Stanford Exploration Project, Report 77, November 17, 1997, pages 21–28

Stanford Exploration Project, Report 77, November 17, 1997, pages 21–28

3-D local monoplane annihilator

Jon F. Claerbout[1]

keywords: *three-dimensional, modeling, deconvolution, two-dimensional*

ABSTRACT

The LOMOPLAN (LOcal MOnoPLane ANnihilator) filter in three dimensions is a deconvolution filter that takes a volume in and produces two volumes out. The x-output volume results from a first order prediction-error filter on the x-axis, and the y-output volume is likewise on the y-axis. Synthetic data studies are promising.

INTRODUCTION

I first expounded the local monoplane concept (Claerbout, 1992b) in two-dimensions. Here we pursue the three-dimensional aspects of this concept.

LOMOPLAN (LOcal MOno PLane ANnihilator) is a data-adaptive filter that extinguishes a local monoplane, but cannot extinguish a superposition of several planes. We presume an ideal sedimentary model as made of (possibly curved) parallel layers. Because of the superposition principle, data can be a superposition of several plane waves, but the ideal model should consist locally of only a single plane. Thus, LOMOPLAN extinguishes an ideal model, but not typical data. I conceived of LOMOPLAN as the "ultimate" optimization criterion for inversion problems in reflection seismology (1992) but it has not yet demonstrated that it can attain that lofty goal. Instead, however, in two dimensions I found it useful in data interpretation and in data quality inspection. In this study, I sought to demonstrate usefulness with 3-D data such as 3-D stacks or migrated volumes, or 2-D prestack data.

Simplified notion of 3-D LOMOPLAN

In experimenting with 3-D LOMOPLAN, I came upon a conceptual oversimplification, which although it is not precisely correct, gives a suitable feeling of the meaning of the operator. Imagine the earth was flat horizontal layers, except for occasional faults. Then, to find the faults you might invoke the horizontal gradient of the 3-D

[1]**email:** not available

continuum of data. The horizontal components of gradient vanish except at a fault, where their relative magnitudes tell you the orientation of the fault. Instead of using the gradient vector, you could use prediction-error filters of first order (two components) along x and y directions. 3-D LOMOPLAN is like this, but the flat horizontal bedding may be dipping or curved. No output is produced (ideally) except at faults. The 3-D LOMOPLAN is like the gradient *along the plane of the bedding.* It is nonzero where the bedding has an intrinsic change.

LOMOPLAN flags the bedding where there is an intrinsic change.

LOMOPLAN DEFINITION IN 2-D

First we review two-dimensional deconvolution filters. The dip and deconvolution character of such filters is described in considerable detail in chapter 8 of my recent book (Claerbout, 1992). The coefficients of deconvolution filters are chosen to produce minimum power out of a given input. The coefficients in a 2-D LOMOPLAN filter fit the pattern:

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

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. 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 (1992). Imagine all the coefficients in (1) vanished but $d = -1$ and the given 1. Such a filter would annihilate an appropriately sloping plane wave.

The coefficients (a, b, c) in the filter (1) have an interesting but not fully understood role. To begin with, their presence is suggested by the representation of 2-D spectra by the inverse of their autoregression filters (1992). The presence of these coefficients will cause the output to be white on the time axis. However, if we were to omit these coefficients, or constrain them to zero, the (d, e, f, g, h, i, j) coefficients would probably attempt to adjust themselves toward making the output temporally white (because the earth tends to have flat dip). Thus, it seems that our goals for studying the spatial aspects of the data become bound together with the temporal aspects. Since data is always oversampled in time, and we are not interested in whitening it in time, I needed to design something like a gapped filter, or do postfiltering to give the output a more normal color. Typically, I included the (a, b, c) coefficients and then applied a triangular smoothing filter on the time axis after LOMOPLAN.

LOMOPLAN FORM IN 3-D

Three-dimensional LOMOPLAN is somewhat like multiple passes of two-dimensional LOMOPLAN, i.e. we first LOMOPLAN the (t, x)-plane for each y, and then we LOMOPLAN the (t, y)-plane for each x. Actually, 3-D LOMOPLAN is a little more complicated than this. Each LOMOPLAN filter is designed on all the data in a small (t, x, y) volume.

I began from my earlier two-dimensional code shown in (Claerbout, 1992b) and made the obvious extensions to three dimensions. For example, the obvious extension to the 2-D code for convolution with a 3-D filter not being allowed to flow outside the faces of the input volume is in subroutine `cinlof3()` on this page.

```
# CINLOF --- Convolution INternal with Lags.  Output is conjugate with FILTER.
#
subroutine cinlof3( conj,add, lg1,lg2,lg3,  xx,n1,n2,n3,  bb,nb1,nb2,nb3,  yy)
integer             conj,add, lg1,lg2,lg3,     n1,n2,n3,     nb1,nb2,nb3
real                                        xx(n1,n2,n3), bb(nb1,nb2,nb3)
real                                        yy(n1,n2,n3)
integer y1,y2,y3,  x1,x2,x3,   b1, b2, b3
call conjnull( conj, add,  bb,nb1*nb2*nb3,  yy,n1*n2*n3)

do b3=1,nb3 {  do y3= 1+nb3-lg3, n3-lg3+1 {  x3= y3 - b3 + lg3
do b2=1,nb2 {  do y2= 1+nb2-lg2, n2-lg2+1 {  x2= y2 - b2 + lg2
do b1=1,nb1 {  do y1= 1+nb1-lg1, n1-lg1+1 {  x1= y1 - b1 + lg1
        if( conj == 0 )
                yy( y1,y2,y3) = yy( y1,y2,y3) + bb( b1,b2,b3) * xx( x1,x2,x3)
        else
                bb( b1,b2,b3) = bb( b1,b2,b3) + yy( y1,y2,y3) * xx( x1,x2,x3)
        }}  }}  }}
return; end
```

Likewise, instead of cutting a window of data into patches, we cut a volume of data into subcubes with `patch3()` on the current page.

```
# patch3 ---- copy the j[123]-th of k[123] subcubes from a volume.
#
subroutine patch3( conj,add, j1,j2,j3, k1,k2,k3, wall, n1,n2,n3, wind, w1,w2,w3)
integer            conj,add, j1,j2,j3, k1,k2,k3,       n1,n2,n3,       w1,w2,w3
integer i1,i2,i3, s1,s2,s3, d1,d2,d3
real                                             wall( n1,n2,n3),wind( w1,w2,n3)
call conjnull(     conj,add,                     wall, n1*n2*n3, wind, w1*w2*w3)

if( k3 != 1) {  s3 = 1.5 + (n3 - w3) * (j3-1.)/(k3-1.)}    else { s3= 1}
if( k2 != 1) {  s2 = 1.5 + (n2 - w2) * (j2-1.)/(k2-1.)}    else { s2= 1}
```

```
if( k1 != 1) {  s1 = 1.5 + (n1 - w1) * (j1-1.)/(k1-1.)}    else { s1= 1}

do i3= 1, w3 {  d3= i3 + s3 - 1
do i2= 1, w2 {  d2= i2 + s2 - 1
do i1= 1, w1 {  d1= i1 + s1 - 1
        if( conj == 0 )
                wind( i1,i2,i3) = wind( i1,i2,i3) + wall( d1,d2,d3)
        else
                wall( d1,d2,d3) = wall( d1,d2,d3) + wind( i1,i2,i3)
        }}}
return; end
```

Likewise weighting function code and conjugate-gradient code extend in the obvious way. When you finish the obvious extensions, you are left with a little interesting decision making on the shape of the three-dimensional least-squares filters that you want to design. (If you don't like LOMOPLAN, you have prepared yourself an alternative to do any other locally variable three-dimensional least-squares filtering. That is how I came to do steep-dip deconvolution and the time-domain (f, x) filtering described elsewhere in this report.)

For LOMOPLAN, the final step is to specialize to a two-dimensional filter ((t, x) or/and (t, y)-space) in a three-dimensional volume. Luckily, the codes look quite simple and elegant (thanks to the `ratfor` and `sat` preprocessors to fortran) and the debugging went quickly.

SYNTHETIC EXAMPLES

The qdome synthetic

Earlier 2-D migration and LOMOPLAN studies such as (1992) used a model called "Sigmoid". I used the sigmoid model for several applications and others also found it useful (Zhang and Claerbout, 1992). Using the same modeling concepts, I set out to make a three-dimensional model. The model has horizontal layers near the top, a Gaussian appearance in the middle, and dipping layers on the bottom with horizontal unconformities between the three regions. Figure 1 shows a vertical slice through the 3-D "qdome" model and components of its LOMOPLAN. There is also a fault that will be described later. The most interesting part of the qdome model is the Gaussian center. I started from the equation of a Gaussian

$$z(x, y, t) \quad = \quad e^{-(x^2+y^2)/t^2} \tag{2}$$

and backsolved for t

$$t(x, y, z) \quad = \quad \sqrt{\frac{x^2 + y^2}{-\ln z}} \tag{3}$$

Then I used a random number generator to make a blocky one-dimensional impedance function of t. At each (x, y, z) location in the model I used the impedance at

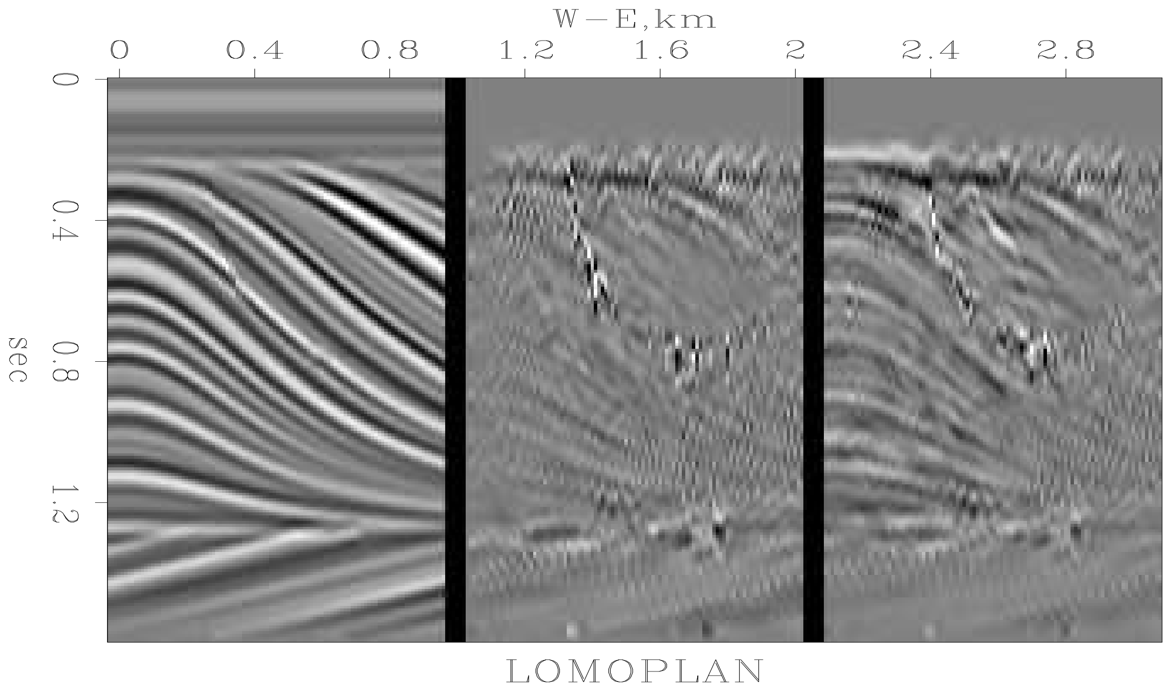

Figure 1: Left is a vertical slice through the 3-D "qdome" model. Center is the in-line component of the LOMOPLAN. Right is the cross-line component of the LOMOPLAN. jon1-qdomesico [ER]

time $t(x, y, z)$, and finally defined reflectivity as the logarithmic derivative of the impedance. Without careful interpolation (particularly where the beds pinch out) a variety of curious artifacts appear. I hope to find time to make a tutorial lesson on interpolation from the experience of making the qdome model. As a refinement to the model, within a certain subvolume the time $t(x, y, z)$ is given a small additive constant. This gives a fault along the edge of the subvolume. Ray Abma defined the subvolume for me in the qdome model. The fault looks quite realistic, and it is easy to make faults of any shape, though I wonder how they would relate to realistic fault dynamics. Figure 2 shows a top view of the 3-D qdome model and components of its LOMOPLAN. Notice the cross-line spacing has been chosen to be double the in-line spacing. Evidently a consequence of this in both Figure 1 and Figure 2 is that the Gaussian dome is not so well suppressed on the crossline cut as on the in-line cut. By comparison, notice that the horizontal bedding above the dome is perfectly suppressed, whereas the dipping bedding below the dome is imperfectly suppressed.

Finally, I became irritated at the need to look at *two* output volumes. Since I rarely if ever interpreted the polarity of the LOMOPLAN components, I formed their sum of squares and show the square root in Figure 3.

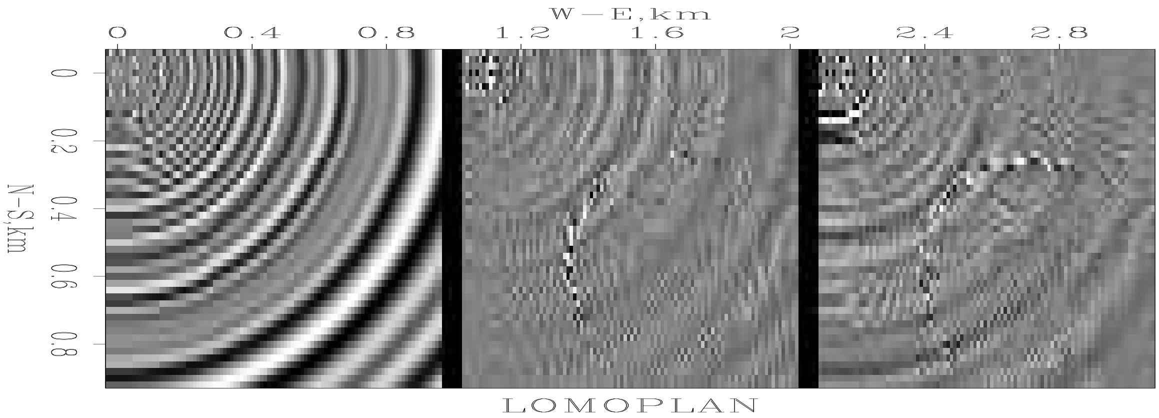

Figure 2: Left is a horizontal slice through the 3-D qdome model. Center is the in-line component of the LOMOPLAN. Right is the cross-line component of the LOMOPLAN. Press button for volume view. jon1-qdometoco [ER]

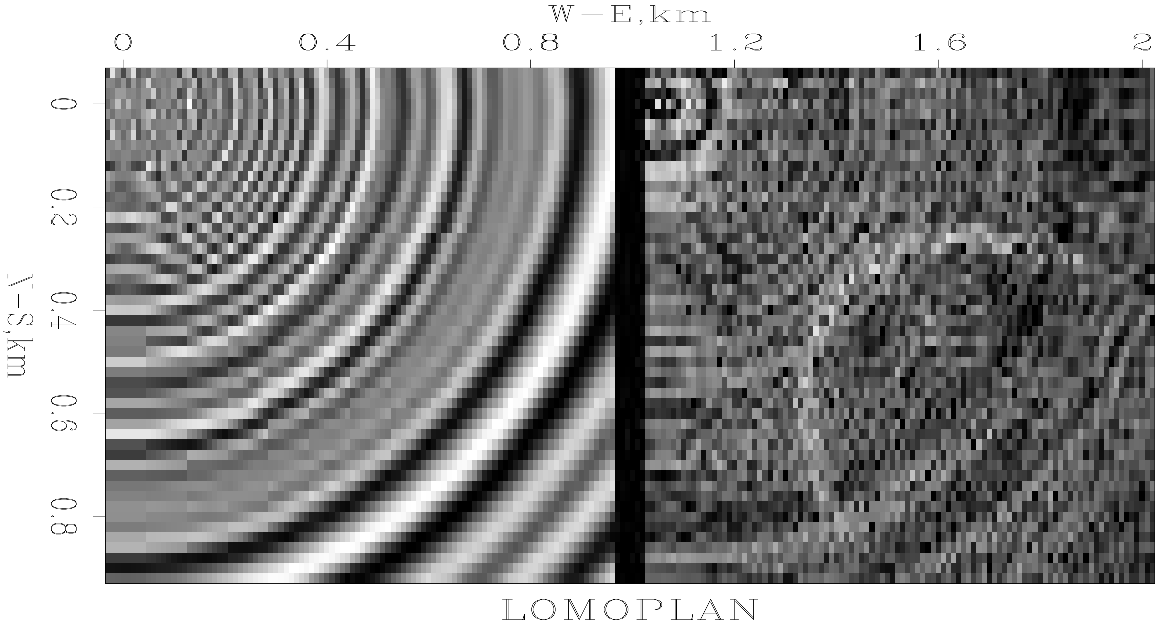

Figure 3: Left is the model. Right is the magnitude of the LOMOPLAN components in Figure 2. Press button for volume view. jon1-qdometora [ER]

FIELD DATA EXPERIMENTS

I experimented with some 3-D data to see if the LOMOPLAN view would prove interesting. Preliminary views were not, so I omit them here. First I looked at the Gulf of Thailand GSI migrations. Unfortunately all I could find was the pixel byte form, so I converted that back to floats. Second, I examined Mihai's unmigrated stack from Halliburton's High Island 3D spec survey. I experimented first with the shallow faults, but these were confounded by multiples and the little dip present made the LOMOPLAN time slices in no way more interesting than those of the input data. A problem I had with the early times on this survey is that the crosslines had noticeable (but inexplicable) timeshifts from one line to another. Thus the crosslines, and hence the 3-D nature of the LOMOPLAN was not exemplified. The crossline problem was absent at later times. Since the computations were fairly demanding however, before results could be obtained I turned to learning the parallel computer, and got caught up with the good results I obtained elsewhere in this report with steep-dip deconvolution (Claerbout, 1993).

REFERENCES

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

Claerbout, J. F., 1992b, Information from smiles: Mono-plane-annihilator weighted regression: SEP–**73**, 409–420.

Claerbout, J. F., 1992c, Nonstationarity and conjugacy: Utilities for data patch work: SEP–**73**, 391–400.

Claerbout, J. F., 1993, Steep-dip deconvolution: SEP–**77**, 245–256.

Zhang, L., and Claerbout, J. F., 1992, Wave-equation resampling of unevenly spaced traces: SEP–**75**, 243–250.