Stanford Exploration Project, Report 111, June 9, 2002, pages 293–306

Fault contours from seismic

Jesse Lomask[1]

ABSTRACT

Fault contours are created by mapping the slip distribution along a fault surface. Fault contours have many useful applications in geophysics and geology including rock stress analysis, interpretation, and processing. Therefore, automatic calculation of fault contours from 3D data would be very valuable. As the first step in that direction, displacement is calculated from a simple 2D model of faulted seismic data. Making a stationary assumption, the displacement is calculated by fitting a smooth line to a cross-correlagram. The cross-correlagram is created by windowed cross-correlation across the fault. To fit this line, I use a non-linear optimization method that has similarities to Simulated Annealing. This removes most of the displacement which will later allow other methods to solve the non-stationary problem.

INTRODUCTION

A lot of useful information can be gained by understanding the slip distribution along a fault surface. Factors governing the slip distribution include the stress field, proximity to other faults, rock strength, tectonic history, and loading rates. If the slip distribution were easily and quickly obtained, it could be added to interpretation and processing workflows (Pollard, 2001).

A present, seismic fault contours are generated by tediously interpreting many horizons and calculating the slip as shifts between those horizons. As a result, only the interpreted data is used in calculating slips, and the process is very labor intensive and time consuming. In poor data quality areas, this method may be all that works. However, in areas where seismic image quality allows, automatic data-based methods would save time and, in some cases, increase the accuracy of the result.

In this paper, I describe a robust method for calculating the fault slip distributions from seismic data in order to bring the power of fault contours to bear on a number of problems. Figure 1 shows a synthetic 2D seismic section with an interpretted fault and a cartoon of fault contours. I create a cross-correlagram by placing windowed cross-correlations side-by-side. Then a non-linear line fitting program is used to fit a smooth line to the cross-correlagram. he lag values of this line remove most of the deformation of the fault in a model example.

[1]**email:** lomask@sep.Stanford.EDU

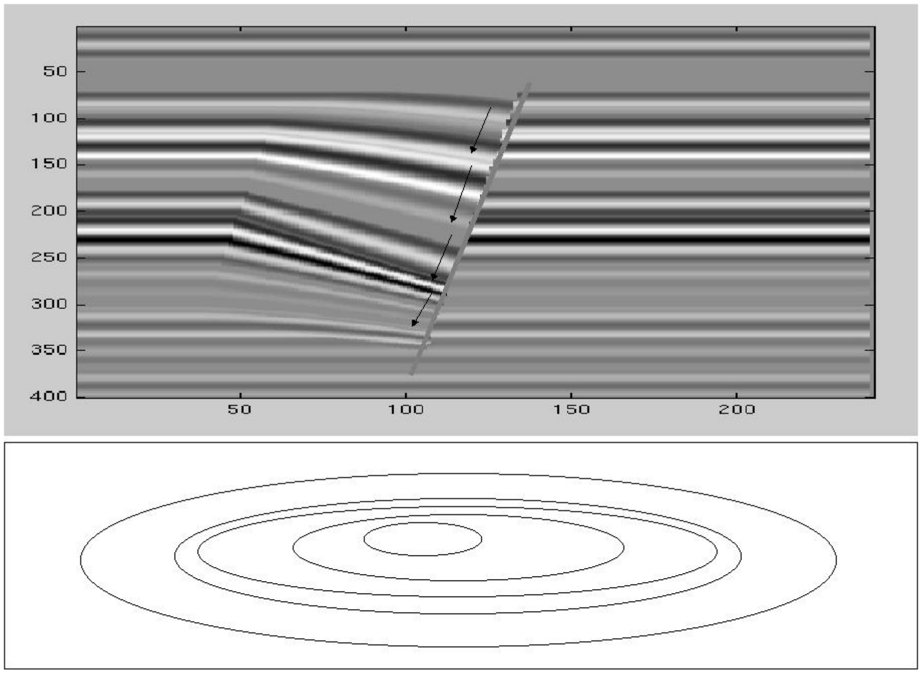

Figure 1: Top: A synthetic seismic section with a fault. Bottom: an idealized fault contour map showing slip distributions along the fault surface. jesse1-fltcont [NR]

APPLICATIONS

Knowing the slip distribution has numerous applications, researchers studying fault mechanics are already familiar with some of these uses. However, it is quite likely that fault slip distribution information has been, in general, under utilized by seismic interpreters and processors. Hopefully, having fault contours automatically calculated will easily integrate them into interpretation and processing workflows.

Stresses and Rock Strengths First of all, one can speculate about the interplay of rock strengths and stresses. The contours are a vital input for researchers studying fault mechanics. They help unravel the stress and strain fields. Understanding the slip distribution can imply slip and position of neighboring faults.

Seals One can use fault contours to see where the mininum displacement should be to insure an adequate seal. For instance, if the thickness of a sand reservoir is known, then you will likely have a shale on sand seal if the displacement is greater than the thickness. In addition, in building a reservoir model, faults which have insignificant displacement would not necessarily be included.

Geological interpretation Combining the contours with other data can help unravel the geological history by revealing periods of growth along the faults in the case of syndepositional faulting. Periods of rapid fault growth can be a result of increased sediment loading that can be tied to mountain building events, eustatic fluctuations, and climate changes.

Well-ties in 3D modeling The slip from a fault contour can be backinterpolated at well locations. In the case of normal faulting, the magnitude of the slip can be compared with the missing section from the wells. In the case of reverse faulting, it can be compared with the repeat section. This information can be incorporated as an additional well tie and input into 3D models.

Automatic horizon interpretation If the slip distribution is known for an entire volume of 3D seismic data, then the faults can be removed. Once they are removed, then auto-interpretation procedures can be used to interpret the horizons with ease.

Constrain processing By tying with the regional stress field, likely faulting strike, dip and magnitude can be estimated. This has potential to help constrain velocity analysis. In addition, fault displacement information from well data can be used as constraints.

Missing section information used in standard fault interpretation Knowing the fault displacement can be incorporated into an interpretation program to help correlate faults. Faults are typically interpreted by an interpreter digitizing fault segments, then correlating the appropriate segments with particular faults, and triangulating fault surfaces. If the slip distribution was calculated on that triangulated fault surface, the interpreter can take advantage of certain fault behavior rules to correlate and measure the quality of the correlation. Figure 2 is a cartoon of a mapview of fault gaps. Fault gaps are mapview

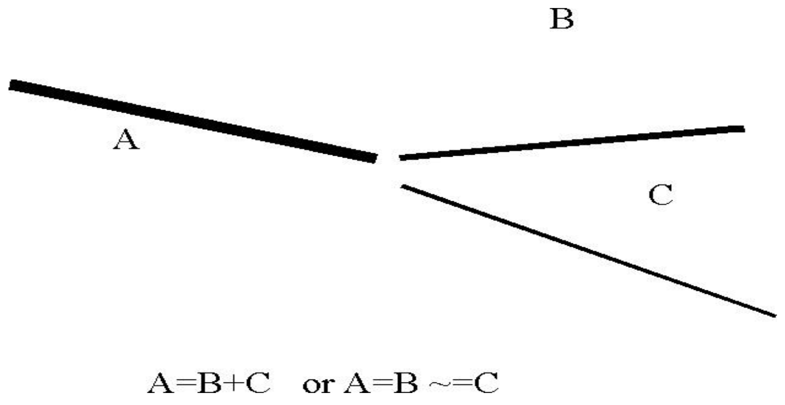

Figure 2: Fault Gaps jesse1-faultgap [NR]

representations of missing sections where normal faulting has caused the horizon to separate. When three fault breaks come together, it is often unclear whether one fault split into two or two different faults just pass closely to each other. In the first case, the missing section on fault A will equal the sum of missing sections on faults B and C. In the second case, perhaps A and B are one fault and C is a separate fault; therefore, the missing section on A will equal B and not equal C.

EXTRACTING LAGS

There are a number of ways to extract the slip distribution along a fault plane. Windowed cross-correlation is the simplest; however, it needs to be constrained. To test this out, I started with a 2D synthetic seismic section. I created 40 randomly spaced reflection coefficients with random magnitudes. I stretched one side to mimic fault deformation, shown in Figure 3. I, then, convolved it with a simple zero phase wavelet, shown in Figure 4.

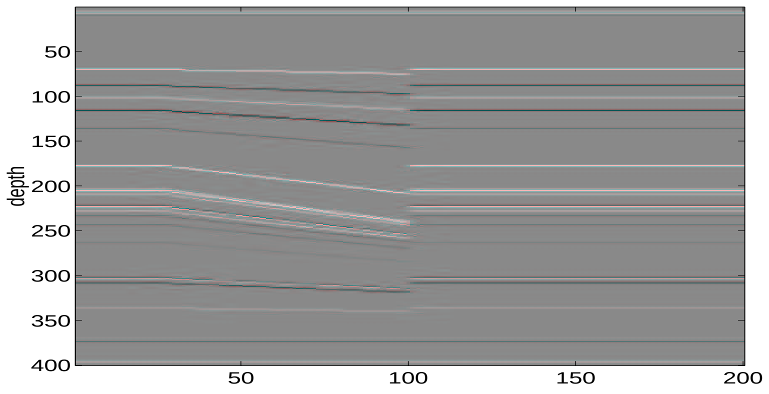

Figure 3: Model of reflection coefficients. jesse1-makemodraw [CR]

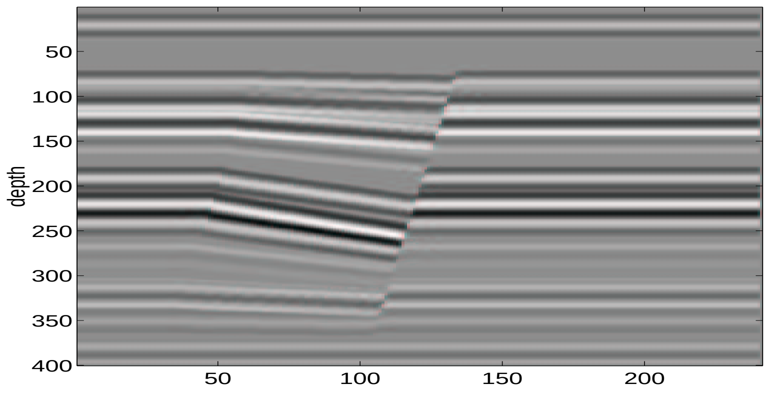

Figure 4: Model after convolving with a source wavelet and bending along a fault. jesse1-makemoda [CR]

The stretching and compressing creates a non-stationary problem. I am still uncertain how much of a problem this would be in real data. In this model, simple cross-correlation techniques can be a good start, but they will not completely remove the deformation.

Regardless, the first step should be to flatten the data on the fault plane, as in Figure 5.

Then one trace can be extracted from each side of the fault. The problem then becomes to find the shift to apply to the trace left of the fault to make it similar to the right side of the fault.

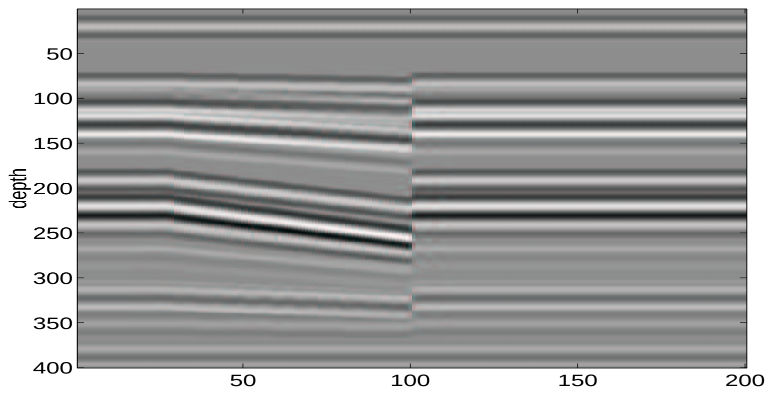

Figure 5: Model after flattening on the fault surface. jesse1-makemod [CR]

Treating as a stationary problem

Rather than trying to account for the non-stationary stretching and compressing, I decided it would be simpler to treat the removal of displacement across the fault as a stationary problem and find a robust method that gives a solution that is close, later addressing the non-stationarity.

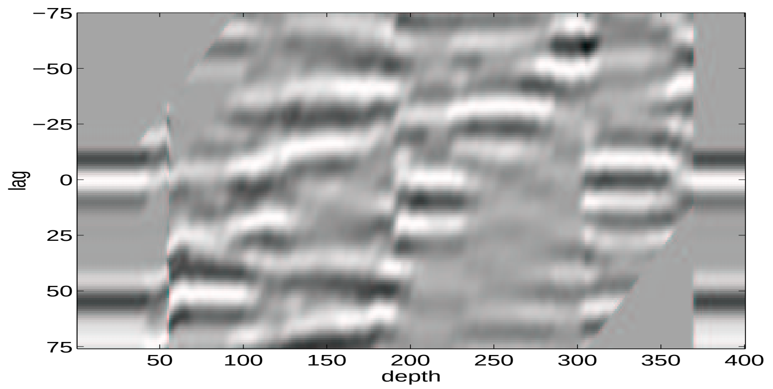

Figure 6: Cross-correlagram: the white color represent mininum values which are actually maximum correlation values. jesse1-makexgram [CR]

The first step was to create a cross-correlagram as shown in Figure 6. This figure is a continuum of windowed cross-correlations from the left and right sides of the fault. As a first step, a smooth line must be fitted to the cross-correlagram. The departure of this line from the center, or zero lag, is the amount of displacement required to remove the fault deformation. Figure 7 shows the cross-correlagram overlaid with the actual displacement used to deform the model plotted. It doesn't land on a peak because in making the model the reflection coeficients were deformed and then convolved with a wavelet, introducing tuning. The tuning causes the cross-correlagram to be skewed from the ideal answer. Also, even knowing the amount of deformation used to create the model, will not remove the deformation completely. Additionally, blindly picking the lag with minimum values will not work as illustrated in Figure 8. In the cross-correlagram, the maximum and minimum are inverted so that we will actually search for the minimum as the line of maximum correlation. In this way, we can keep with the inversion convention of searching for minima.

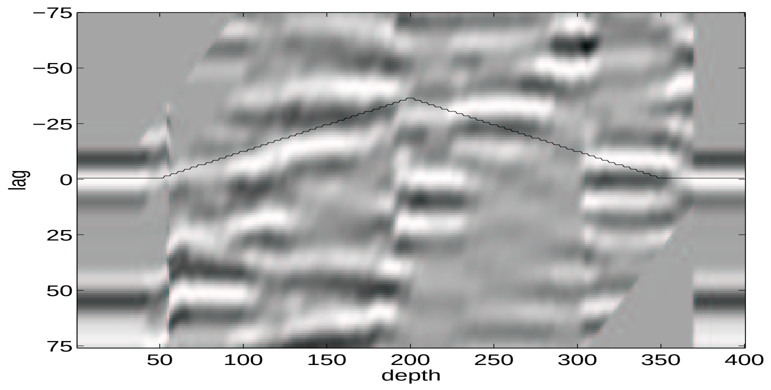

Figure 7: Cross-correlagram overlain with known displacement. jesse1-plotans [CR]

Using linear least squares to fit a smooth line to the maximum path across this cross-correlagram will not work. This is a non-linear problem that is full of local minima. Instead, this problem requires a non-linear approach. A derivative of Simulated Annealing proved to work (Kirkpatrick, 1983).

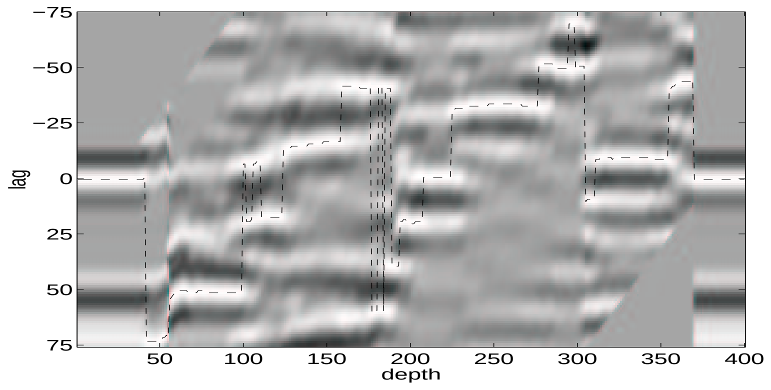

Figure 8: Cross-correlagram with minima at all depths. jesse1-plotmin [CR]

Simulated Annealing

Simulated Annealing is a non-linear optimization method. The idea is to find the global minimum in a way similar to cooling magma forming crystals. If the magma cools too quickly, glass forms. Glass represents local minima. When the magma is cooled slowly enough, then crystal forms. Crystal represents global minima. Rothman used this approach for statics estimation (Rothman, 1985).

We can define our optimization goal as:

$$\mathbf{E_1} = \sum_i \mathbf{C}(m_i) \qquad (1)$$

For this problem, we needed to apply some sort of regularization to ensure that the solution is smooth.

$$\mathbf{E_2} = \sum_i \text{abs}(\frac{\text{d}}{\text{dt}}\mathbf{m}) \tag{2}$$

These two equations can be combined and balanced with ϵ as

$$\mathbf{E} = \mathbf{E_1} + \epsilon \mathbf{E_2} \tag{3}$$

A computation template for the method of simulated annealing is

```
$\mathbf{rate} = \mathbf{f}\,(\mathbf{iteration})$    $\longleftarrow$    define cooling schedule
$\mathbf{m}$    $\longleftarrow$    $\mathbf{0}$.
iterate {
        $\mathbf{n}$    $\longleftarrow$    random numbers
        if  $\mathbf{E_n} < \mathbf{E_m}$ {
            $\mathbf{m}$    $\longleftarrow$    $\mathbf{n}$
        } else {
            $\mathbf{P} = e^{-\Delta \mathbf{E} \cdot \mathbf{rate}}$
            }
        if  $\mathbf{P} >$ random number {
            $\mathbf{m}$    $\longleftarrow$    $\mathbf{n}$
            }

        }
```

The model (**m**) is a vector which defines the extracted lags from the cross-correlagram (**C**). Energy (**E**) is what we are trying to minimize by finding the smoothest path with the lowest energy across the cross-correlagram. The trial model (**n**) is a vector the same size as **m**. We define a **rate** parameter to follow a cooling schedule as a function of iterations. A computation template for the method of simulated annealing is **P**, a probability value assigned to a trial vector (**n**). If a random value is less than this probability value, the trial vector will be accepted even if it has a larger energy. This will allow the solution to escape local minima. This is analogous to cooling magma; when the temperature is high, it is less likely to get locked into position. As the number of iterations increases, **P** becomes smaller, making it less likely to accept a solution with greater energy. If the rate is sufficiently small, this method should converge to the global minimum.

I applied this method to estimate fault slip. The result is in Figure 9. The dots represent the annealing solution. The initial guess for **m** was the maximum, the solid line. This required numerous iterations and, although it appears that it may be heading toward convergence (the light solid line), it may not be. In fact, the added complication of the ϵ parameter makes it quite likely to converge to a flat model. The regularization is causing the slow convergence because randomly changing model points tend to cause an extremely unsmooth solution. Therefore, the regularization part is throwing out most of the possible solutions because they are too rough.

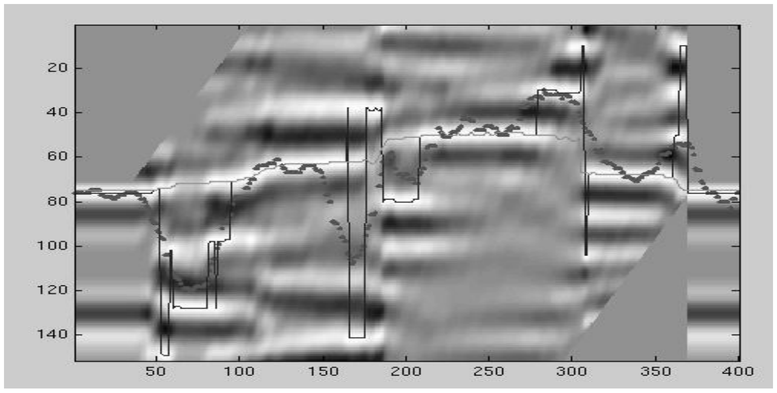

Figure 9: Simulated Annealing result after many iterations, dots. Heavy solid line, picked maximum initial solution. Light solid line, desired result. jesse1-Anneal1bw [NR]

Another approach with similarities to Simulated Annealing

To allow the possible solutions to change radically (sometimes necessary to escape local minima), I tried an algorithm that tries random but smooth solutions at the beginning when the temperature is hot. As the temperature cools, it tries solutions that have more and more roughness. Rather than randomly changing all of the components of **n**, only a few widely spaced samples are randomly changed. Then the rest of the points are interpolated between them and the energy is calculated over the interpolated line.

$$\mathbf{E_1} = \sum_i \mathbf{C}(m_i) \tag{4}$$

This process is repeated and the sampling interval becomes smaller and smaller with increasing interations. This method does not require the use of the probability to get convergence, although that may still have its use later.

A computation template for this method is

```
$\mathbf{sample\ interval} = \mathbf{f}\ (\mathbf{iterations})$    $\longleftarrow$    define cooling schedule
$\mathbf{m}$    $\longleftarrow$    $\mathbf{0}$.  or best initial guess
iterate {
        $\mathbf{n}\ (\mathbf{sample\ interval})$    $\longleftarrow$    random numbers
        $\mathbf{interpolate}\ (\mathbf{n})$
        if  $\mathbf{E_n} < \mathbf{E_m}$ {
            $\mathbf{m}$    $\longleftarrow$    $\mathbf{n}$
            }

        }
```

At coarse sampling intervals, the function is inherently smooth but as the sampling interval gets smaller and smaller, the trial solutions may become too rough. Therefore, I added an additional constraint on the maximum amount of roughness allowed. This results in throwing out trial solutions only near the end when the sampling interval is very small.

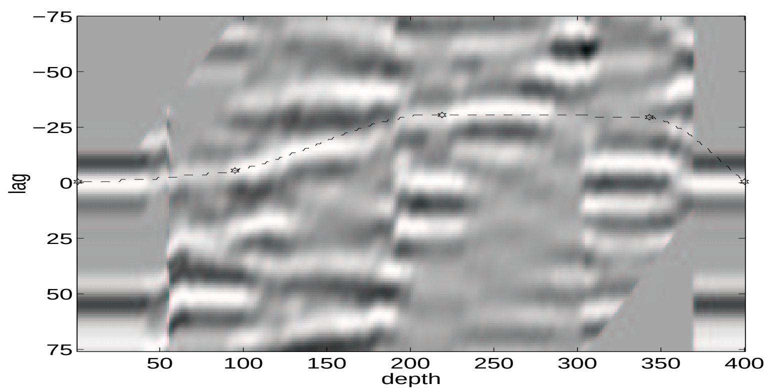

Figure 10: Cross-correlagram with solution overlay after few iterations. The dots reflect the coarse sample interval. jesse1-anneal_course [CR]

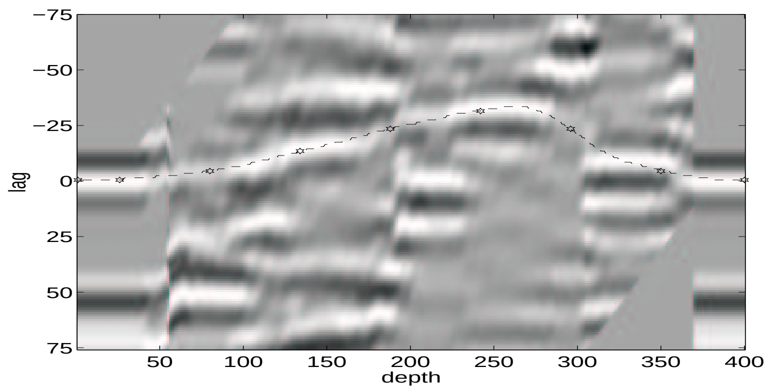

Figure 11: Cross-correlagram with solution overlay after many iterations. The dots reflect the fine sample interval. jesse1-anneal_fine [CR]

Figure 10 shows a frame taken early on in the estimation process. The large dots show the randomly moving points and the dashed line is the interpolated line. Figure 11 shows another frame taken later; the density of the dots illustrates the shrinking sampling interval.

Cooling schedule

As Rothman (1985) points out, the selection of the cooling function is very important and can greatly speed convergence. In my method, I treat the sample interval function as a cooling function. To find a good cooling function, I first created a linear cooling function and plotted how the energy decreased as a function of sample interval. This is shown in Figure 12. In this particular example, there appear to be three sample intervals associated with large drops in energy: 185, 130, and 70. I decided to create a cooling schedule that treats the sized 70 sample interval as the critical temperature. This is shown in Figure 13. The critical temperature is where convergence is most significant.

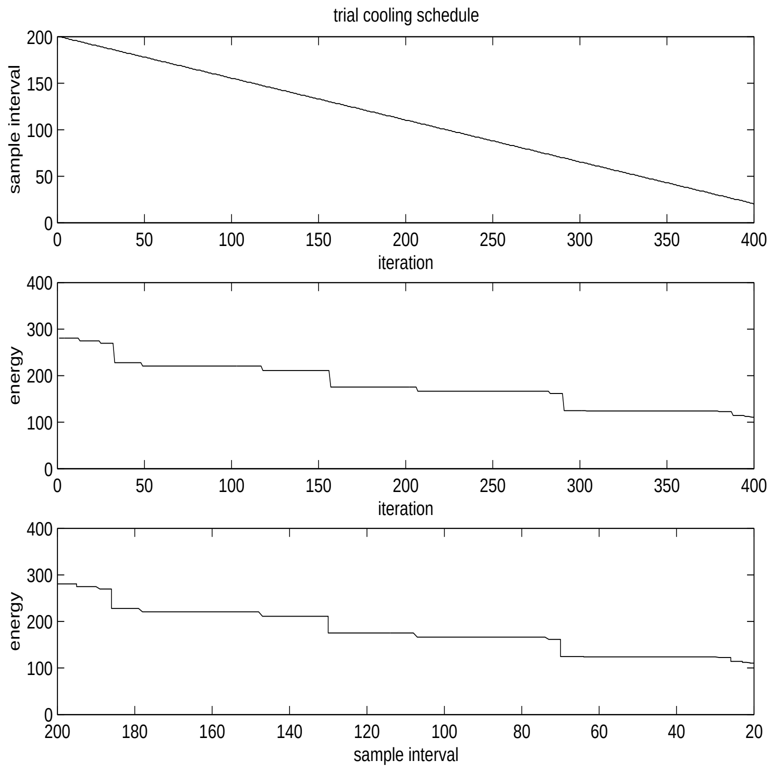

Figure 12: Trial cooling schdule. Top: input cooling schedule drops linearly with iterations. Middle: energy(**E**) drops in steps with iterations. Bottom: energy(**E**) drops in step with sample interval. This is used to determine the more efficient cooling schedule in Figure 13. jesse1-trialschedule [CR]

Notice that in Figure 13, the energy drops off much quicker than in Figure 12 and therefore requires fewer iterations.

Figure 14 shows the result of applying the cooling schedule in Figure 13. It has converged to the desired event. Figure 15 shows the application of the calculated displacements to the left side of the fault. Its results are about the same quality as Figure 16, which shows the results of applying the actual known displacement to the left side of the fault.

CONCLUSIONS AND FUTURE WORK

The iterative method for removing most of the deformation in this model works very well. Calculating a solution to this small model, it takes just a couple of minutes on a PC and successfully removes the fault deformation as is evident in Figure 15.

The next step is to apply this to real data. First to a simple 2D line and then to a 3D volume.

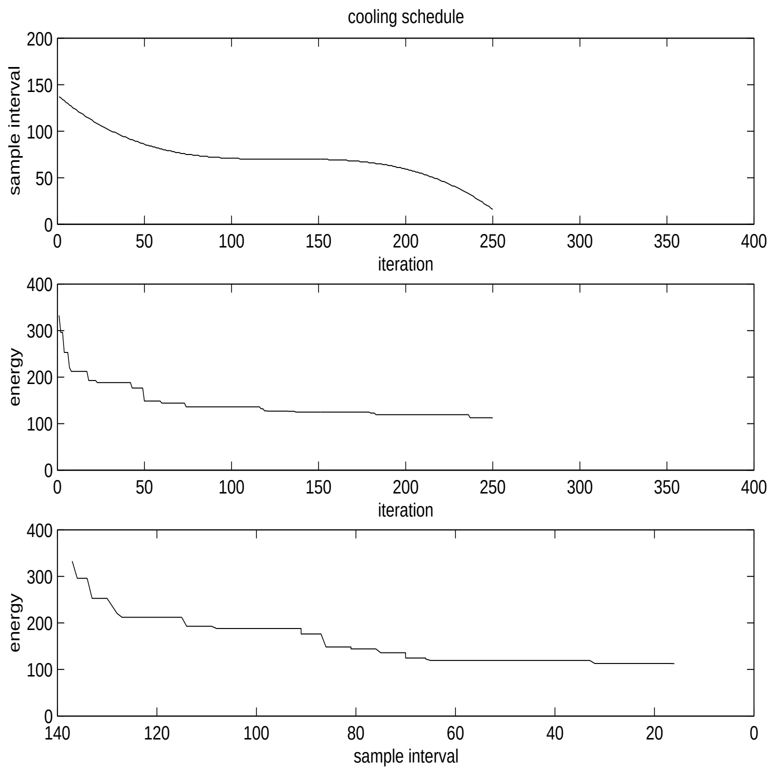

Figure 13: Cooling schedule. Top: input cooling schedule based on trial cooling schedule designed to spend more iterations near the critical tempurature. Middle: energy(**E**) drops in steps with iterations. Bottom: energy(**E**) drops in step with sample interval. jesse1-coolingschedule [CR]

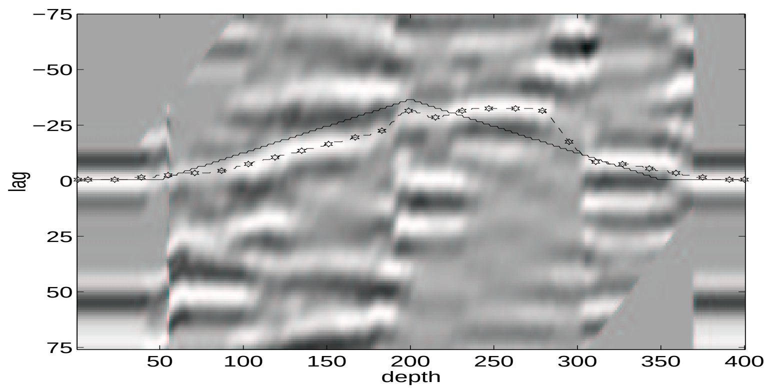

Figure 14: Cross-correlagram with overlay of final solution and known displacement. jesse1-anneal_final [CR]

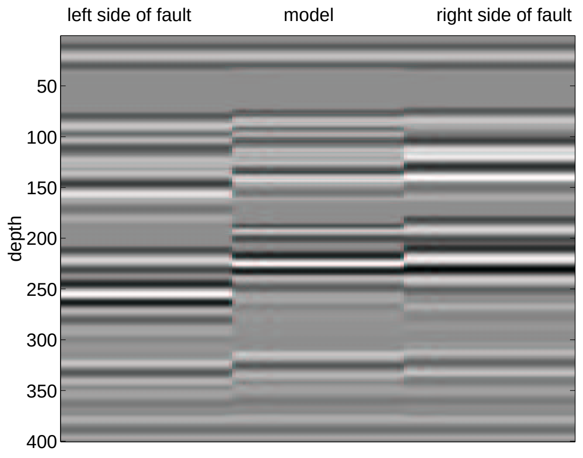

Figure 15: Applied result, the "model" in the center shows the result of applying the calculated displacement to the left side of the fault. jesse1-makefinalmod [CR]

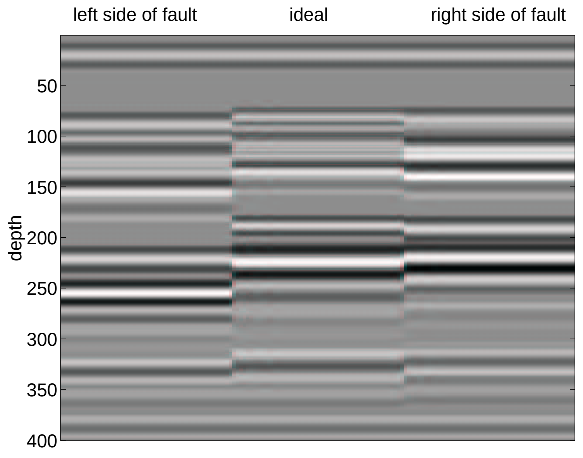

Figure 16: Ideal applied result, the "ideal" in the center shows the result of applying the known displacement to the left side of the fault. jesse1-makefinalideal [CR]

Other types of non-linear solver methods can be tested on this problem as well. Automatic picking algorithms may work faster and easier than my method.

If this method is successful at removing the bulk of deformation on real data, then we may want to address the non-stationary aspect to improve the results.

Lastly, this work can possibly be utilized to aid in automatic fault interpretation. Starting with a raw 3D seismic volume and a coherency cube, fault slips can be used to constrain the automatic interpretation of the faults themselves.

A logical next step to finding the fault contours would be to automatically calculate the entire deformation ellipsiod as outlined in Figure 17. In this simple model it seems simple enough, but in the chaotic world of real geology, that may be an entirely different matter.

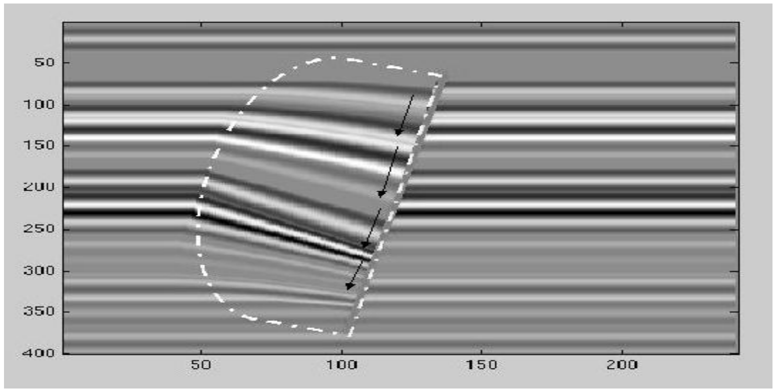

Figure 17: Deformation Ellipse jesse1-ellipse1 [NR]

ACKNOWLEDGEMENTS

During a conversation, Biondo Biondi suggested that we should try to come up with a way of automatically extracting fault slip distributions from seismic.

REFERENCES

KirkpatrickS. snd Gelatt, C. a. V. M., 1983, Optimization by simulated annealing: Science, **220**, 671–680.

Pollard, D. Structural geology and geomechanics:. Internet, 2001. http://pangea.stanford.edu/geomech/Research/Research.htm.

Rothman, D. H., 1985, Large near-surface anomalies: SEP–**45**, 5–18.