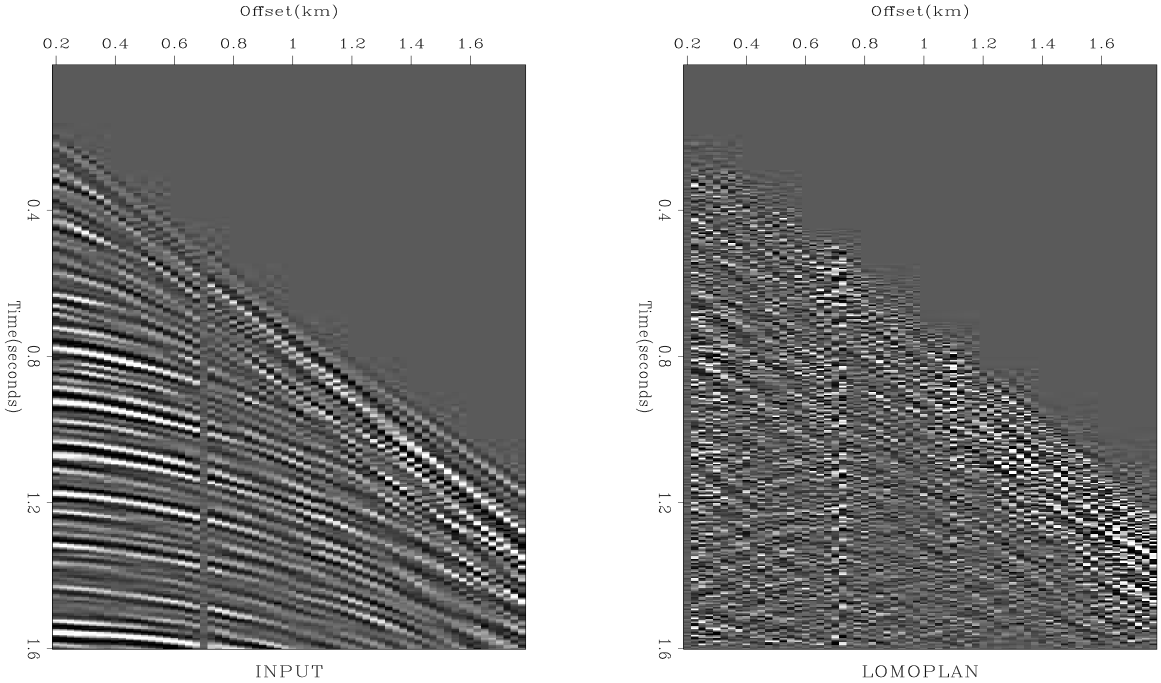

Figure 5: Portion of Yilmaz and Cumro data set 27 (left) and after LOMOPLAN (right). Press button for movie. jon6.moplan-yc27 [ER]

THREE-DIMENSIONAL EXTENSION

We seek the general form of a filter to apply to a data cube $f(x, y, \tau)$ that will annihilate a plane function $f(x, y, \tau) = g(\tau - p_x x - p_y y)$ where $g()$ is an arbitrary one-dimensional function. In practice we usually need to allow the filter to vary with (x, y, τ), but first we investigate the constant dip case. For warmup we consider the case when $(p_x, p_y) = \mathbf{0}$ and later we consider nonzero values of (p_x, p_y). Clearly we can annihilate this plane with either the operator $\partial_x \equiv \partial/\partial x$ or the operator $\partial_y \equiv \partial/\partial y$. The problem with the operator ∂_x is that it annihilates not only the vertical plane function $f(\tau)$ but it annihilates other things such as functions of the form $f(\tau, y)$, an example of which is the family of planes, $f(\tau - p_y y)$. A way to annihilate one plane and only one plane is to choose a filter with *two* outputs, namely the filter (∂_x, ∂_y). For this two-component output to be annihilated in a regression, both components must be annihilated. This is the lowest order filter that will do the required job. We can make a higher order filter that annihilates the plane by multiplying the column vector filter $(\partial_x, \partial_y)'$ by its conjugate getting the negative of the Laplacian $\partial_{xx} + \partial_{yy}$. An apparent advantage of the Laplacian filter is that its output is a single volume. In time we will learn the practical distinction between these two monoplane annihilators. For now, I am betting on the two-output lower order filter.

False start

Curiously I find that people do not like the idea of a filter needing to have two outputs. A false start (for me too) is to guess an operator like

$$G_1 \quad = \quad p_y \frac{\partial}{\partial x} + p_x \frac{\partial}{\partial y} + 2 p_x p_y \frac{\partial}{\partial t} \tag{2}$$

to kill the plane $f(\tau - p_x x - p_y y)$. While it is true that G_1 kills this plane, it also kills the two planes $f_1(\tau - 2p_y y)$ and $f_2(p_x x - p_y y)$ so it does not do the job we need, a low order filter that rejects one and only one plane.

Thus, the way to annihilate any signal on any dipping plane $f(\tau - p_x x - p_y y)$ without annihilating signals on any other planes is to annihilate the two volume outputs of the vector filter $(\partial_x + p_x \partial_\tau,\ \partial_y + p_y \partial_\tau)$. Where these two volumes are already zero the migrated data fits our notions of what a migrated data set should be, so there is nothing more to be done—no parameters can be fitted to embellish the model. Where either of the two volumes is nonzero, there more than one planar event occurs on the input (or the (p_x, p_y) field was incorrectly estimated).

Confusing direction

It is confusing to contrast my plane destructor in 3-D made from two planar filters (each with volume outputs) with the Burg style 3-D PEF from PVI page 198 shown in Figure 6.

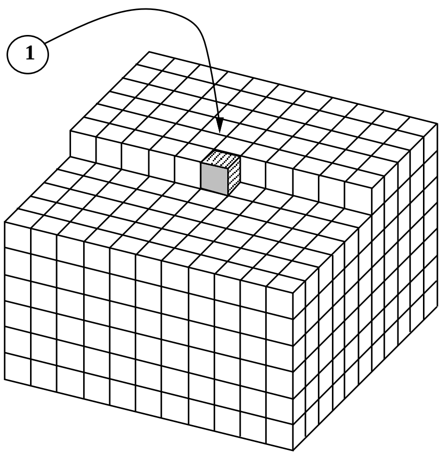

Figure 6: Three-dimensional prediction-error filter. jon6.moplan-3dpef [NR]

APPENDIX

This appendix is a *slightly* clarified version of a very important proof found in PVI.

Proof that PE filter output is white

[2] The basic idea of least-squares fitting is that the residual is orthogonal to the fitting functions. Applied to the PE filter, this idea means that the output of a PE filter is orthogonal to lagged inputs. The orthogonality applies only for lags in the past because prediction knows only the past while it aims to the future. What we want to show is different, namely, that the output is uncorrelated with *itself* (as opposed to the input) for lags in *both* directions; hence the output spectrum is white.

We are given a signal y_t and filter it by

$$x_t \quad = \quad y_t - \sum_{\tau>0} a_\tau y_{t-\tau} \tag{3}$$

We found a_τ by setting to zero $d(\sum x_t^2)/da_\tau$:

$$\sum_t x_t y_{t-\tau} \quad = \quad 0 \qquad \text{for } \tau > 0 \tag{4}$$

We interpret this to mean that the residual is orthogonal to the fitting function, or the present PE filter output is orthogonal to its past inputs, or one side of the crosscorrelation vanishes. Taking an unlimited number of time lags and filter coefficients, the crosscorrelation vanishes not only for $\tau > 0$ but for larger values, say $\tau + s$ where $\tau \geq 0$ and $s > 0$. In other words, the future PE filter outputs are orthogonal to present and past inputs:

$$\sum_t x_{t+s} y_{t-\tau} \quad = \quad 0 \qquad \text{for } \tau \geq 0 \text{ and } s > 0 \tag{5}$$

Recall that if $\mathbf{r} \cdot \mathbf{u} = 0$ and $\mathbf{r} \cdot \mathbf{v} = 0$, then $\mathbf{r} \cdot (a_1\mathbf{u} \pm a_2\mathbf{v}) = 0$ for any a_1 and a_2. So for any a_τ we have

$$\sum_t x_{t+s}(y_t \pm a_\tau y_{t-\tau}) \quad = \quad 0 \qquad \text{for } \tau \geq 0 \text{ and } s > 0 \tag{6}$$

and for any linear combination we have

$$\sum_t x_{t+s}(y_t - \sum_{\tau>0} a_\tau y_{t-\tau}) \quad = \quad 0 \qquad \text{for } \tau \geq 0 \text{ and } s > 0 \tag{7}$$

Therefore, substituting from ((3)), we get

$$\sum_t x_{t+s} x_t \quad = \quad 0 \qquad \text{for } s > 0 \tag{8}$$

which is an autocorrelation function and must be symmetric. Thus,

$$\sum_t x_{t+s} x_t \quad = \quad 0 \qquad \text{for } s \neq 0 \tag{9}$$

Since the autocorrelation of the prediction-error output is an impulse, its spectrum is white.

[2] I would like to thank John P. Burg for this proof.

REFERENCES

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