Stanford Exploration Project, Report 113, July 8, 2003, pages 247–261

Flattening 3-D data cubes in complex geology

Jesse Lomask[1]

ABSTRACT

3-D volumes of data can be efficiently flattened with Fourier domain methods as long as the reflections are continuous and depth invariant. To handle faults (discontinuous reflections), I pose the flattening problem in the time-space (T-X) domain to apply a weight. This ignores fitting equations that are affected by the faults. This approach successfully flattens a synthetic faulted model. Also, the flattening method is applied repeatedly in the T-X domain to flatten a synthetic model that has pinch-outs and structure that varies with depth. There are two possible schemes for handling unconformities. One scheme requires that the unconformity be picked, the data separated into different volumes, flattened individually, and then recombined. Another scheme is to apply the flattening method which picks travel-times for all horizons at once without being restricted to time-slices. It is expected that this method will be much more computationally intensive but will require less initial picking. Both of these methods need more development and testing.

INTRODUCTION

The flattening method described in Lomask and Claerbout (2002) to automatically flatten entire 3-D seismic cubes with minimal picking, has major difficulties with faults, unconformities, and pinch-outs. The original method resolves local dips into time shifts via a least-squares problem that is solved quickly in the Fourier domain. This is very similar to an approach used by Bienati et al. (1999a,b); Bienati and Spagnolini (2001, 1998), yet here the full data volume is flattened at once. This approach works efficiently with unfaulted and depth invariant reflections. However, if the data contains discontinuities caused by faulting then it has trouble. The dip at the faults are estimated incorrectly and these incorrect dip values are integrated along with the rest of the correct dip values to cause significant errors. Because this method integrates dips along time-slices, it has difficulties flattening cubes where the structure varies with depth. Flattening pinch-outs and unconformities is also problematic because the flattening solution can be non-unique as two horizons maybe compressed into one by flattening.

In this paper, I review the basic flattening process. I also present a time-distance (T-X) least-squares approach for flattening faulted seismic data cubes. Using the T-X domain rather than the Fourier domain permits the application of a weight to throw out fitting equations affected by bad dip estimates at faults. I also discuss solutions for handling pinch-outs. This flattening method can be applied repeatedly to handle the common case of depth-varying

[1]**email:** lomask@sep.stanford.edu

structure. Lastly, I review methods to flatten unconformities. One method proposed by Sergey Fomel does not restrict the integration of dips to time-slices. Although this may be considerably more computationally expensive, this method might be able to flatten data with unconformities with less picking.

REVIEW OF BASIC FLATTENING METHODOLOGY

The basic idea is similar to phase unwrapping (Claerbout, 1999), but instead of summing phase differences to get total phase, dips are summed to get total time shifts that are then used to flatten the data. To apply the shifts, the central trace is held constant as a reference and all other traces are shifted vertically to match it.

The first step is to calculate dips everywhere in the 3-D seismic cube. Thus far only a simple dip estimation method has been used. Dip can be easily calculated using a plane-wave destructor as described in Claerbout (1992). For each point in the data cube two components of dip, p_x and p_y, are estimated in the x direction and y direction, respectively. These can be represented everywhere on the a mesh as vectors as $\mathbf{p}_x$ and $\mathbf{p}_y$. For the dip in the x direction of a seismic cube with a wave field represented by $u(x,y,t)$, I calculate the following locally:

$$\mathbf{p}_x = -\frac{\mathbf{u}_x \cdot \mathbf{u}_t}{\mathbf{u}_t \cdot \mathbf{u}_t}, \tag{1}$$

where $\mathbf{u}_x$ is a vector with components $\partial u/\partial x$ taken on a mesh in (x,t) and $\mathbf{u}_t$ is a vector with components $\partial u/\partial t$. Because I am calculating local dips, it is necessary to smooth the dips. I apply a triangle filter to both the numerator and denominator of equation (1). Since the dip estimation technique described in equation (1) is very dependent on the amplitude from trace-to-trace, amplitude variations along the horizons will create inaccuracies in dip estimation. This will, in turn, affect the quality of the flattening result. Therefore, an improved dip estimator will be very beneficial. For example, plane wave destructor filters (Fomel, 2001) will provide more robust dip information as they are less sensitive to aliasing and may require less smoothing than the dip estimation technique in equation (1).

The most basic flattening approach is to integrate dips on each time-slice in the data cube to get total time shifts ($\mathbf{t} = t(x,y,z)$). The gradient ($\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$) of the time shifts can be related to the estimated dip ($\mathbf{p} = (\mathbf{p}_x, \mathbf{p}_y)$) in an overdetermined system with the following regression:

$$\nabla \mathbf{t} \quad = \quad \begin{bmatrix} \frac{\partial \mathbf{t}}{\partial \mathbf{x}} \\ \\ \frac{\partial \mathbf{t}}{\partial \mathbf{y}} \end{bmatrix} \approx \begin{bmatrix} \mathbf{p}_x \\ \\ \mathbf{p}_y \end{bmatrix}. \tag{2}$$

The dips are summed to find a total time shift vector using:

$$\nabla^2 \mathbf{t} \quad \approx \quad \nabla' \mathbf{p} \tag{3}$$

where ∇^2 is the Laplacian and ∇' is the divergence. Solving this equation for time-slices in both the Fourier domain and time-space domain is the basic flattening method described in the following sections.

Integrating dips in the Fourier domain

Using the integration method described below (Lomask and Claerbout, 2002), I first apply the divergence (∇') to the dip ($\mathbf{p}$). I then convert to Fourier space where I integrate twice by dividing by the Laplacian. Then I convert back to the time domain. The resulting $\mathbf{t}$ can be thought of as the time shifts to apply to each point in the data to flatten it.

The flattening corrections for a 3-D volume can be generated in two different ways. In 2-D integration, the dips across each time-slice are integrated separately. In 3-D integration, the dips for all slices are integrated at once.

Integrating time slices independently: Beginning with input dip information across each horizon I have:

$$\mathbf{p} \quad = \quad \left(\mathbf{p}_x, \mathbf{p}_y\right). \tag{4}$$

The analytical solution to equation (3) is found with:

$$\mathbf{t} \quad \approx \quad \mathrm{FFT_{2D}}^{-1}\left[\frac{\mathrm{FFT_{2D}}\left[\nabla'\mathbf{p}\right]}{-Z_x^{-1}-Z_y^{-1}+4-Z_x-Z_y}\right] \tag{5}$$

where $Z_x = e^{iw\Delta x}$ and $Z_y = e^{iw\Delta y}$.

The chief stumbling block for this approach is that the zero frequency component is neglected from the denominator of equation (5). This means that each time slice has a constant shift applied to it. It works out that this shift is equal to the average absolute time value. The danger here is if there is any noisy dip values in one slice not present in adjacent slices, the time correction to flatten a volume of data may actually swap data values. One way to prevent this is to regularize in 3-D by integrating all time-slices at once. Another way would be to adequately smooth the dip values.

Integrating all time slices at once: Beginning with my input dip data:

$$\mathbf{p} \quad = \quad \left(\mathbf{p}_x, \mathbf{p}_y, \mathbf{p}_t\right) \tag{6}$$

where $\mathbf{p}_t$ is all ones for smoothness in time (explained below).

The analytical solution is found with:

$$\mathbf{t} \quad \approx \quad \mathrm{FFT_{3D}}^{-1}\left[\frac{\mathrm{FFT_{3D}}\left[\nabla'\mathbf{p}\right]}{-Z_x^{-1}-Z_y^{-1}-Z_t^{-1}+6-Z_x-Z_y-Z_t}\right] \tag{7}$$

where $Z_x = e^{iw\Delta x}$, $Z_y = e^{iw\Delta y}$, and $Z_t = e^{iw\Delta t}$.

The denominator is the Z-transform of the 3-D Laplacian. The zero frequency term of the Z-transform of the denominator is neglected. This means that the resulting surface in space will have an unknown constant shift applied to it. However, by adding the t dimension and assuming the gradient in the t direction to be all ones, I am ensuring that the integrated time varies smoothly in the t direction.

Integrating in three dimensions enforces vertical smoothness. In this case, the dip in the t direction is all ones. However, it isn't necessary to use ones. If zeros were used instead then this would ensure that the time-shifts do not change much in the t direction. By integrating in 3-D, I prevent my method from swapping sample positions in time. Unfortunately, by preventing swapping of sample values this method cannot flatten data with overturned reflections. As will be discussed later, having a smooth output is beneficial if I plan to iterate on the result to remove any residual structure caused by dip changing with depth.

A major drawback to this approach rather than the 2-D approach is it involves taking a 3-D FFT. This is more computationally taxing and may restrict the size of the data volumes being integrated.

FAULTS

To handle faults, I will have to leave the Fourier domain behind. The Fourier based method will estimate erroneous dips across faults. It will then try to honor these erroneous dips creating a result that behaves erratically. However, in the time-space domain, I should be able to handle all faults that have at least one half of the fault tip-line within the data cube. My approach is to create a masking operator ($\mathbf{W}$) that will throw out dip estimates along faults. The method will try to remove all deformation except at the faults where it will allow complete slippage.

I want to find the time shifts, $\mathbf{t}(x,y)$, such that their gradient (∇) is the dip, $\mathbf{p}(x,y)$. This sums across time-slices and is similar to equation (5). A time-space equivalent of equation (7) has also been implemented. I assume the dip, $\mathbf{p}(x,y)$, is not a function of the unknown $\mathbf{t}(x,y)$ and write the fitting goal:

$$\nabla \mathbf{t} \quad \approx \quad \mathbf{p}. \tag{8}$$

This is multiplied by the masking operator ($\mathbf{W}$) to throw out fitting equations at the faults as:

$$\mathbf{W}\nabla \mathbf{t} \quad \approx \quad \mathbf{Wp}. \tag{9}$$

The time shifts ($\hat{\mathbf{t}}$) can be found in a least-squares sense with:

$$\hat{\mathbf{t}} \quad \approx \quad (\nabla' \mathbf{W^2} \nabla)^{-\mathbf{1}} \nabla' \mathbf{W^2 p}. \tag{10}$$

Faulted model - test case

The synthetic model seen in Figure 1 has a fault starting from the center. The dip is constant with depth. Surfaces within this model gradually climb in the clockwise direction, much like a parking garage. This model is flattened using a Fourier space method in Figure 1. Because of erroneous dips at the fault, it does a poor job of flattening. Figure 2 shows the results of applying the T-X space approach using conjugate gradients and a weight that throws out fitting equations at the fault as in equation (10). Notice it is now well flattened. Similarly, using the 3-D integration version in T-X space, shown in Figure 3, the results are adequately flattened.

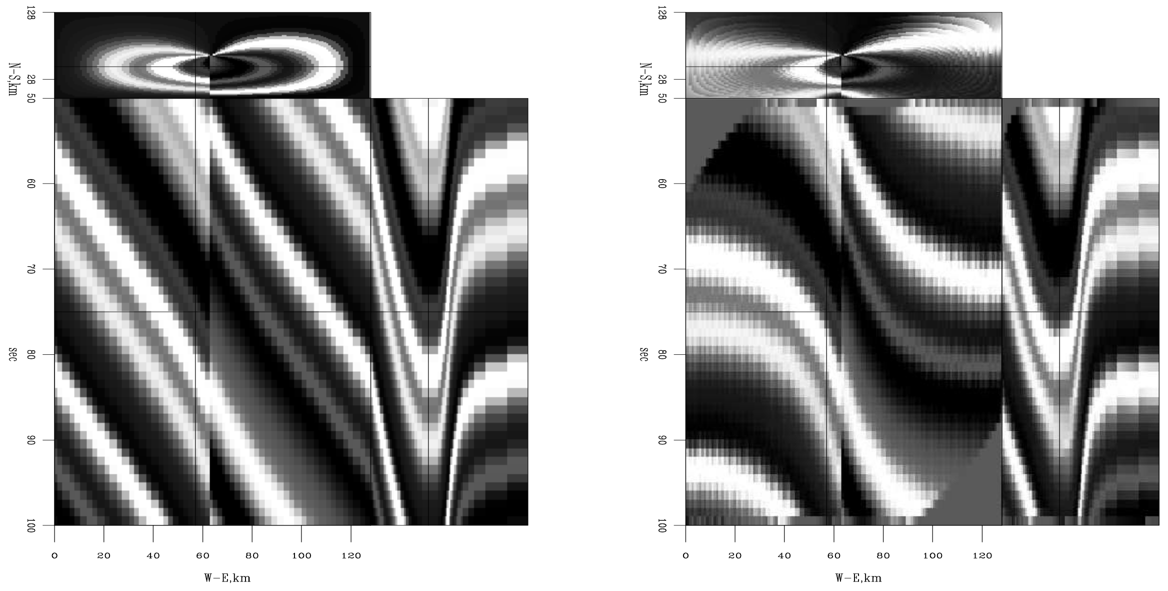


Figure 1: Faulted model. Left is unflattened. Right is flattened. It does a poor job because it calculates erroneous dips at the fault. jesse1-parking.2D_sm [ER]

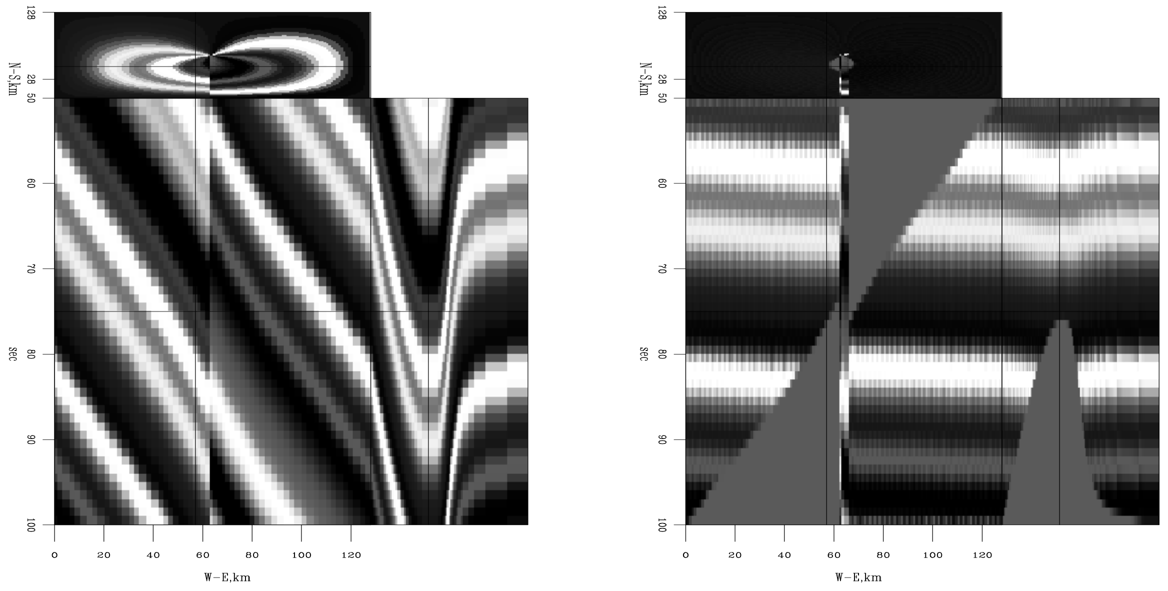


Figure 2: Same as Figure 1 except solved in T-X space rather than Fourier. This allows a weight to be applied to remove fitting equations corrupted by bad dips at the fault. Notice it does a much better job at flattening. jesse1-parking.time [ER]

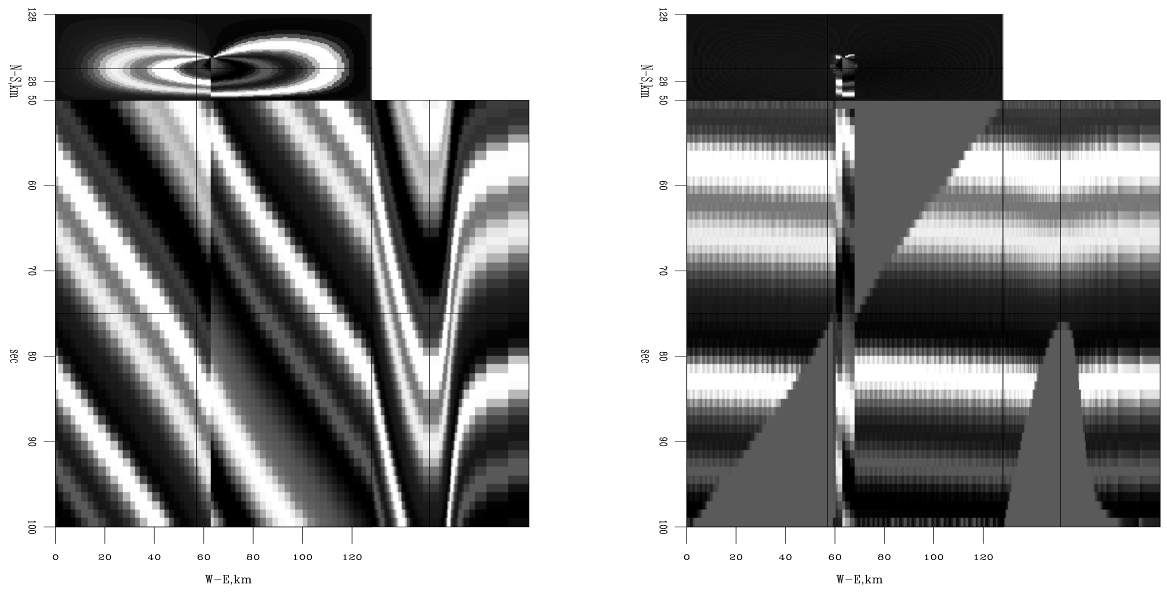


Figure 3: Faulted model in T-X using 3-D integration jesse1-parking_3D.time [ER]

Although completely separated fault blocks present an obvious challenge to this method because the dip is unknown at fault discontinuities, I have some plausible solutions. The fault slip displacement would be the perfect dip information at a fault. If you have a fault block that is bounded on all sides by faults within the 3-D cube, then there is really no way to remove the deformation across the faults without at least knowing two points that correlate on each side of the fault. From an interpretation standpoint it will be necessary to restore both sides of the fault to see the best geological picture. Under such circumstances it maybe necessary to first calculate the fault contours of the fault using a method similar to that described in Lomask (2002) and put the values of the fault slip into the dip cube as input into the flattening process. This fault contour procedure basically just cross-correlates both sides of each fault.

The weights applied to the residual, equation (9), to throw out fitting equations at faults and to weaken fitting equations with low dip semblance or high noise can be created from rough fault models or coherency cubes. Additionally, this method still can be integrated with automatic fault tracking schemes. The output from these programs can be passed to my flattening scheme and used as fault weights. The fault contours, which are an easy output from the flattening method, can be used to quality check and improve the automatic fault tracking output.

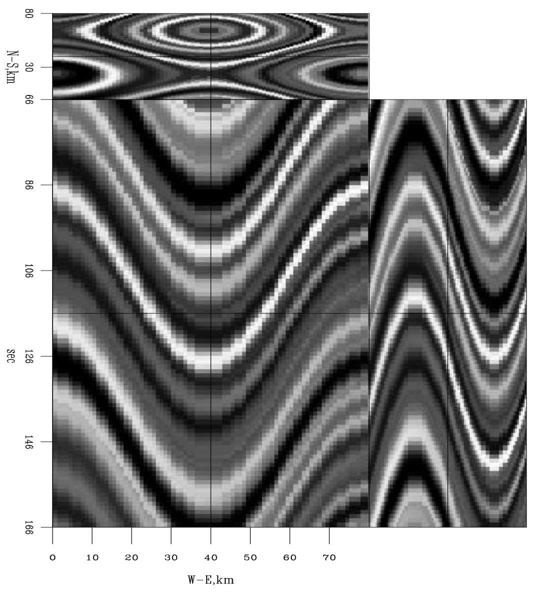
PINCH-OUTS, DIP IS CONSTANT WITH DEPTH

Pinch-outs can be thought of as a single horizon that splits into a double horizon. If the reference trace for flattening goes through the single horizon, then the shifts from the flattening process should collapse the double horizon into one and therefore the solution is non-unique. On the other hand, if the reference trace goes through the double horizon, then the flattening process should result in replicating or stretching the single horizon into two.

Warped pinch-out model - test case

This pinch-out model was created from a series of flat reflectors convolved with wavelets that gradually increase in dominant frequency from one corner of the model to the other. These flat reflectors were then warped with a cosine function. The 2-D and 3-D flattening results are shown in Figures 4 and 5, respectively. Here, again, both 2-D [equation (5)], and 3-D [equation (7)], integration methods do a similar job. There still is some residual warping in both. Since these flattening methods are operating on time-slices, the subtle pinch-out shifts are too small to be removed with one pass of this method. Also, the smoothing of the dips reduces sensitivity to the pinch-out.

In summary, the pinch-outs in this model are not severe enough to determine how these flattening methods deal with the issues non-unique and replicating solutions. These issues need further evaluation with models that have more dramatic pinch-outs.



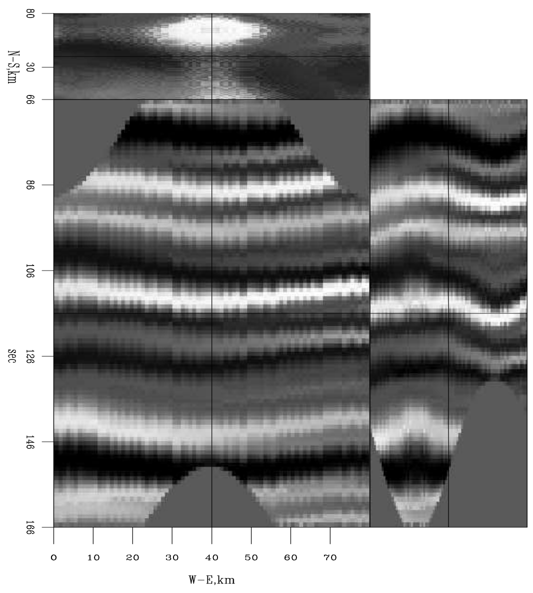
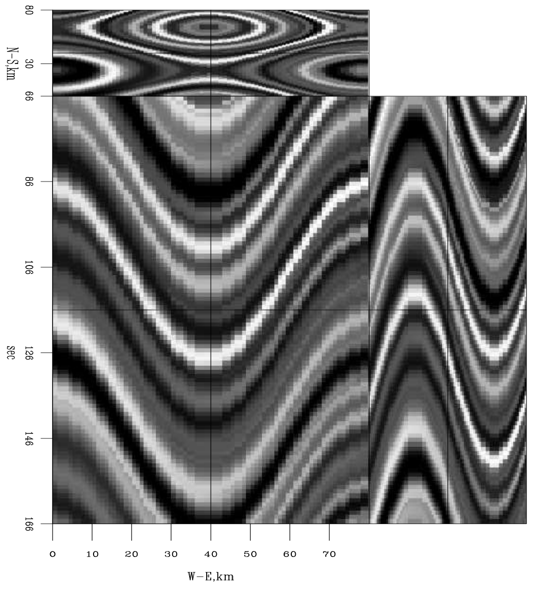


Figure 4: Pinch-out model warped by a cosine function. Left is unflattened. Right is flattened using the 2-D integration method described in equation (5). jesse1-pinchout_cos.flat_comp2D [ER]

PINCH-OUTS, DIP VARYING WITH DEPTH

The basic assumption of the Fourier methods described in equations (5) and (7) is that the shifts required to flatten a cube can be determined from time-slices. In other words, the dip is invariant with depth. The following model has reflectors with pinch-outs that become steeper with depth.



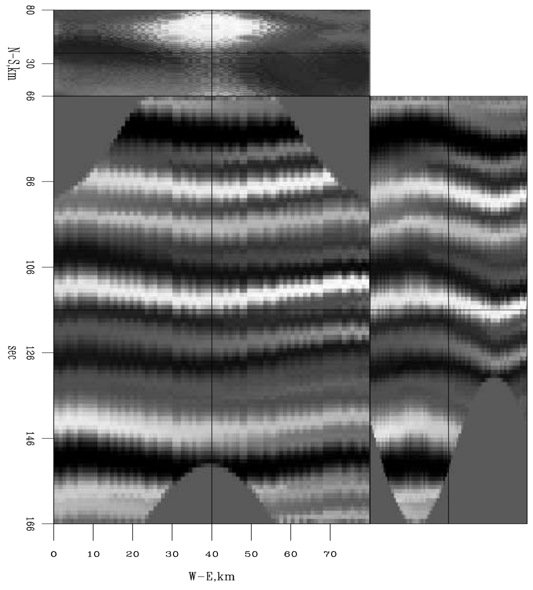


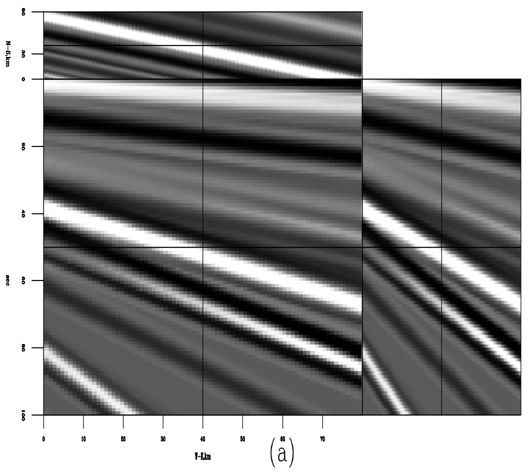
Figure 5: Same as Figure 4 except using 3-D integration described in equation (7). jesse1-pinchout_cos.flat_comp3D [ER]

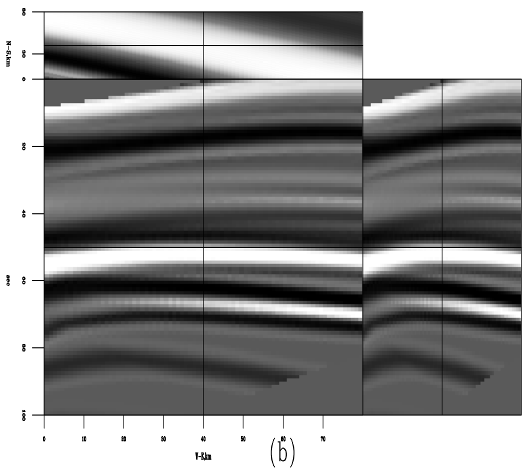
Thinning model - test case

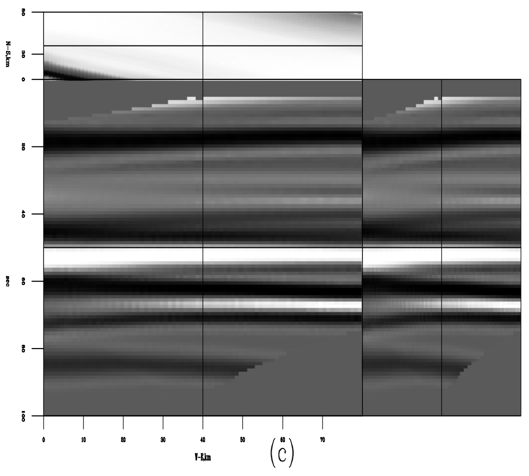
This pinch-out model was created by taking planar reflection coefficients and gradually increasing dip with depth. The reflection coefficients were then convolved with the same wavelet. This creates the numerous pinch-outs seen in Figure 6a. This model should cause inaccuracies in the flattening method because I assume dip is not a function of time. Surprisingly, both methods in equations (5) and (7) are able to remove most of the deformation in one iteration. The results of one iteration of equation (7) is shown in figure 6b. To implement this iteratively, it is necessary to use equation (9) to apply weights where gaps were created by the previous flattening iteration.

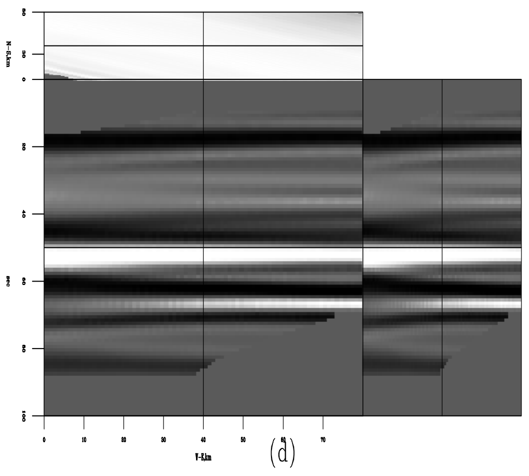
Since the structure is changing with depth, one iteration will not necessarily provide sufficient information for flattening the data. This is mainly because I am assuming that dip is not a function of time by integrating on only time-slices. In this case, the output of the flattening method will need to be flattened again, possibly several times. Repeated application will also improve deficiencies in the flattening caused by inaccuracies in the dip estimation.

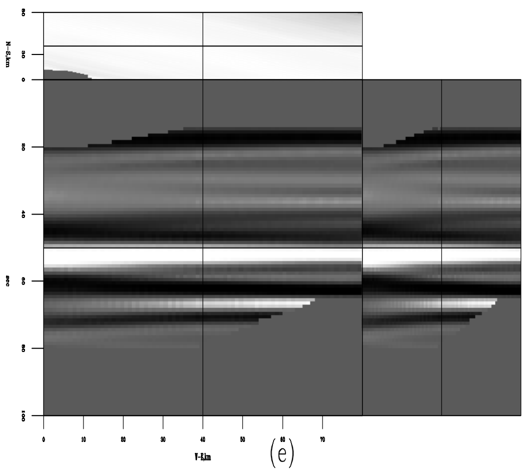
Figure 6 shows the results of five iterations of the flattening method. The image shrinks with each subsequent iteration because the weight is growing to account for the smoothing of dips. This can be rectified by adding a regularization term to the fitting goal in equation (9) to fill empty spaces. Notice that after five iterations, the horizontal view is almost all white indicating that it is flat.


(a)


(b)


(c)


(d)


(e)

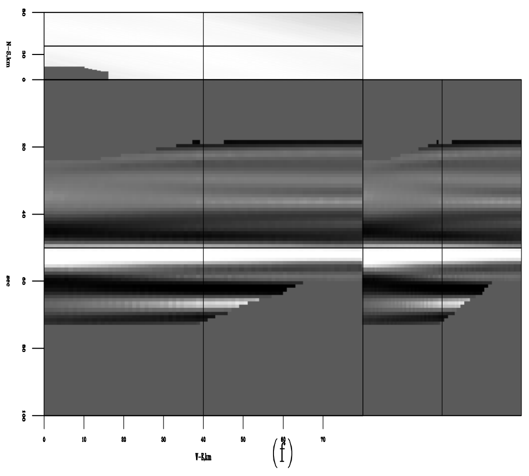

(f)

Figure 6: (a) The thinning model. (b) One iteration of flattening applied. (c) Two iterations. (d) Three iterations. (e) Four iterations. (f) Five iterations. Notice that the horizontal view is almost all white indicating that it is flat. jesse1-thinning_compos.time [ER]

UNCONFORMITIES

Angular unconformities present a major challenge for this methodology. An angular unconformity is a horizon-like feature that has two dips calculated everywhere. One dip for the layers above and another dip for the layers below. These can be handled in different ways.

Perhaps the simplest way to handle unconformities is to break up the cube. If an unconformity is already identified, then the data can be broken up into different cubes, flattened separately, and then recombined.

Analytical solution for calculating time shifts

Another possible solution would be to solve the following equation to find the absolute time ($t(x,y)$) that minimizes (Fomel, 2002, personal communication)

$$J(t) = \int\int \left[\left(p_x(t,x,y) - \frac{\partial t}{\partial x} \right)^2 + \left(p_y(t,x,y) - \frac{\partial t}{\partial y} \right)^2 \right] dx\, dy \tag{11}$$

where p_x is the dip in the x direction and p_y is the dip in the y direction.

Although this is for one horizon, it will be useful to solve this equation for all horizons simultaneously. This would allow the problem to be regularized and its solution to be truly global. In particular, it would no longer be necessary to iterate to remove residual structure in cases of the dip changing with time.

The Euler-Lagrange equation is used in calculus of variations (Farlow, 1993) to find a function (t) that will minimize a functional (J). It is analogous to the calculus expression for finding x to minimize the function $f(x)$ in:

$$\frac{\partial f(x)}{\partial x} = 0. \tag{12}$$

The Euler-Lagrange equation is applied to equation (11) to find (a simplified 2-D development is shown in Appendix A):

$$\frac{\partial^2 t}{\partial x^2} + \frac{\partial^2 t}{\partial y^2} = \frac{\partial p_x}{\partial x} + \frac{\partial p_y}{\partial y} + \frac{1}{2}\frac{\partial ({p_x}^2 + {p_y}^2)}{\partial t}. \tag{13}$$

Figure 7 is a 2-D example summarizing this flattening process for a single horizon. The 2-D version of equation (13) is:

$$\frac{\partial^2 t}{\partial x^2} = \frac{\partial p_x}{\partial x} + \frac{1}{2}\frac{\partial {p_x}^2}{\partial t}. \tag{14}$$

Figure 7a is a cartoon image of the horizon to be flattened. Figure 7b shows the dip field of that horizon. Figure 7c is the output time field. In this case, we are looping over the output field. Each value in Figure 7c contains time values of where to sample the data in order to flatten it. Figure 7d shows the time shift values from the output time field for this horizon.

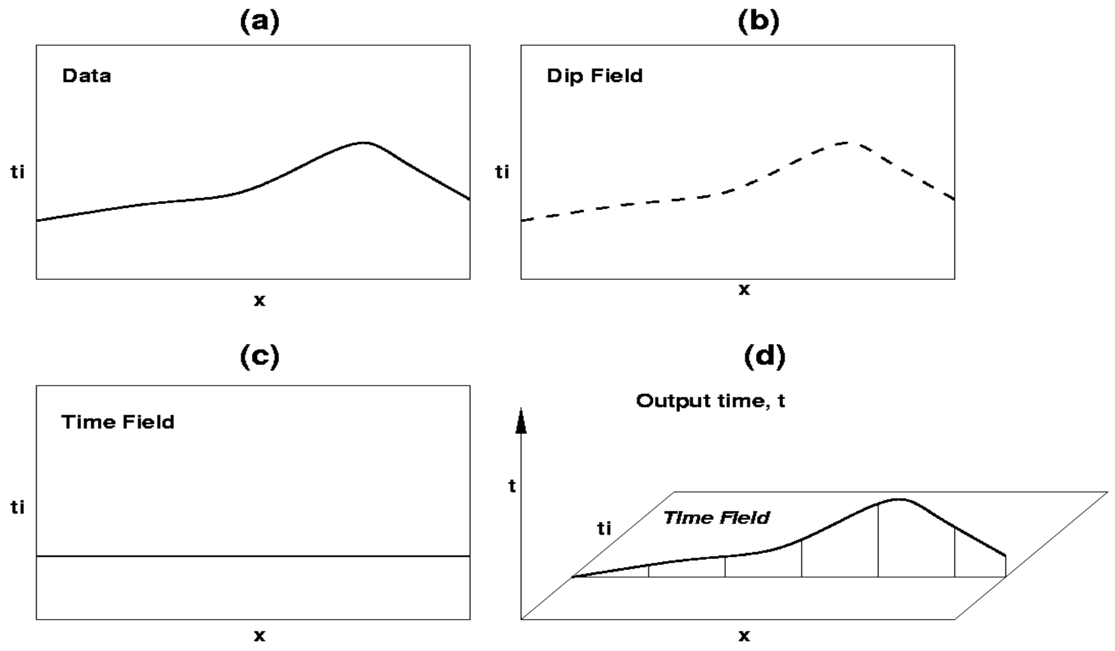


Figure 7: (a) A 2-D cartoon image of a horizon to be flattened. (b) A 2-D image of the dip field showing only dip values along the horizon. (c) The output time field showing where the horizon will be mapped. (d) The calculated time shift values of the time field. jesse1-1horiz [NR]

Equation (13) is non-linear and therefore tricky to solve because dip functions,p_x and p_y, are dependent on the unknown, t, and the last term is taking a derivative with respect to the unknown, t.

Incidentally, dropping the last term from equation 13, the equation can be simplified to:

$$\frac{\partial^2 t}{\partial x^2} + \frac{\partial^2 t}{\partial y^2} = \frac{\partial p_x}{\partial x} + \frac{\partial p_y}{\partial y}. \qquad (15)$$

This is another way of writing equation (3), except that here the dip is a function of the output, t. Repeated applications of this equation partially accommodates for the dropping of the last term.

CONCLUSIONS

Overall, the ability of these flattening schemes to flatten 3-D volumes of data in complex geology is steadily developing. A method for handling faults has been defined but needs more testing. The issues related to unconformities and pinch-out are being addressed but still need more development and testing.

The T-X domain approach was able to flatten the faulted model. This was possible because half of the fault's tip-line was encased within the data cube. If the fault were to cut all the way across the data cube then it would be necessary to first calculate the slip distribution along the fault and use the slip values as dips.

Multiple iterations of the flattening method were able to remove the residual structure from the test case with structure varying with depth. The data cube was essentially flat after four iterations. The anticipated issues of multiple data points mapping into a single point associated with pinch-outs have not yet been addressed because the dip becomes too small to be sensitive to this.

The two possible solutions for dealing with angular unconformities, need to be tested. The first solution is to merely identify the unconformity, break the data into different cubes, flatten separately, and then recombine. The second solution requires solving a challenging differential equation but may require less picking.

The present procedure for applying the shifts to cubes is very simplistic and needs enhancement. The current approach is to hold the central trace constant and shift all other traces vertically to match it. However, a procedure for actually applying the shifts in the presence of numerous non-vertical faults, over-turned beds, pinch-outs, and unconformities needs to be developed.

ACKNOWLEDGMENTS

I would like to thank Jon Claerbout and Sergey Fomel for many useful insights.

REFERENCES

Bienati, N., and Spagnolini, U., 1998, Traveltime picking in 3d data volumes: 60th EAGE Meeting, Extended Abstracts.

Bienati, N., and Spagnolini, U., 2001, Multidimensional wavefront estimation from differential delays: IEEE Trans on Geoscience and Remote Sensing, , no. 3, 655–664.

Bienati, N., Nicoli, M., and Spagnolini, U., 1999a, Automatic horizon picking algorithms for multidimensional data: 61st EAGE Meeting, Helsinki, Extended Abstracts.

Bienati, N., Nicoli, M., and Spagnolini, U., 1999b, Horizon picking for multidimensional data: an integrated approach: Proc. 6th International Congress of the Brazilian Geophysical Society, Rio de Janeiro.

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

Claerbout, J., 1999, Geophysical estimation by example: Environmental soundings image enhancement: Stanford Exploration Project, `http://sepwww.stanford.edu/sep/prof/`.

Farlow, S. J., 1993, Partial differential equations for scientists and engineers: Dover Publications.

Fomel, S., 2001, Three-dimensional seismic data regularization: Ph.D. thesis, Stanford University.

Lomask, J., and Claerbout, J., 2002, Flattening without picking: SEP–**112**, 141–150.

Lomask, J., 2002, Fault contours from seismic: SEP–**111**, 295–309.

APPENDIX A

Euler-Lagrange solution for flattening

The Euler-Lagrange equation can be used to find the absolute time ($t(x,y)$) that minimizes (Fomel, 2002, personal communication)

$$J(t) = \int\int \left[\left(p_x(t,x,y) - \frac{\partial t}{\partial x} \right)^2 + \left(p_y(t,x,y) - \frac{\partial t}{\partial y} \right)^2 \right] dx\, dy \tag{A-1}$$

where p_x is the dip in the x direction and p_y is the dip in the y direction.

This can be simplified for the 2-D case to find the absolute time ($t(x)$) that minimizes

$$J(t) = \int ((p_x(t,x) - \frac{\partial t}{\partial x})^2\, dx \tag{A-2}$$

The Euler-Lagrange equation (Farlow, 1993) is used to find the function ($t(x)$) that minimizes the equation of this form

$$J(t) = \int F(x,t,t')\, dx \approx 0 \tag{A-3}$$

where the unknown t is a function of x. The Euler-Lagrange equation is

$$\frac{\partial F}{\partial \overline{t}} - \frac{d}{dx}\left[\frac{\partial F}{\partial \overline{t'}} \right] = 0 \tag{A-4}$$

To apply this equation I find

$$\frac{\partial F}{\partial t} = 2\left[p_x - \frac{\partial t}{\partial x} \right] \frac{\partial p_x}{\partial t} \tag{A-5}$$

and

$$\frac{\partial F}{\partial t'} = -2\left[p_x - \frac{\partial t}{\partial x} \right] \tag{A-6}$$

Substituting equations (A-5) and (A-6) into equation (A-4) and simplifying, I get:

$$\frac{\partial^2 t}{\partial x^2} = \frac{\partial p_x}{\partial x} + \frac{1}{2}\frac{\partial {p_x}^2}{\partial t} \tag{A-7}$$

It is straight forward to extend to the 3-D case.

www.VolvoOceanRace.org
© RICK TOMLINSON
THE VOLVO OCEAN RACE
ROUND THE WORLD
djuice.com
ASSA ABLOY