ENHANCING FAULTS AND AXIAL PLANES – PROGRAM fault_enhancement

Contents

Overview

The fault enhancement attribute is a post-stack attribute which enhances locally planar features within a seismic attribute volume. We suspect the most common application will be to improve fault images previously approximated by a similarity attribute. However, through proper choice of parameters, one can also enhance unconformities and other discontinuities parallel or subparallel to reflector dip. This algorithm will also enhance axial planes delineated by most-positive and most-negative curvature volumes. In addition to sharpening hypothesized faults, **fault_enhancement** also generates ancillary fault dip magnitude and fault dip azimuth volumes.

Computation flow chart

The input to program **fault_enhancement** includes a primary attribute that approximates the faults or axial planes that you wish to enhance. For faults, this will usually be one of the similarity/coherence attributes computed using program **similarity3d**. For axial planes, this will usually be the most-positive or most-negative principal curvatures. Program **fault_enhancement** will allow the user to suppress or enhance attribute features with respect to reflector dip. For this reason, the inline and crossline components of reflector dip are additional input volumes.

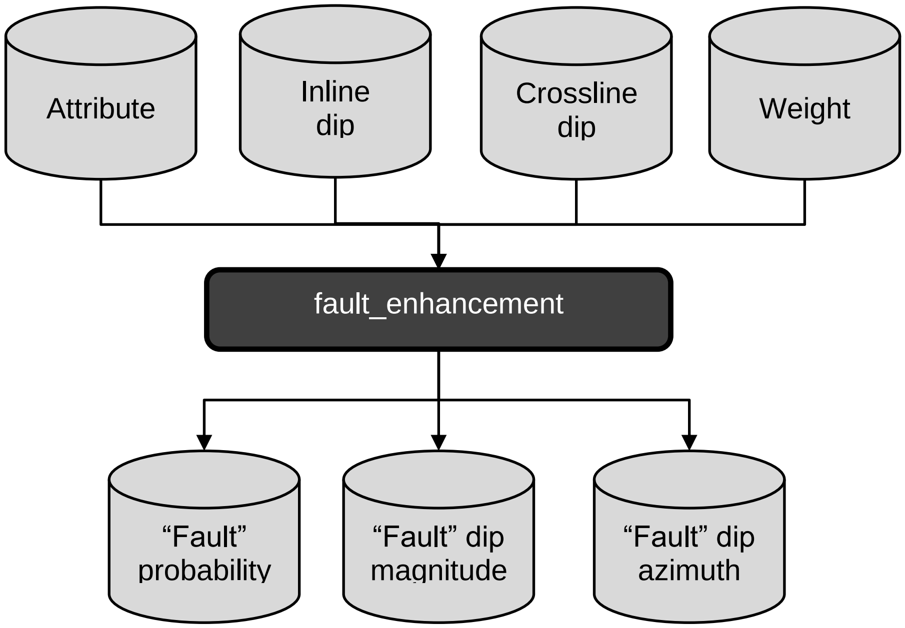


Figure 1. Fault enhancement flow diagram.

Output file naming convention

Program **skeletonize3d** will always generate the following output files:

Output file description	File name syntax
Program log information	fault_enhancement_*unique_project_name_suffix*.log
Program error/completion information	fault_enhancement_*unique_project_name_suffix*.err

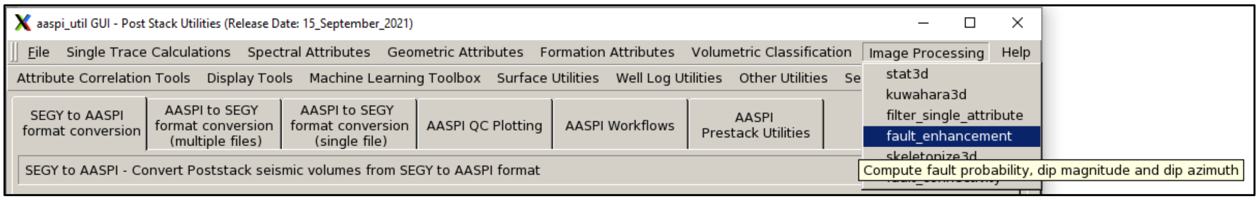
where the values in red are defined by the program GUI. The errors we anticipated will be written to the *.err file and be displayed in a pop-up window upon program termination. These errors, much of the input information, a description of intermediate variables, and any software trace-back errors will be contained in the *.log file.

Program **fault_enhancement** will also generate these output files useful for 3D visualization:

Output file description	File name syntax
Fault probability	fault_probability_*unique_project_name_suffix*.H
Fault dip azimuth	fault_dip_azimuth_*unique_project_name_suffix*.H
Fault dip magnitude	fault_dip_magnitude_*unique_project_name_suffix*.H

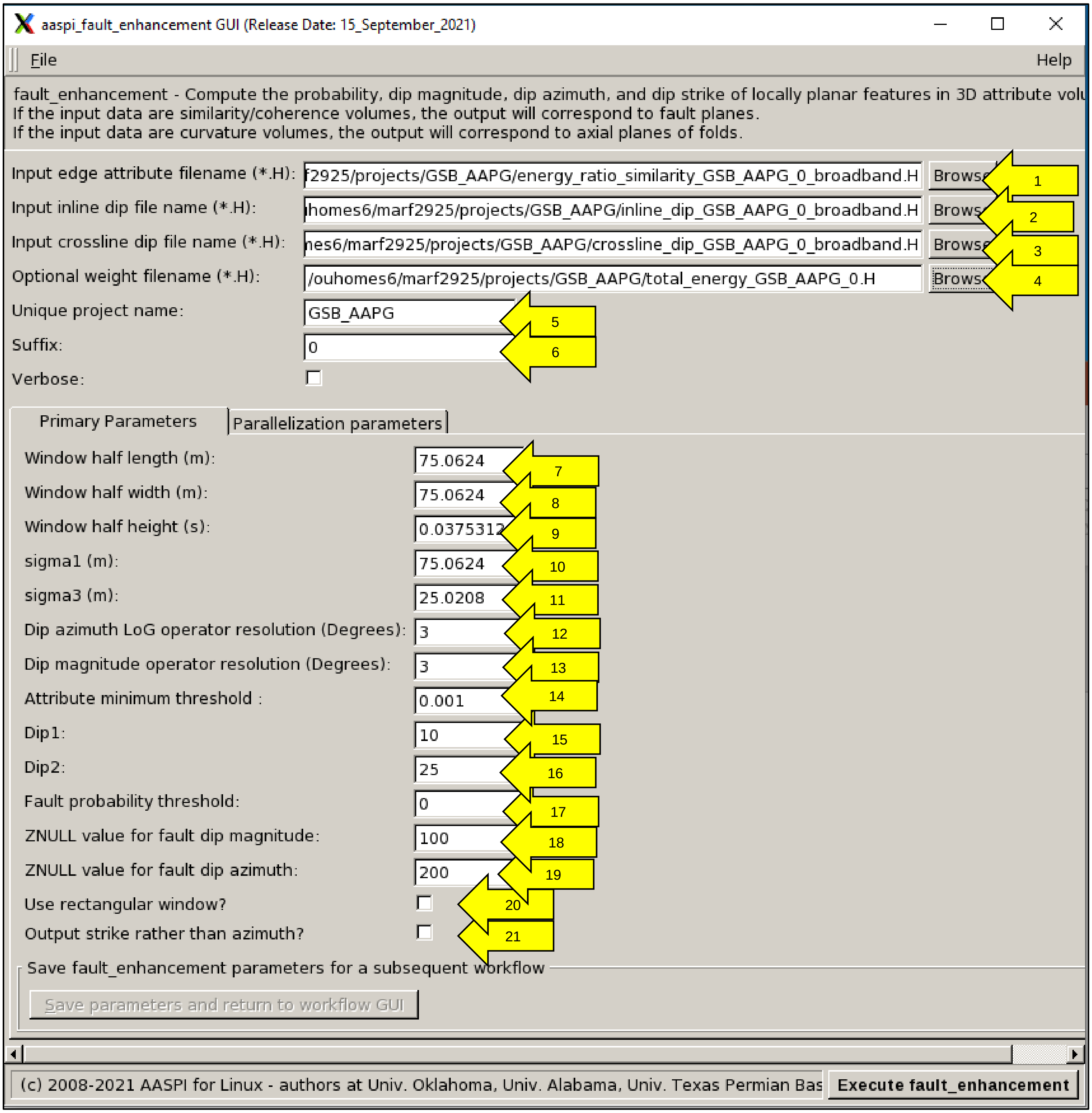
Computing fault enhanced volumes

The fault enhancement program is located under the *Image Processing* tab -> **fault_enhancement** of the main **aaspi_util** window:



The following GUI will appaear:

The user needs to specify (1) the input seismic attribute volume, (2) the inline dip, (3) the crossline dip, and if desired, (4) an optional weighting volume, which in this example is the envelope computed in program **instantaneous_attributes**. These file names are followed by parameters common to most AASPI applications, including (5) a unique project name, and (6) a



suffix. Once the inline dip component has been loaded, the lower three parameters (window length, width, and height) will be filled with default values that can be modified. For time domain data, the value of velocity used in program **dip3d** is used to estimate a vertical window of comparable size to the window length and width. In this example from the Great South Basin (courtesy of NZPM for public use), the cdp increment, dcdp=12.5 m and line increment dline=12.5 m. However, the line index increment, d3=2 rather than 1, giving a crossline trace separation of 25 m. The current default is to use a window whose radius is three times the greater of these two spacings. In the case above, the (7) window half-length and (8) half width are approximately 75 m while the (9) window half height is 37 ms, thereby, which define a spherical operator in 3D.

The parameters (10) sigma1, and (11) sigma3 ($\sigma_1= \sigma_2$ and σ_3 in equation 7) define the width of Gaussian filters used in smoothing and sharpening, respectively. Because of the numerical support of the grid, the user should avoid using values of σ_3 less than the larger of the cdp or line spacing (in this example, 25 m).

Program **fault_enhancement** precomputes the LoG operator as well as the attribute weighting matrix for azimuths ranging between -180° and +180° and dip magnitudes ranging between 0° and 90°. Depending on your data and your computer resources, defining these operators at (12 and 13) 1° increments may cause you to run out of memory. If this occurs, increasing these values to 5° or so will reduce the memory requirements. Initial testing indicates that there on several data volumes indicate no significant differences between sampling at 1°, 3°, or 5°. The computation time is the same, independent as to how densely you store your operators.

Let's assume you use coherence as an input attribute to be enhanced. Internal to the program, the coherence, c, is first converted to a fault probability, p=1-c, such that voxels exhibiting high coherence c≈1 are converted to $p≈0.0$. If all of the values of p in the analysis window are less than the (14) attribute minimum threshold, no fault enhancement is attempted and the resulting fault probability is set to 0.0, thereby significantly reducing the computation cost.

The parameters (15) ϑ_1=*Dip1* and (16) ϑ_2=*Dip2* define a Tukey filter that rejects fault attributes that fall beyond dip magnitude ϑ_1 and retains fault features beyond ϑ_2. If the numerical value of $\vartheta_1 > \vartheta_2$, then fault features subparallel to reflector dip are retained rather than suppressed. For example, if ϑ_1= 10° and ϑ_2=25° then all the discontinuities with a dip less than 10° will be rejected, discontinuities with dip magnitude greater than 25° will be retained, and discontinuities with dip magnitudes falling between 10° and ϑ_2=25° will be suppressed using the filter described in *Figure 2*. In contrast, if ϑ_1=25° and ϑ_2=10° then faults with a dip magnitude $\vartheta>25^0$ will be rejected and discontinuity features subparallel (<10°) to reflector dip will be retained. In general, every voxel in the volume will have a valid fault dip magnitude and azimuth. If the fault attribute (probability) is small, these values are meaningless.

When using interpretation software packages with a flexible definition of opacity such as Petrel, Seisworks, or Voxelgeo, the interpreter simply sets the low values (for example black values) of fault probability to be opaque and high values to be transparent as shown later in this documentation. Many of the less sophisticated (less expensive!) interpretation software packages only allow a constant opacity value for a given volume. In this case, the user can define a cutoff (17) *Fault Opacity* value, below which the fault dip magnitude and fault dip azimuth are set to be user defined (18 and 19) *znull* values. These znull values may depend on your interpretation workstation software. The *znull* value for each volume will also be stored in the output fault dip magnitude and fault dip azimuth *.H files. A simple workflow is then to plot voxels with a znull value to be black, gray, or white.

Numerical experimentation has shown that using a spherical window provides nearly the same result as (20) using a rectangular prism window, but costs 6/π≈2 times less.

Fault planes are defined by a probability and a vector perpendicular to the fault plane. The default output is to generate a fault dip azimuth that is oriented perpendicular to the upward oriented side of a dipping fault plane. However, earthquake seismologists and some workers in the microseismic analysis community prefer to define a fault plane by a (21) strike that ranges between -180° and +180°, where your right-hand lays on the fault face and the strike is defined as the orientation of your thumb.

The parallelization parameters are identical to those in all other AASPI programs running under MPI.

Clicking the *Execute fault_enhancement* button on the lower right submits the program.

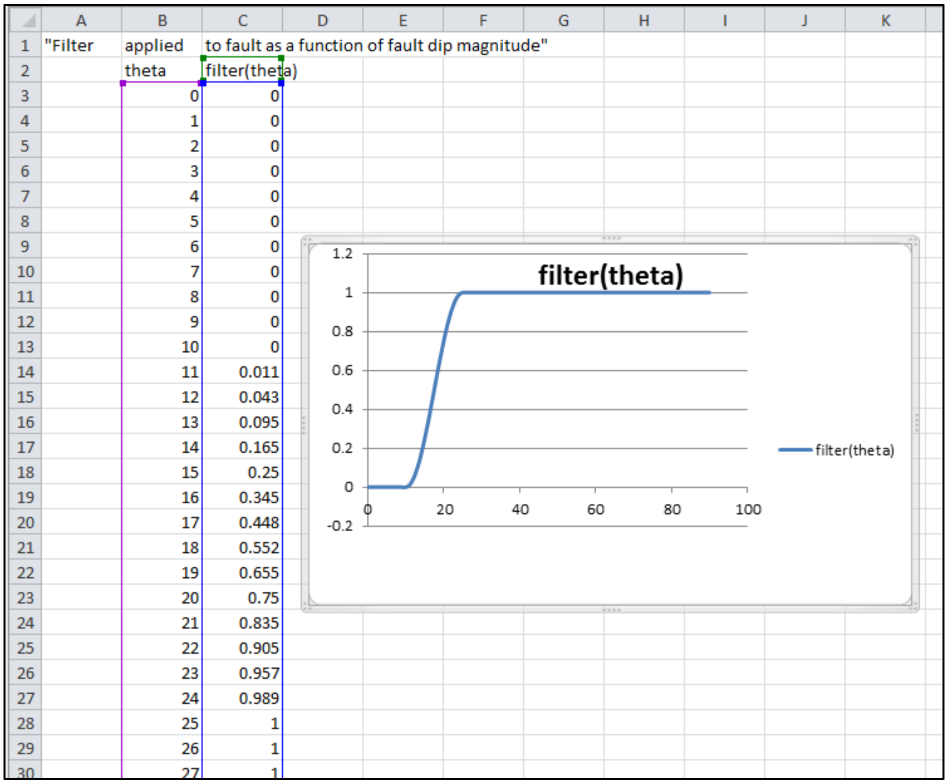
The following files were generated for the parameters chosen above:

```
-rw-r--r-- 1 marf2925 faculty    1334 Jan 16 17:03 fault_enhancement.parms
-rw-r--r-- 1 marf2925 faculty      71 Jan 16 17:03 live_processor_list
-rw-r--r-- 1 marf2925 faculty       0 Jan 16 17:03 fault_enhancement_GSB_small_with_energy_weights.err
-rw-r--r-- 1 marf2925 faculty    2915 Jan 16 17:03 fault_dip_filter.txt
-rw-r--r-- 1 marf2925 faculty    2586 Jan 16 17:03 fault_probability_GSB_small_with_energy_weights.H@@
-rw-r--r-- 1 marf2925 faculty    5459 Jan 16 17:03 fault_probability_GSB_small_with_energy_weights.H
-rw-r--r-- 1 marf2925 faculty    2586 Jan 16 17:03 fault_dip_azimuth_GSB_small_with_energy_weights.H@@
-rw-r--r-- 1 marf2925 faculty    5518 Jan 16 17:03 fault_dip_azimuth_GSB_small_with_energy_weights.H
-rw-r--r-- 1 marf2925 faculty    2588 Jan 16 17:03 fault_dip_magnitude_GSB_small_with_energy_weights.H@@
-rw-r--r-- 1 marf2925 faculty    5521 Jan 16 17:03 fault_dip_magnitude_GSB_small_with_energy_weights.H
-rw-r--r-- 1 marf2925 faculty  106747 Jan 16 17:04 fault_enhancement_GSB_small_with_energy_weights.out
```

The *fault_enhancement.parms* file simply provides parameters to the python script and reads as follows:

```
marf2925@tripolite:~/projects/GSB_small $ cat fault_enhancement.parms
use_mpi=y
processors_per_node=2
node_list="jade:16 kwiatkowski:16 hematite:16"
build_lsf_script=n
build_pbs_script=n
build_slurm_script=n
max_run_time=10
maximum_number_of_tasks_per_node=40
batch_queue=""
unique_project_name="GSB_small"
suffix="with_energy_weights"
verbose=n
weight_fn="/ouhomes6/marf2925/projects/GSB_small/total_energy_d_mig_GSB_small_20_ms_window.H"
attribute_fn="/ouhomes6/marf2925/projects/GSB_small/energy_ratio_similarity_d_mig_GSB_small_20_ms_window_broadband.H"
inline_dip_fn="/ouhomes6/marf2925/projects/GSB_small/inline_dip_d_mig_GSB_small_sum_of_g_and_gh.H"
crossline_dip_fn="/ouhomes6/marf2925/projects/GSB_small/crossline_dip_d_mig_GSB_small_sum_of_g_and_gh.H"
fault_probability_fn="fault_probability_GSB_small_with_energy_weights.H"
fault_dip_azimuth_fn="fault_dip_azimuth_GSB_small_with_energy_weights.H"
fault_dip_magnitude_fn="fault_dip_magnitude_GSB_small_with_energy_weights.H"
fault_dip_strike_fn="fault_dip_strike_GSB_small_with_energy_weights.H"
output_fn="fault_enhancement_GSB_small_with_energy_weights.out"
error_fn="fault_enhancement_GSB_small_with_energy_weights.err"
dip1=10
dip2=25
fault_opacity=-1
fault_dip_magnitude_znull=100
fault_dip_azimuth_znull=200
window_length=75
window_width=75
window_height=0.0375312
sigma1=75.0624
sigma3=25.0208
rectangular_window=n
wr_fault_dip_strike=n
```

The definition of the files is fairly obvious and represent the fault probability, fault dip magnitude, and fault dip azimuth. The *fault_dip_filter.txt* file is an ASCII-format file that can be plotted using excel. In this case it appears as follows (see next page):



	A	B	C	D
1	"Filter	applied	to fault as a function of fault dip magnitude"	
2		theta	filter(theta)	
3		0	0	
4		1	0	
5		2	0	
6		3	0	
7		4	0	
8		5	0	
9		6	0	
10		7	0	
11		8	0	
12		9	0	
13		10	0	
14		11	0.011	
15		12	0.043	
16		13	0.095	
17		14	0.165	
18		15	0.25	
19		16	0.345	
20		17	0.448	
21		18	0.552	
22		19	0.655	
23		20	0.75	
24		21	0.835	
25		22	0.905	
26		23	0.957	
27		24	0.989	
28		25	1	
29		26	1	
30		27	1	

Figure 2. The filter applied to resulting fault probability. In this example, the fault probability of faults having a fault dip magnitude less than 10° to reflector dip will be set to zero, while those having a fault dip magnitude greater than 25° to reflector dip will be retained. The result will be that unconformity and low-reflectivity coherence anomalies that are subparallel to reflector dip will appear as white streaks when plotted against a white-gray-black color bar, as shown in the images below; which indicates that the fault probability of features with dip magnitude less than 10° will be set to zero and those with dip magnitude greater than 25° will be unchanged.

Theory: Fault orientation using the second-order moment tensor

(After Qi et al., 2018)

The second-order moment tensor

Machado et al.'s (2016) fault enhancement workflow is based on earlier work by Barnes (2006), who constructed a second-order moment tensor of an edge attribute to determine the hypothesized fault anomalies' orientation. The second-order moment tensor is built from a coherence attribute a_m within an M-voxel analysis window, where a_m has been modified so that coherent portions of the survey have a value of zero. (The same algorithm also enhances edge anomalies computed using Sobel-filters and aberrancy). We modify Machado et al.'s (2016) algorithm by observing that all coherence anomalies are not equally important. Specifically, we wanted to minimize fault "stair step" artifacts in coherence images of dipping faults. Lin and Marfurt (2016) recognized that seismic migration images the seismic wavelet perpendicular to the reflector dip. Using an analytic seismic trace (the original trace and its Hilbert transform) as input, a reflector discontinuity occurs where the instantaneous envelope is maximum. In the absence of nearby reflectors, away from this point, wavelet sidelobes result in the discontinuity continuing perpendicular to the reflector rather than along the true fault face. To partially address this problem, we assign a greater weight to discontinuities where the wavelet envelope (or energy) is strongest. Additionally, we modified Machado et al.'s (2016) algorithm by noting that the 2nd moment tensor should be computed about the center of mass rather than about the center of the window, further improving the results. We define the location of the center of mass, $\boldsymbol{\mu}$ of coherence anomalies a_m within the analysis window to be:

$$\begin{cases} \mu_1 = \frac{\sum_{m=1}^{M} W_m a_m x_{1m}}{\sum_{m=1}^{M} W_m a_m} \\ \mu_2 = \frac{\sum_{m=1}^{M} W_m a_m x_{2m}}{\sum_{m=1}^{M} W_m a_m}, \\ \mu_3 = \frac{\sum_{m=1}^{M} W_m a_m x_{3m}}{\sum_{m=1}^{M} W_m a_m} \end{cases} \tag{1}$$

where x_m the vector distance of the m^{th} voxel from the center of the analysis window and where W_m is a weight that depends on the local reflector strength. The 2nd moment tensor in the analysis window can be written as:

$$C = \begin{pmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{pmatrix}, \tag{2}$$

where the components of the 2nd moment tensor are:

$$I_{jk} = \sum_{m=1}^{M} W_m (x_{jm} - \mu_j)(x_{km} - \mu_k) a_m, \tag{3}$$

Eigen decomposition of the energy weighted second-order moment tensor results in three eigenvalues, λ_j and three eigenvectors, $\mathbf{v_j}$. The values of λ_3 and $\mathbf{v_3}$ are key to the subsequent analysis. If the three eigenvalues are ordered as $\lambda_1 \geq \lambda_2 \gg \lambda_3$, we have a planar coherence anomaly, where the first and second eigenvectors $\mathbf{v_1}$ and $\mathbf{v_2}$ represent directions parallel to the planar anomaly.

In contrast, the third eigenvector $\mathbf{v_3}$ represents the direction perpendicular to the planar anomaly in the spherical analysis window. The eigenvectors $\mathbf{v_1}$, $\mathbf{v_2}$, and $\mathbf{v_3}$ have three components:

$$\begin{cases} \mathbf{v}_1 = \widehat{\mathbf{X_1}} v_{11} + \widehat{\mathbf{X_2}} v_{12} + \widehat{\mathbf{X_3}} v_{13} \\ \mathbf{v}_2 = \widehat{\mathbf{X_1}} v_{21} + \widehat{\mathbf{X_2}} v_{22} + \widehat{\mathbf{X_3}} v_{23}, \\ \mathbf{v}_3 = \widehat{\mathbf{X_1}} v_{31} + \widehat{\mathbf{X_2}} v_{32} + \widehat{\mathbf{X_3}} v_{33} \end{cases} \tag{4}$$

Where the unit vectors $\widehat{\mathbf{X_1}}$, $\widehat{\mathbf{X_2}}$, and $\widehat{\mathbf{X_3}}$ are oriented to North, East, and down. Machado et al. (2016) and Qi et al. (2017) show the fault dip azimuth attribute to be ATAN2(v_{31},v_{32}) and fault dip magnitude to be ACOS(v_{33}) .

Theory: Fault orientation using the second-order moment tensor
(After Qi et al., 2018)

Iterative directional smoothing and sharping

Laplacian and Gaussian operators are commonly used in filtering photographic images. The Laplacian of a Gaussian (LoG) operator smooths short-wavelength artifacts of images by the Gaussian operator prior to sharpening the images by the Laplacian operator. Taking into account the hypothesized fault orientation (the eigenvectors $\mathbf{v_1}$, $\mathbf{v_2}$, and $\mathbf{v_3}$), the Laplacian of a Gaussian operator can directionally smooth parallel to fault surfaces and sharpen perpendicular to fault surfaces. After the first process, the output fault probability will be input to the energy-weighted LoG filtering iteratively until the fault image is sufficiently smoothed and sharpened. The filter usually stabilizes after three iterations. Because we will wish to sharpen perpendicular to the hypothesized fault (along eigenvector v_3) and smooth parallel to the fault (along eigenvectors v_1 and v_2), we first rotate our natural (east, north, vertical) x-coordinate system, to a ξ-coordinate system aligned with the eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ axes:

$$\xi = \mathbf{Rx}, \tag{5}$$

where **R** is the rotation matrix given by:

$$\begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{pmatrix} = \begin{pmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \tag{6}$$

where v_{ij} are the directional components in equation 4. In our implementation, we want to smooth more along and less perpendicular to the faults, so we set:

$$\sigma_3^2 < \frac{1}{3}\sigma_1^2 = \frac{1}{3}\sigma_2^2, \tag{7}$$

where σ_3 is the larger of the two bin dimensions. Finally, we wish to modify the LoG operator to be directional: sharpening along the direction perpendicular to the planar discontinuity (along the ξ_3 axis):

$$\left(\frac{d^2}{d\xi_3^2}G\right)a = \sum_{m=1}^{M}\left(-\frac{1}{\sigma_3^2} + \frac{\xi_{3\mathrm{m}}^2}{\sigma_3^4}\right) exp\left[-\left(\frac{\xi_{1\mathrm{m}}^2}{2\sigma_1^2} + \frac{\xi_{2\mathrm{m}}^2}{2\sigma_2^2} + \frac{\xi_{3\mathrm{m}}^2}{2\sigma_3^2}\right)\right] a_m, \tag{8}$$

where G represents the Gaussian operator to be elongated along the planar axes. σ_1, σ_2, and σ_3 are standard deviation of the Gaussian operator. After the first iteration, the output fault probability will be used as input to the next iteration of energy-weighted LoG filtering until the fault image is sufficiently smoothed and sharped. It usually takes three to five iterations to obtain a reasonable result. As with structure-oriented filtering to improve the signal to noise ratio of the amplitude data (e.g. Fehmers and Höcher, 2003), iterative application of smaller windows provides both a computationally more efficient algorithm but also one that adapts to curved surfaces.

Example 1: Great South Basin (New Zealand dataset)

The following data from the Great South Basin is publically available and is provided courtesy of the New Zealand Petroleum and Minerals (NZP&M). The input to **fault_enhancement** is an energy ratio similarity volume. At present, the amount of sharpening is not significant, suggesting that we may wish to apply more aggressive filters or follow this process by skeletonization. Let's use **energy_ratio_similarity** as the input data volume. In the example below, I have used the default vertical window height in program **similarity3d** of ±5 samples (window_height=0.020 s for a sample increment Δt=0.004 s). Using very small vertical window may provide a smaller stair step artifact, but also increases spurious coherence anomalies which are later smoothed by the **fault_enhancement** algorithm. A good rule of thumb is to compute coherence using a window size that approximates the period of the highest frequency of interest.

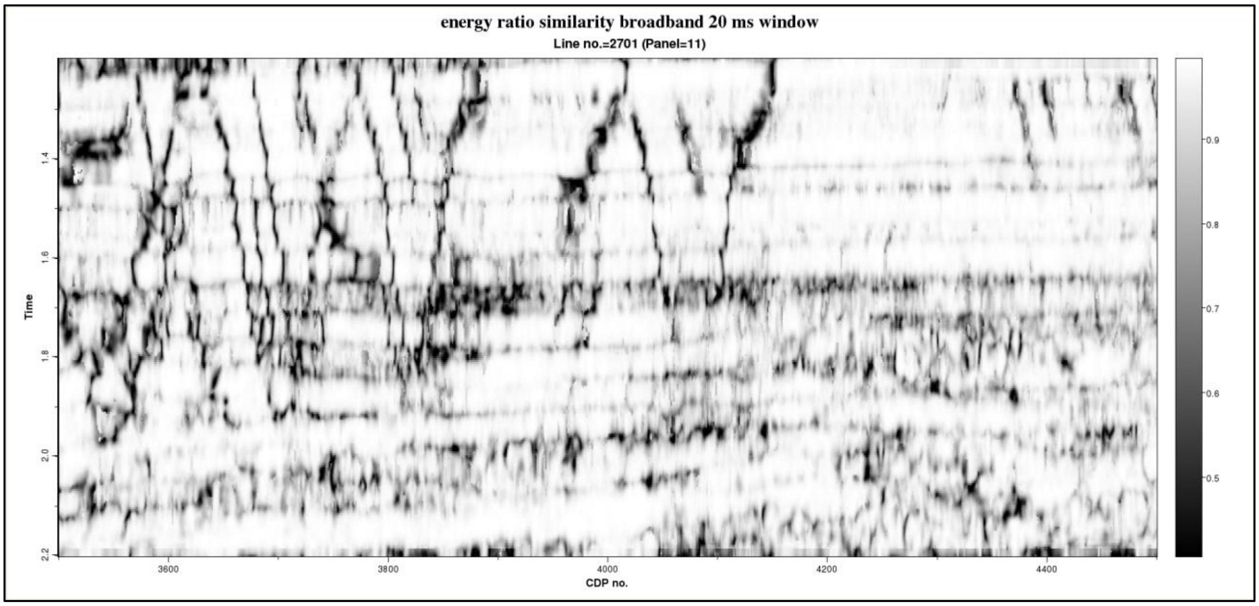


Figure 3.

The data are sampled on a 12.5 m by 25 m grid. For the first example, I do not use a weighting function and obtain the following image:

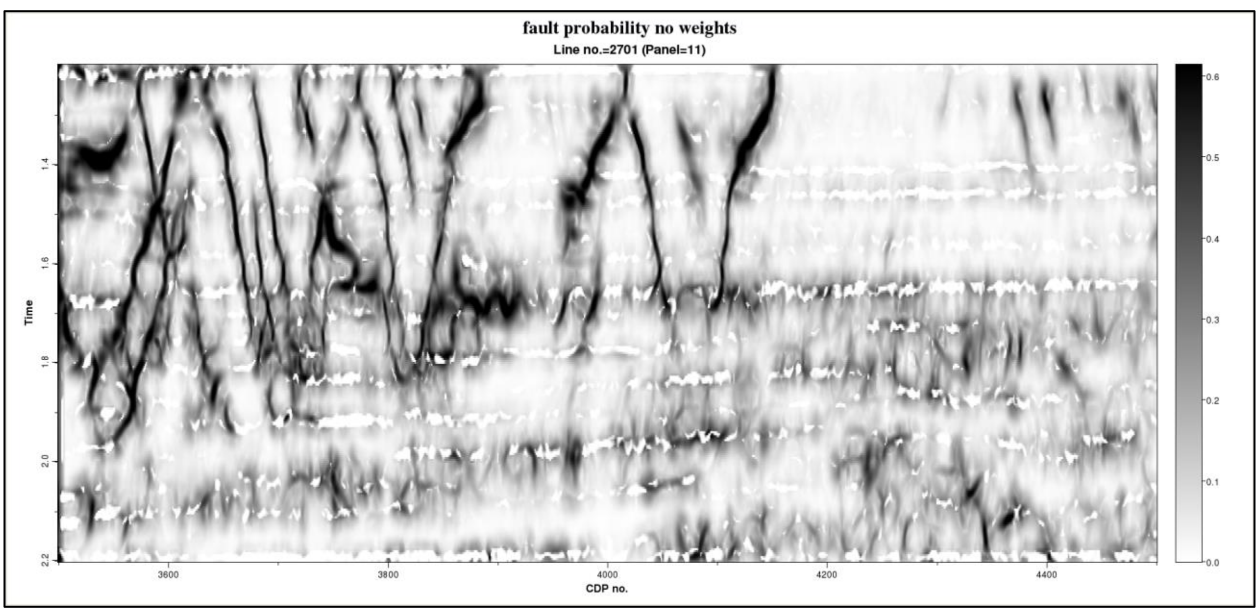


Figure 4.

Note some of the coherence anomalies aligned with stratigraphy have been suppressed (and appear as white), since I used values of ϑ_1=10^0 and ϑ_2=25^0.

Next, I use the total_energy attribute computed in program **similarity3d** as a weighting function in equation 1:

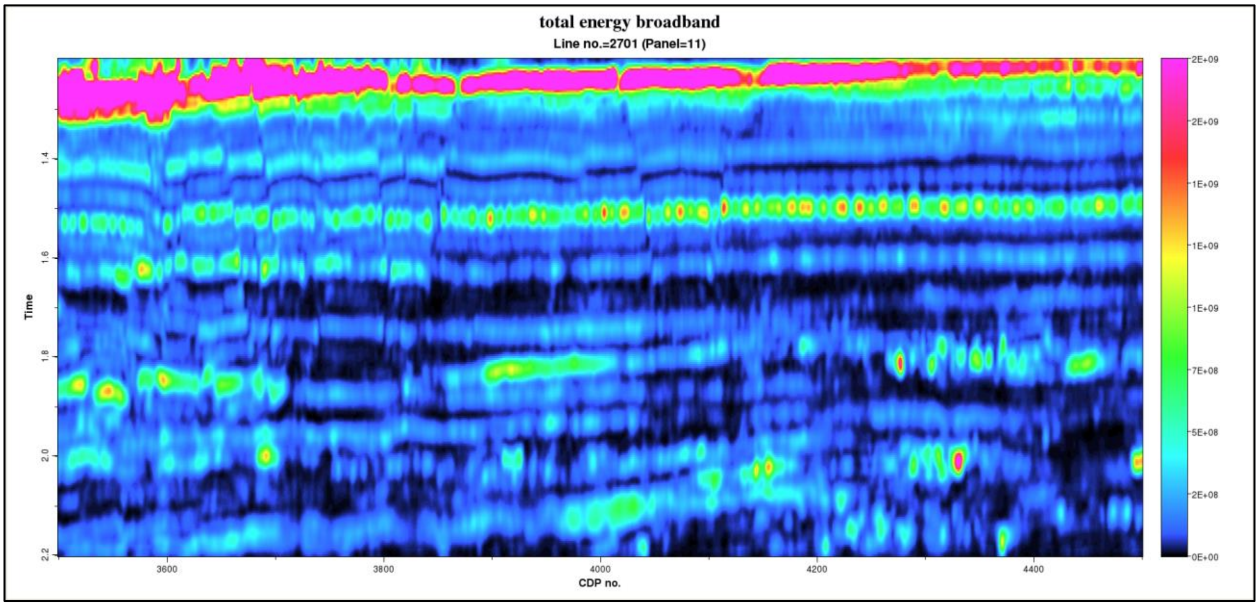


Figure 5.

Using these weights results in the following image.

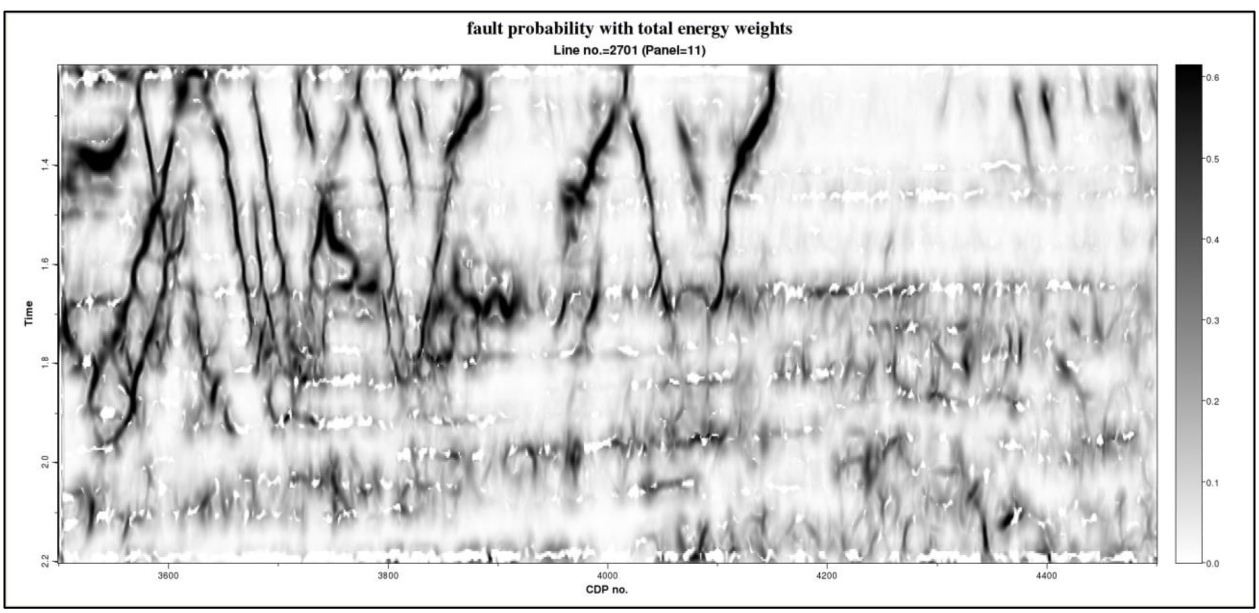


Figure 6.

The output fault probability can serve as input to a second pass of fault enhancement, resulting in an iterative workflow, and gives the following results. There is somewhat greater continuity of the steeply dipping faults, though they do not appear sharper or longer. Recall that faults dipping subparallel to this inline direction will appear to be smeared, though in reality, they are not. Following this workflow, I input the data fault probability computed from two iterations of fault enhancement to a third iteration and obtain the next image.

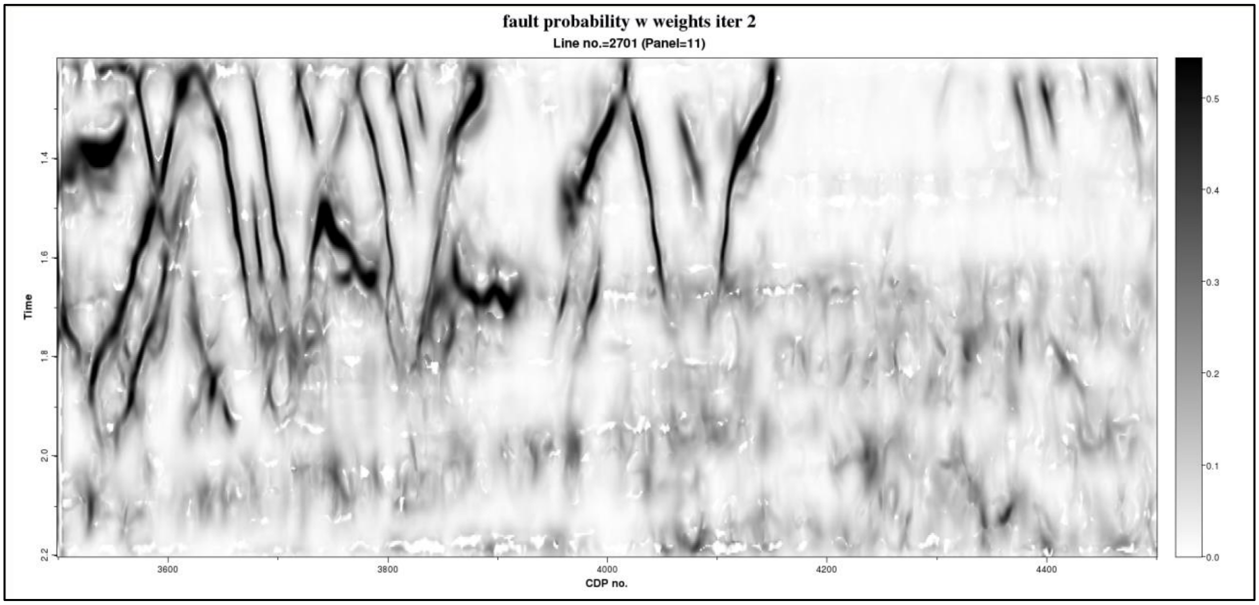


Figure 7.

A third iteration gives:

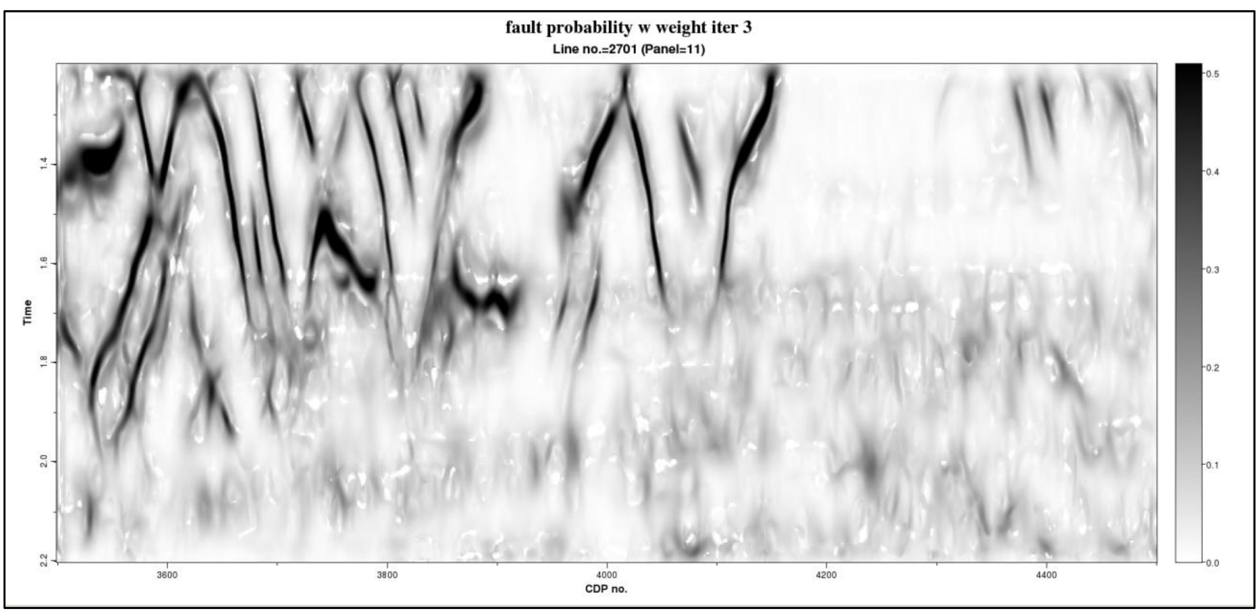
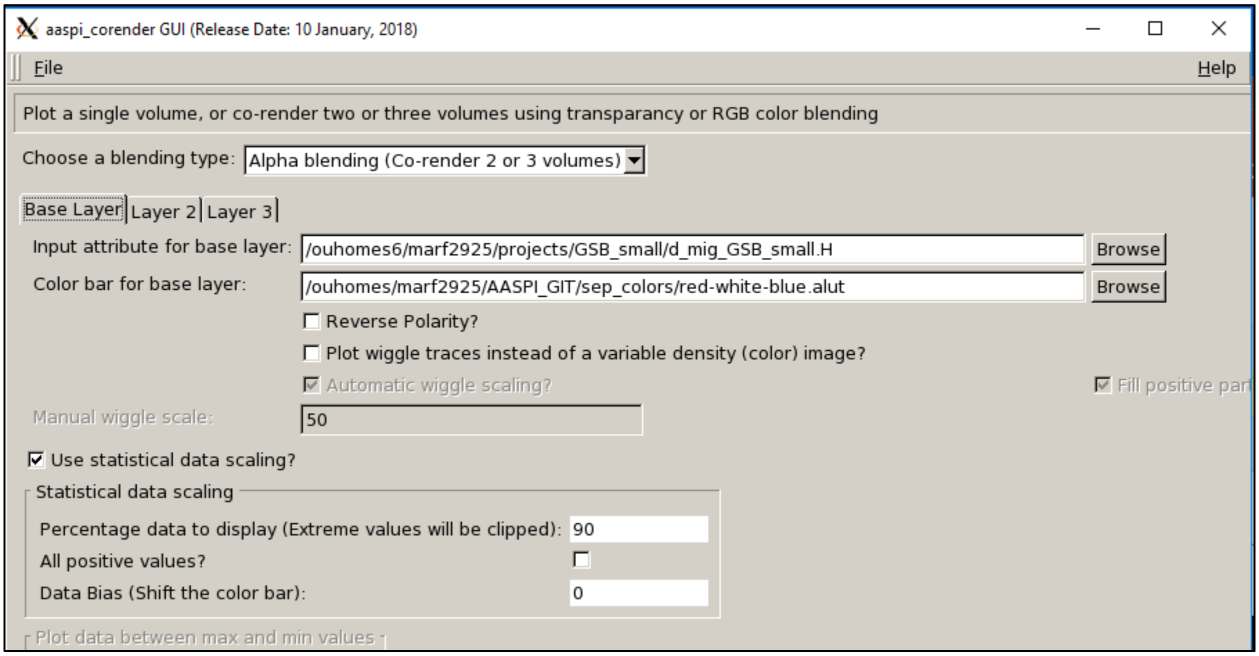
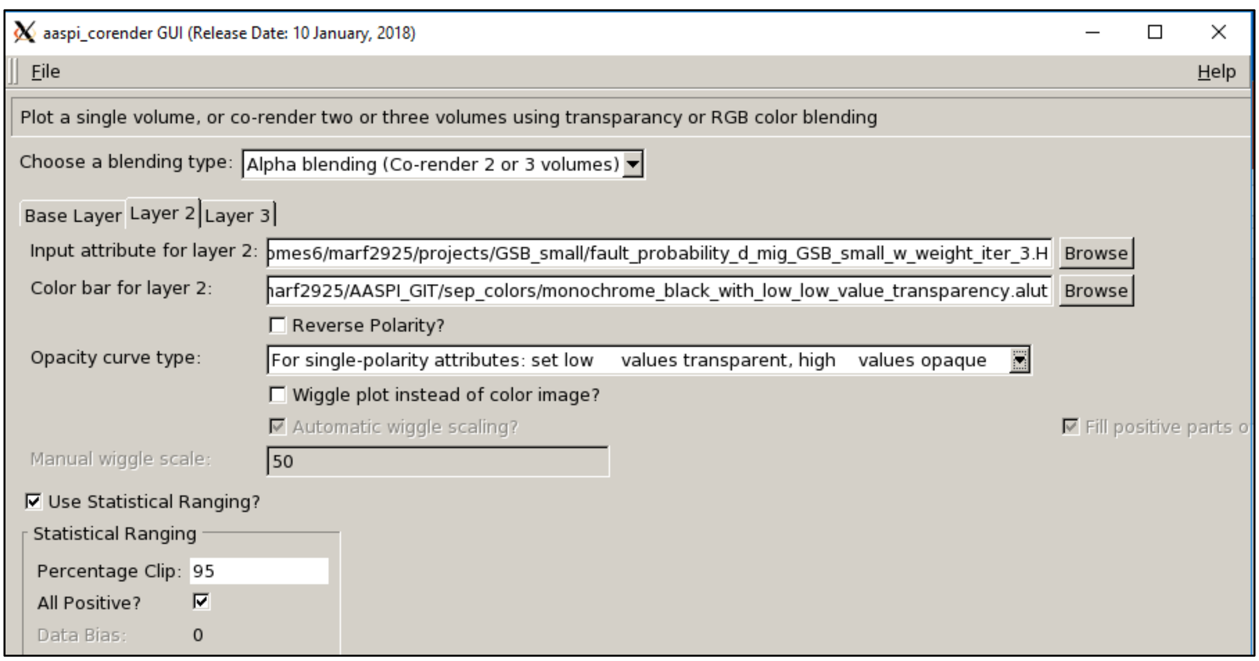


Figure 8.

Image Processing: Program **fault_enhancement**

Using AASPI program corender to display this image with the seismic amplitude gives (see next page):





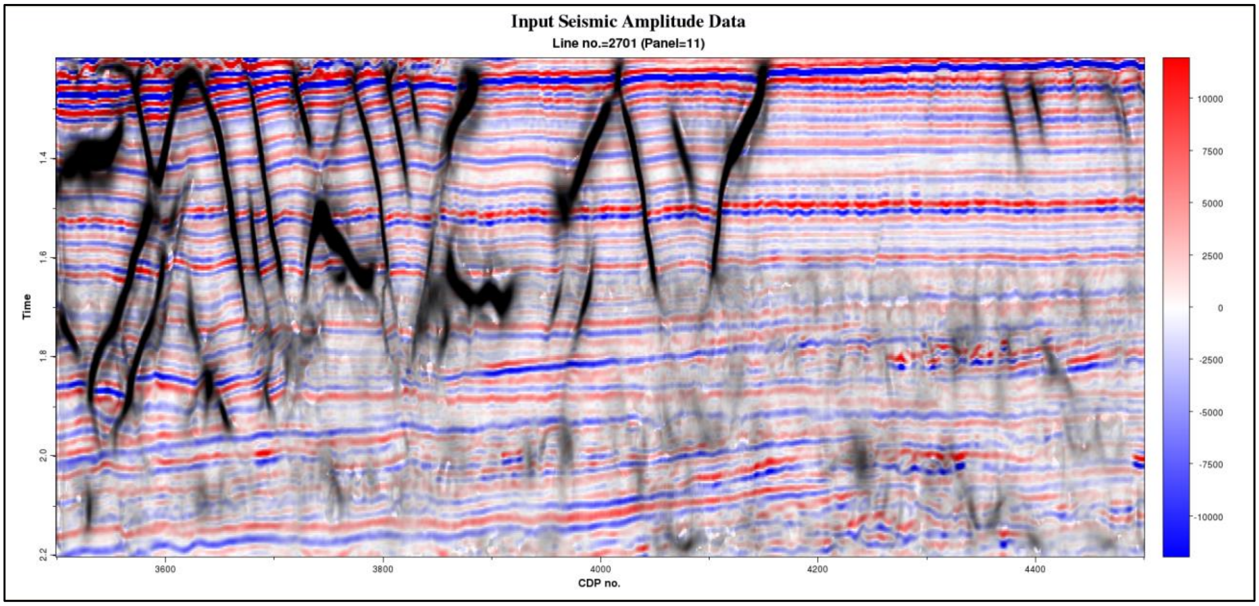


Figure 9.

Co-rendering the seismic amplitude with energy ration similarity computed using a +/-0.000 s window gives the following image:

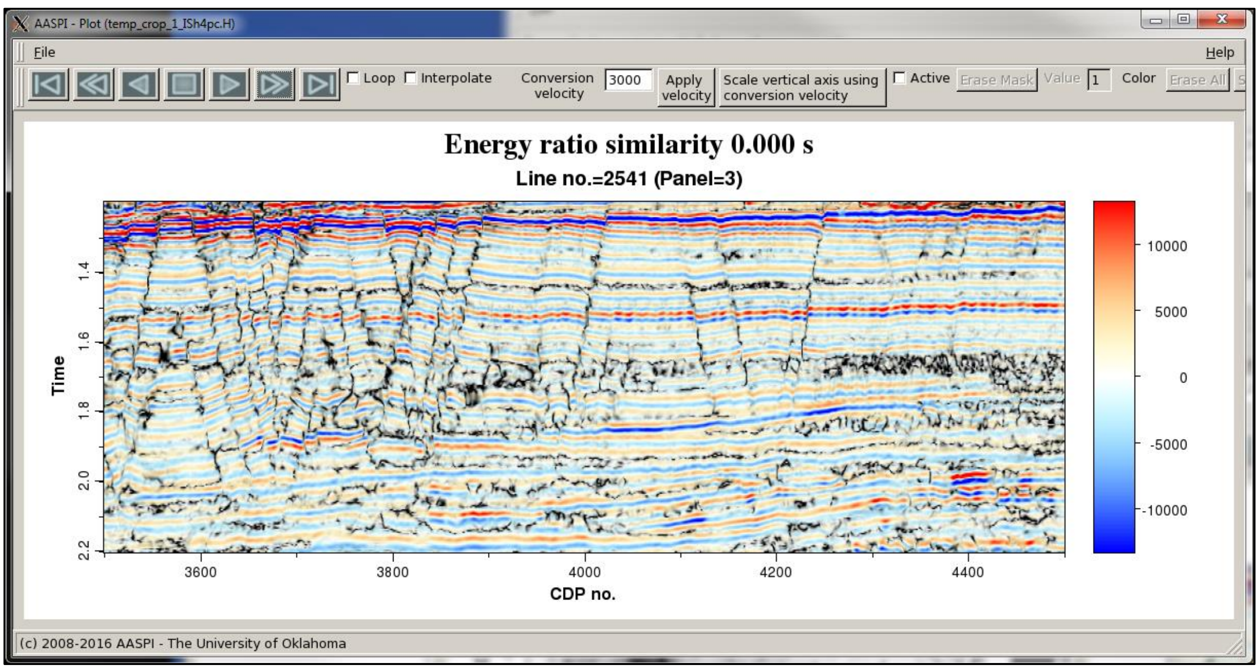


Figure 10.

while co-rendering the seismic amplitude with energy ratio similarity computed with a +/-0.020 s window gives (see next page) :

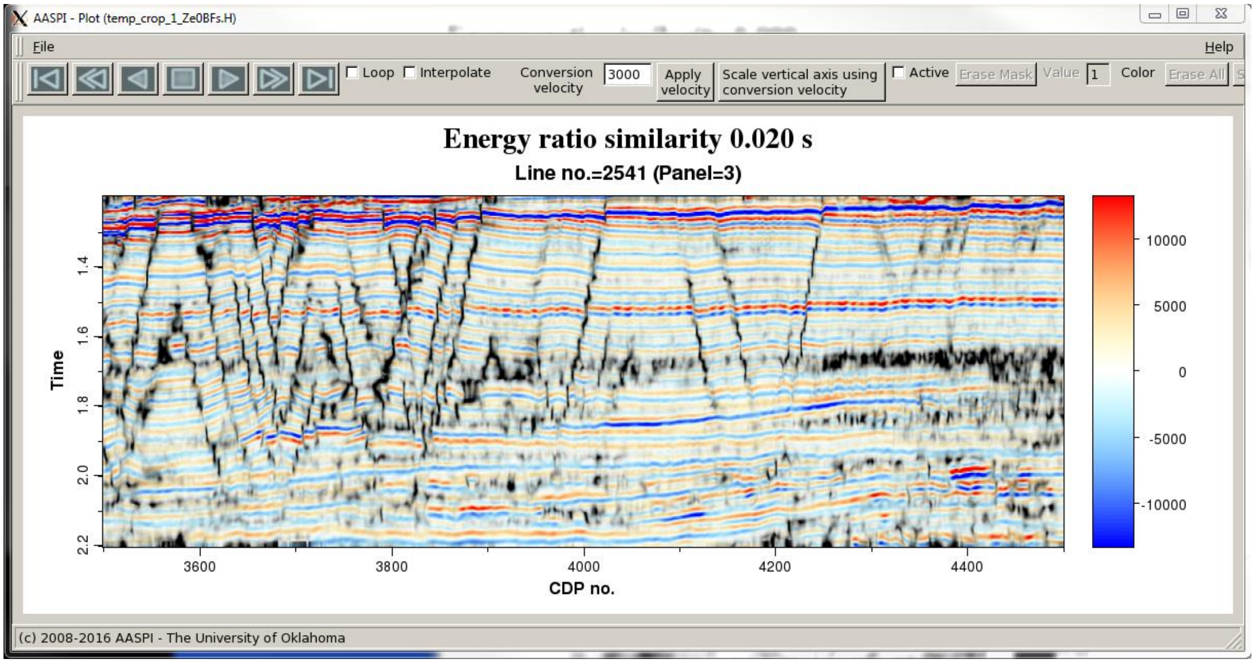


Figure 11.

Now, let's examine a representative time slice. First, let's look at the **energy_ratio_similarity** computed using a 0.020 ms (±5-sample) vertical analysis window.

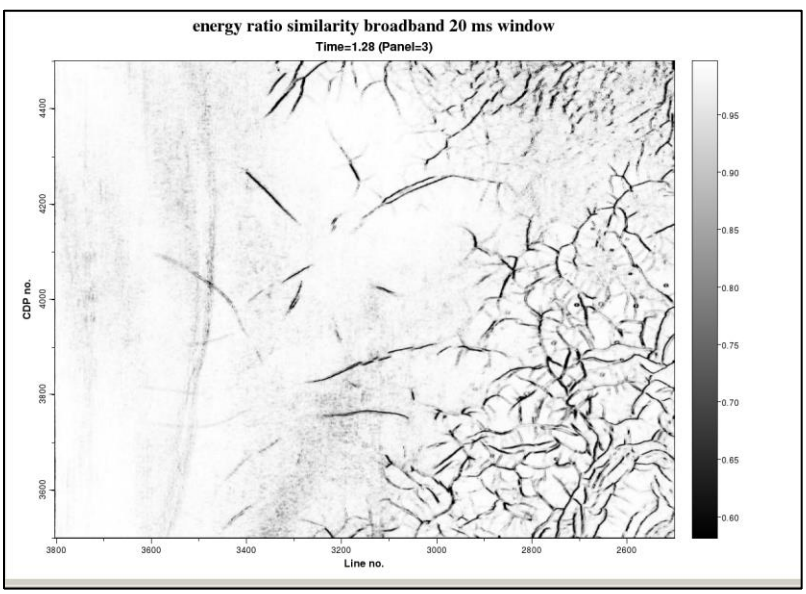


Figure 12.

The corresponding total energy volume looks like this:

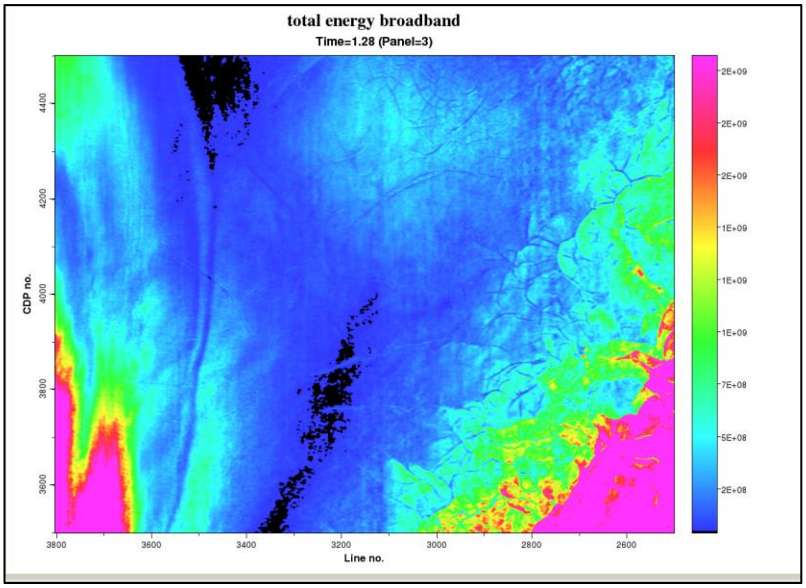


Figure 13.

Using these two volumes, the fault enhancement is as shown in the following image:

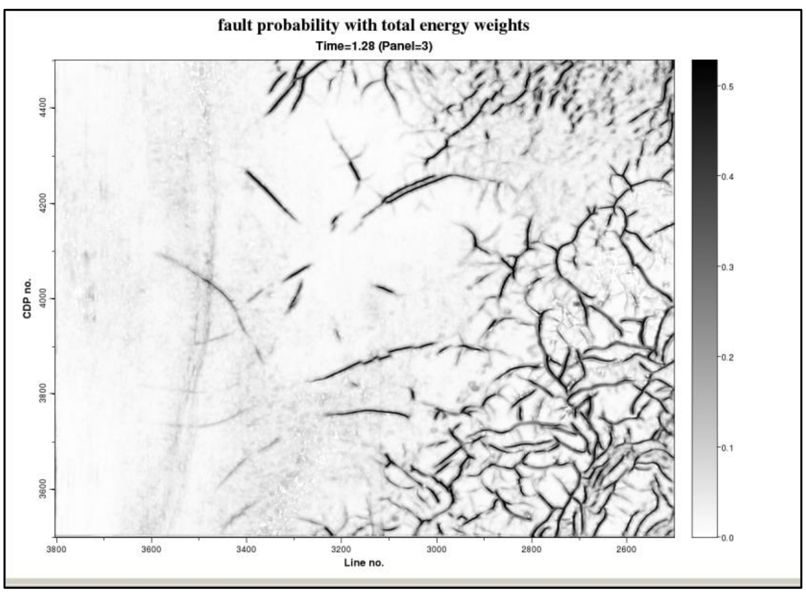


Figure 14.

After three iterations, I obtain:

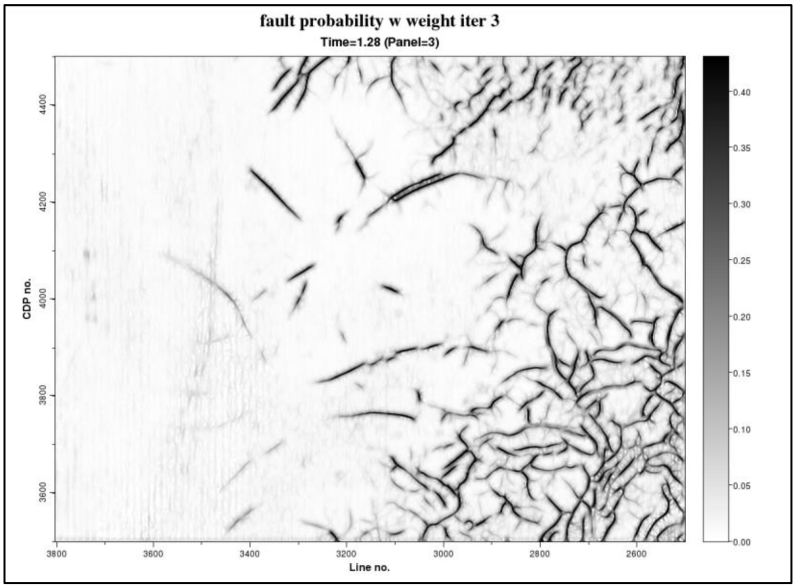


Figure 15.

The faults in these images may be "thicker" than desired. Using the above image (after three iterations of fault enhancement) I run program **skeletonize3d** and obtain:

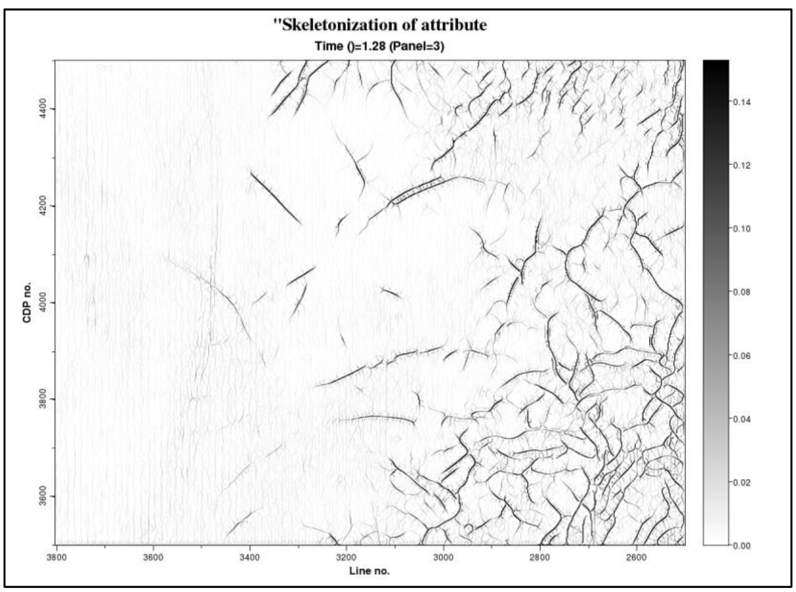
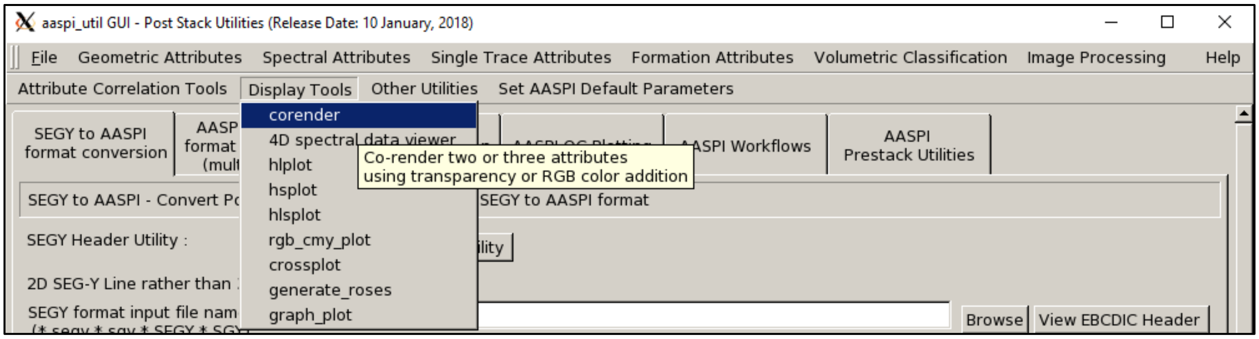
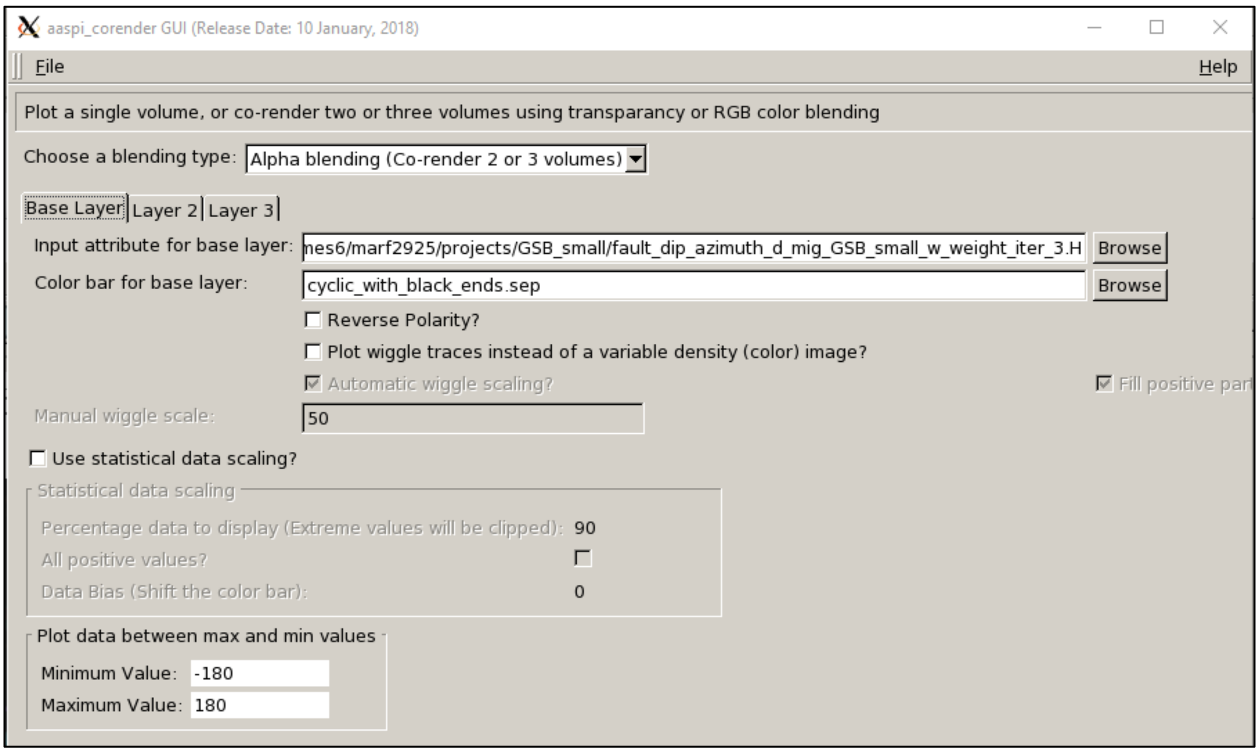


Figure 16.

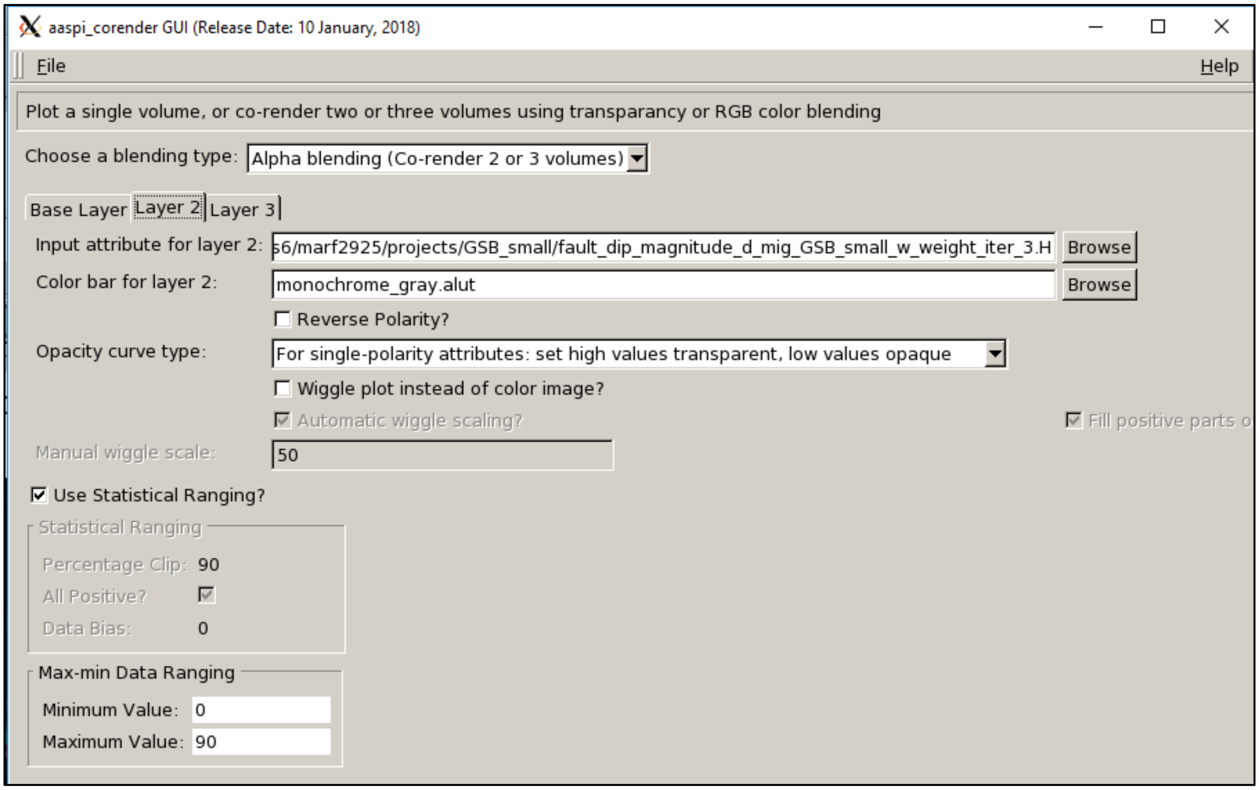
We can co-render the three fault attributes using program **corender** which is found under the *Display Utilities* tab on **aaspi_util**:



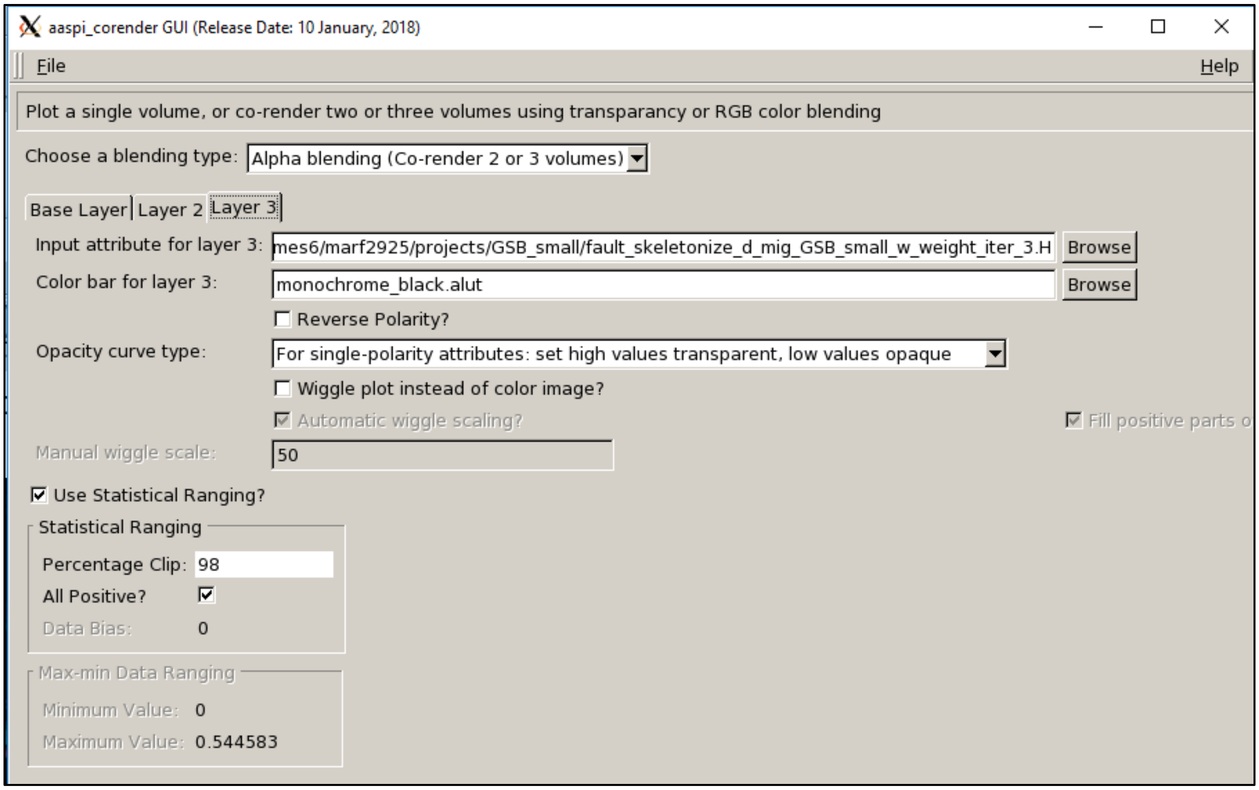
The first, or base layer will be the fault dip azimuth volume, which should be plotted against a cyclic color bar, with values ranging between -180° and +180°:



The second layer will be the fault dip magnitude volume, which should be plotted against a monochrome gray color bar, (with low values opaque and high values transparent) with values ranging between -90° and +90°:



The third and final layer will be either the fault probability or the skeletonized fault probability. It should be plotted against a monochrome black color bar (with low values opaque and large value transparent), using a *Statistical Data Ranging*:



The resulting vertical slices with and without skeletonization look like the following images:

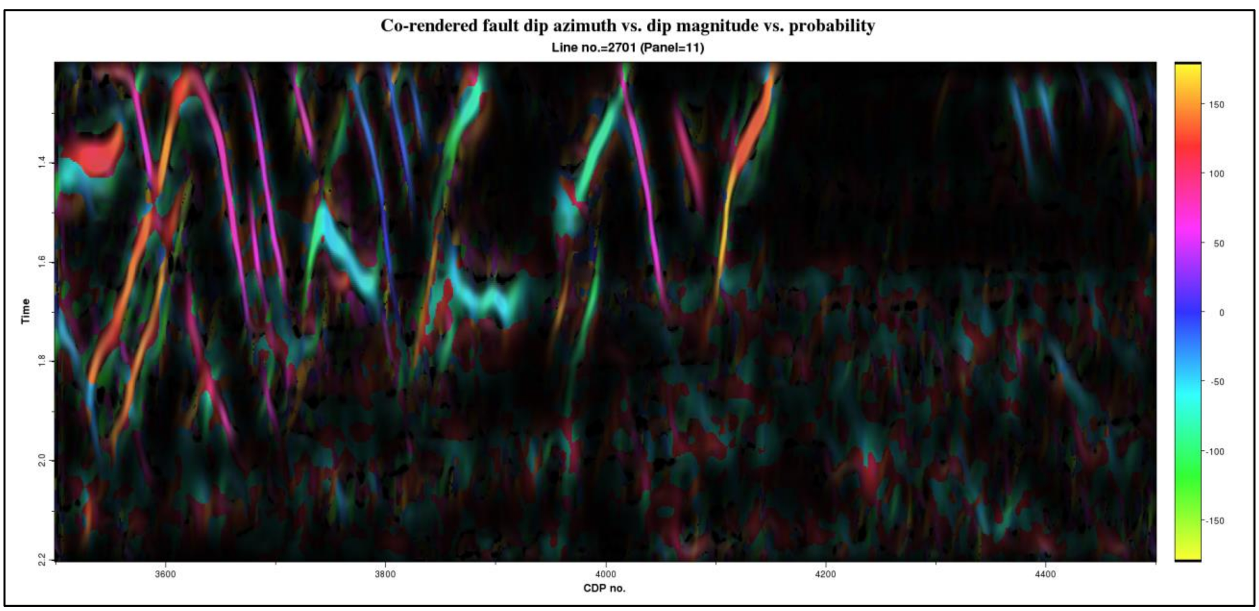


Figure 17.

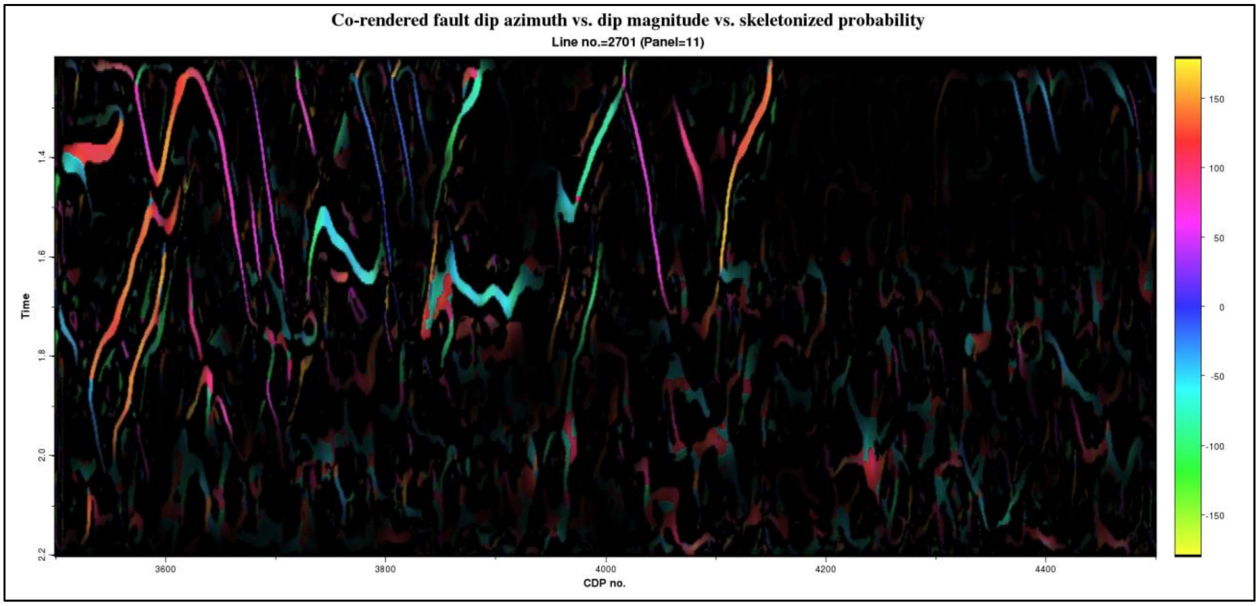


Figure 18.

Recall that a vertical slice that is subparallel to a fault will result in a broader image. The inline direction is approximately N30°E such that those faults whose azimuth is along that direction (striking perpendicular) appearing as magenta if dipping NNE at approximately 30° and as green if dipping SSW at approximately -150° are sharp, and those whose azimuth is perpendicular to the line (striking parallel) appearing as cyan if dipping NNW at approximately -60° and red if dipping SSE at approximately 120° appear blurred.

Time slices at t=1.28 s look the following images:

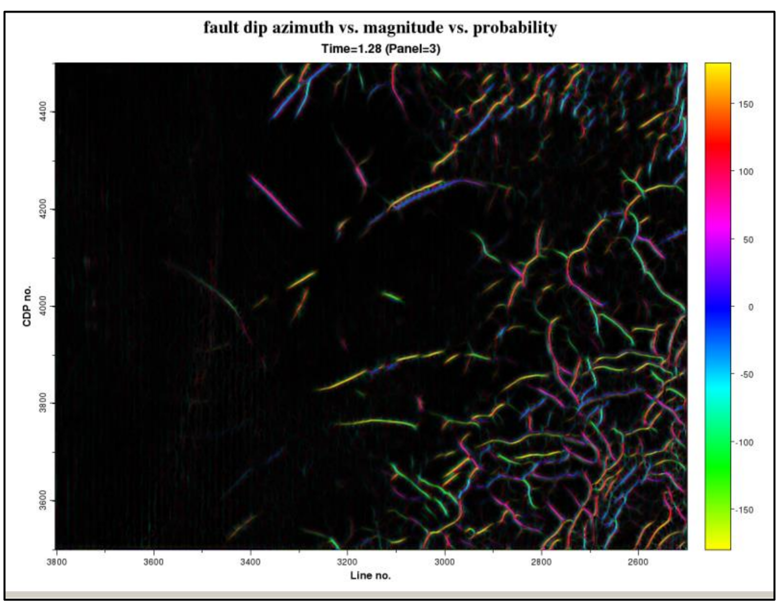


Figure 19.

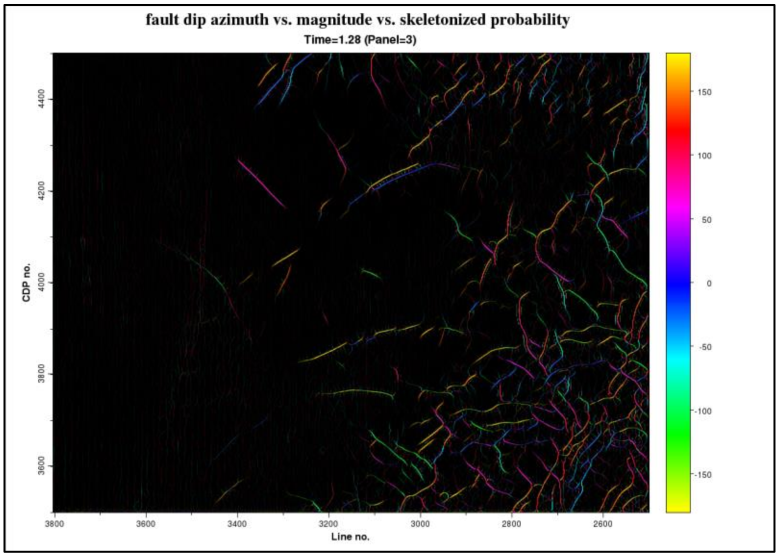


Figure 20.

Please recall that for this survey, the inline azimuth is at N30°.

Comparison to Petrel's *Ant Tracker* software

Randen et al. (2001) developed a swarm intelligence algorithm that both connects and skeletonizes the discrete discontinuities generated by the coherence family of attributes. Although the algorithm does not output fault dip azimuth and fault dip magnitude volumes it does allow the interpreter to limit the orientation of faults mapped by this algorithm through the clever use of a stereonet control panel. In this example, I will use multispectral coherence rather than variance as the input data volume. Coherence maps discontinuities to low values of coherence, *c*, and coherent reflectors to relatively high values, approaching c=1.0 . Petrel's variance can be shown to be mathematically equivalent to 1-*s*, where *s*=semblance. AASPI's version of semblance is the more general *outer product similarity* provided by program **similarity3d**, where the differences include computing the semblance of the analytic trace (the original data and its Hilbert transform), and if the variable window size option is chosen, to smoothly taper the edges of the spatial analysis window. This tapering requires computing semblance as the outer product, $s=\mathbf{u}^T\mathbf{C}\mathbf{u}/\text{Trace}(\mathbf{C})$, where **C** is the *J* by *J* tapered covariance matrix, Trace(**C**) indicates the sum of the matrix diagonals, and $\mathbf{u}=(1\ 1\ 1\ \ldots\ 1)/\text{SQRT}(J)$.

The time slice at t=1.280 s through the multispectral coherence volume looks like this:

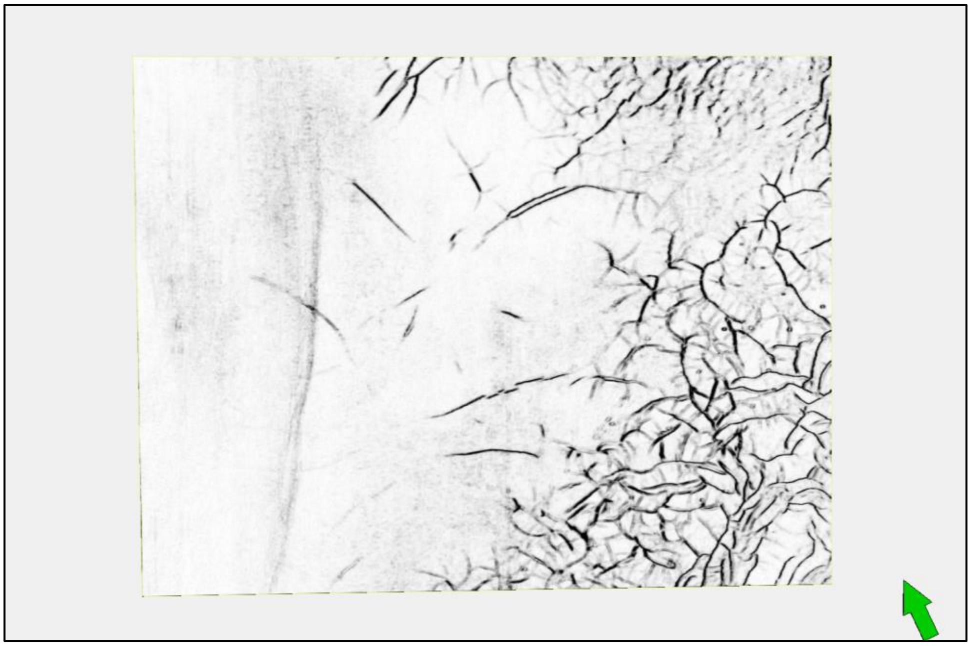


Figure 21.

To run the ant tracker, I first use the Petrel's attribute calculator to compute s=1-$c_{\mathrm{multispectral}}$. Then, using default parameters, the ant-tracker provides the following image:

Figure 22.

Corendering the two (with the ant tracker on top) gives the following image:

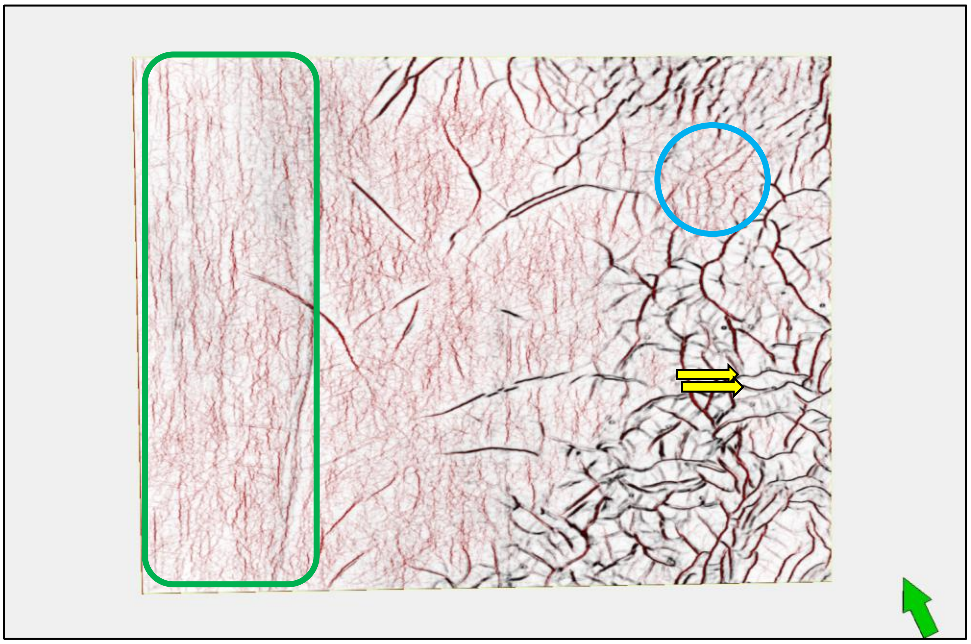


Figure 23.

There are several differences between the ant tracker and the input coherence volume. First, and perhaps most bothersome is that the amplitude and thickness of the ant track anomalies are poorly correlated to the amplitude and thickness of the original coherence anomalies. For example, the two faults indicated by the yellow arrows are well mapped by coherence, but give rise to a very small ant track anomaly. Indeed, the ant track anomalies not associated with faulting within the green box are of comparable or even greater. On a positive note, the ant tracker anomalies in the area indicated by the cyan circle are small faults whose offset falls below seismic resolution that are accurately mapped by curvature and aberrancy using AASPI program **curvature3d**.

Many of the other smaller faults identified by the ant tracker in the SE part of the eastern part of the survey are found in the original coherence volume by simply modifying the color bar in a nonlinear manner:

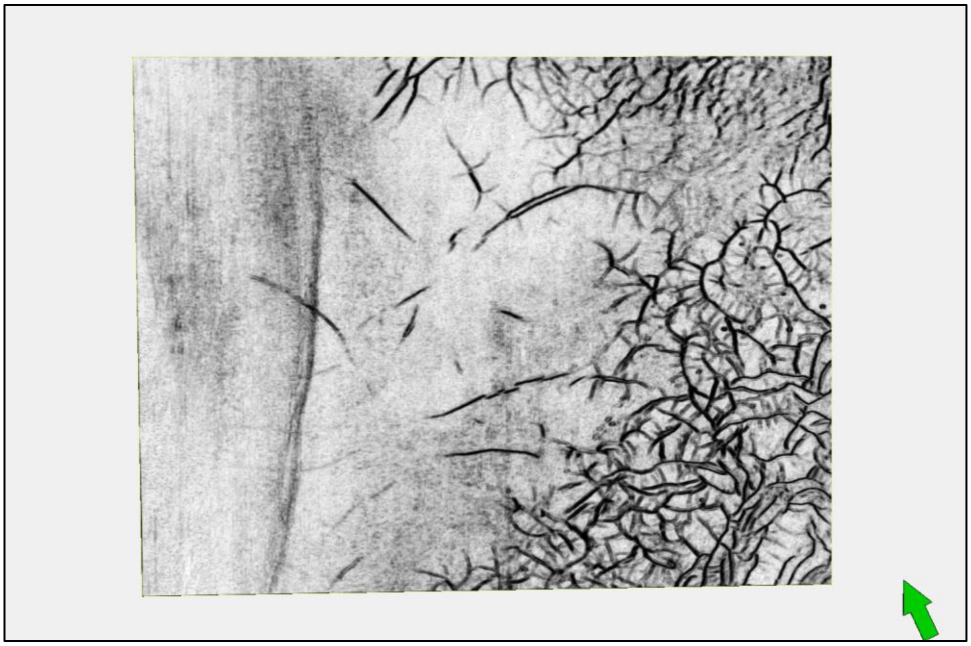


Figure 24.

Next, let's compare the same inline shown in a previous figure where I have corendered the multispectral coherence and the ant tracker output:

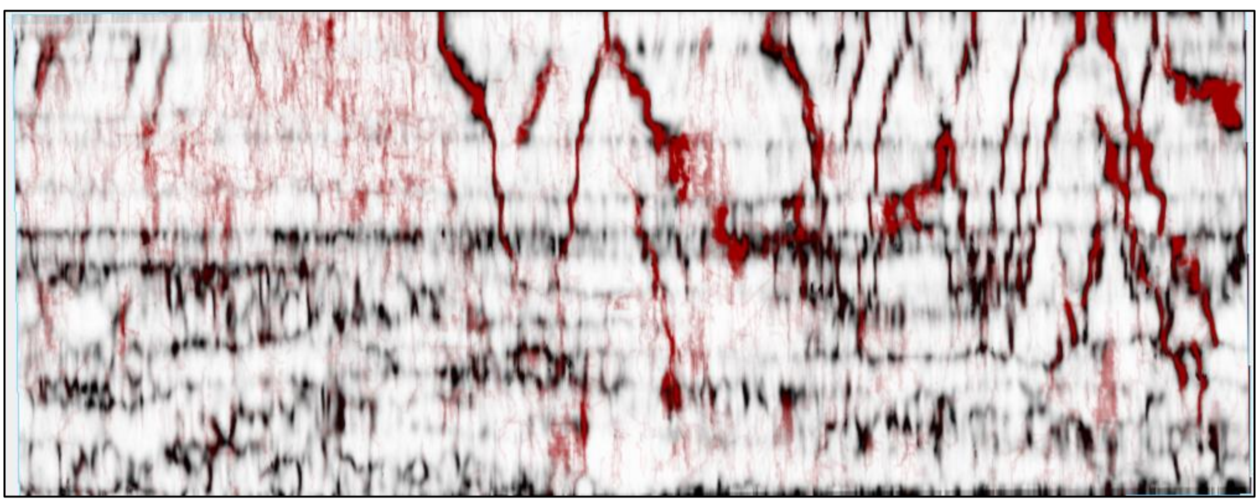


Figure 25.

Recall the stairstep artifacts generated by coherence when the reflector dip is not perpendicular to the dip of the fault plane. In this image, it appears that the ant tracker faithfully tracks the coherence artifacts, whereas the fault enhancement results are constrained to favor fitting a smooth surface through coherence anomalies that occur at the higher energy reflection points.

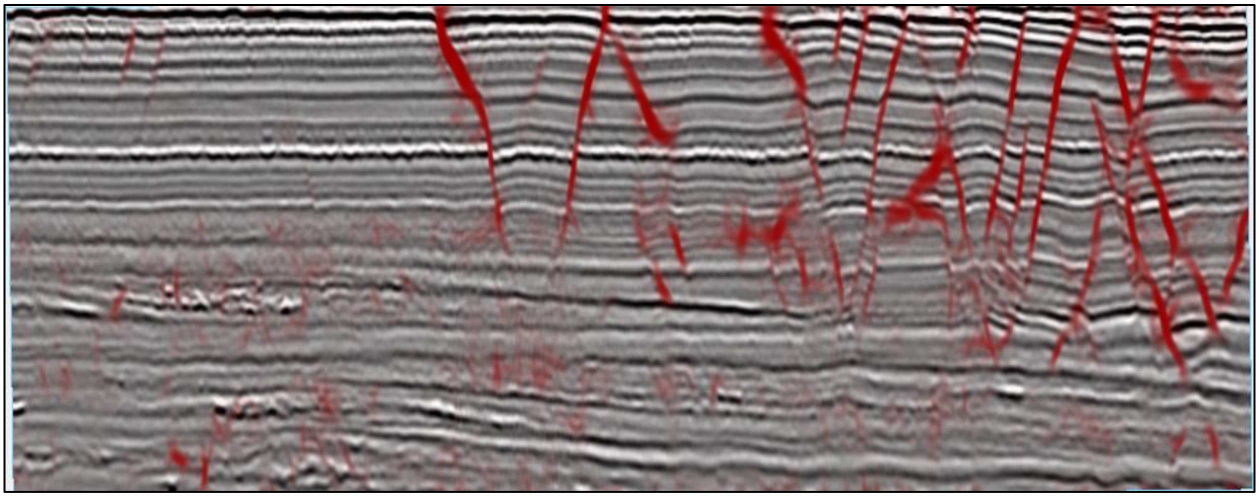
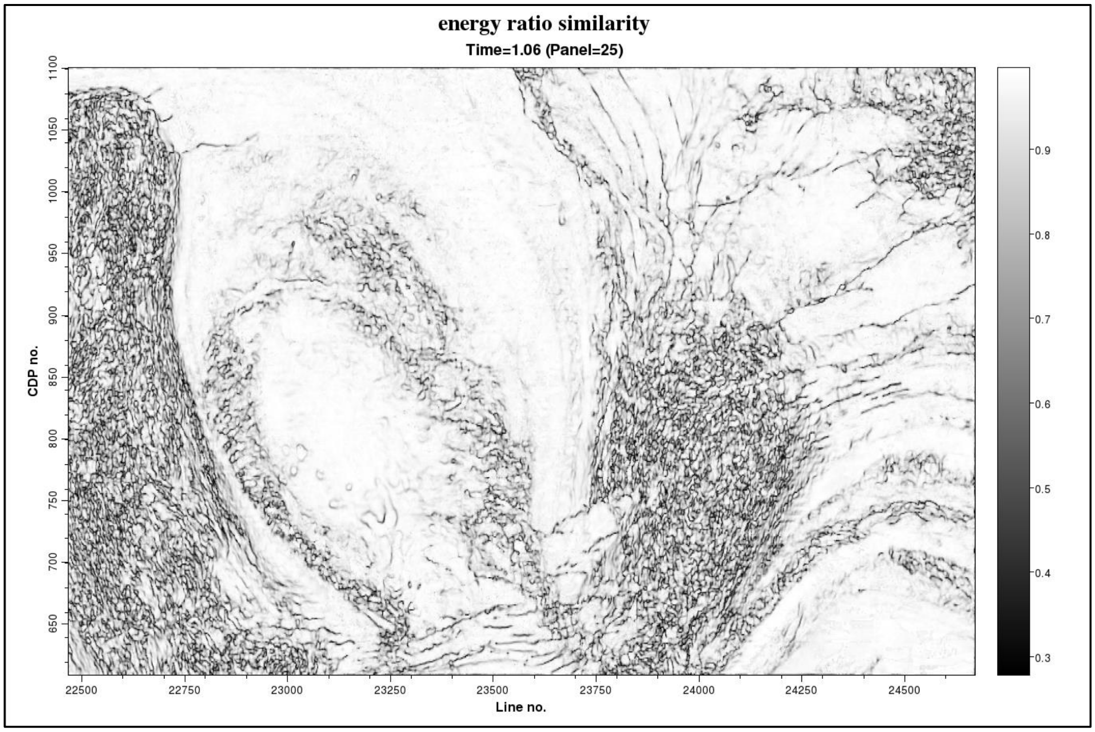
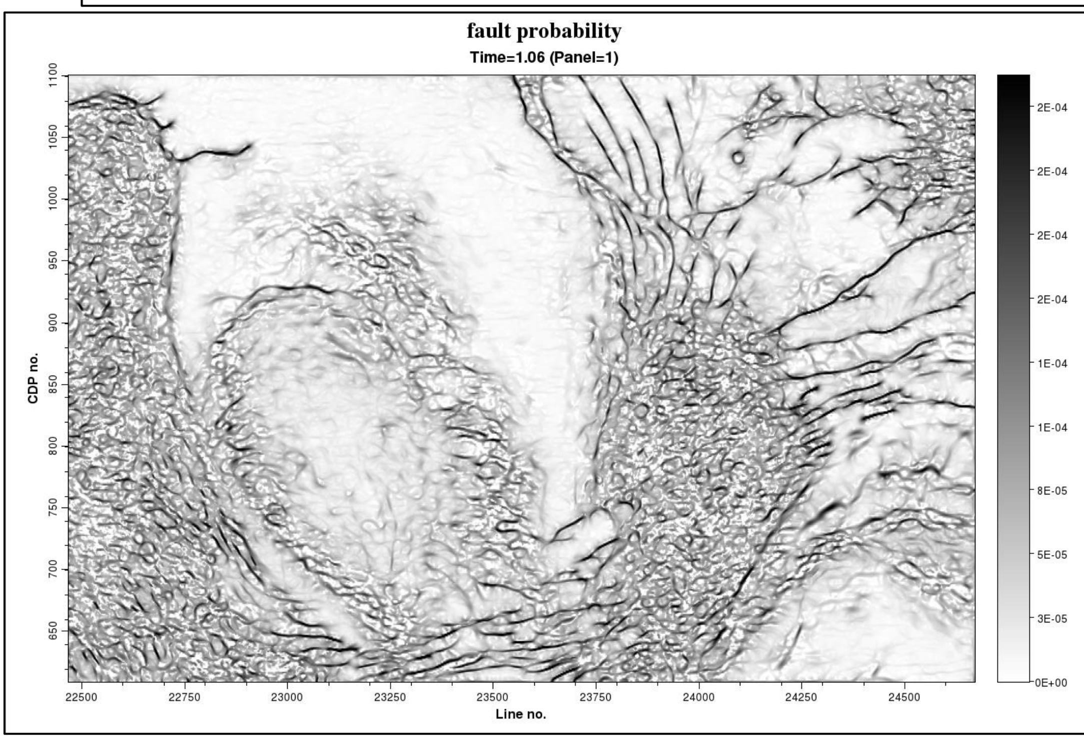


Figure 26.

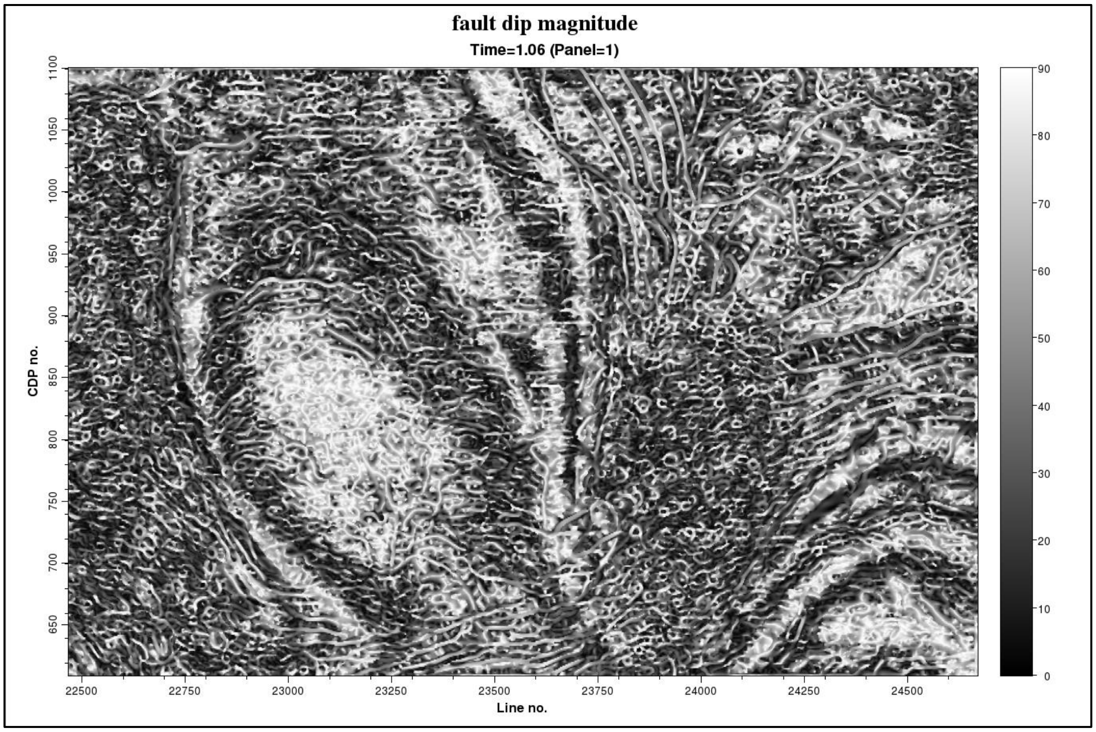
Example 2

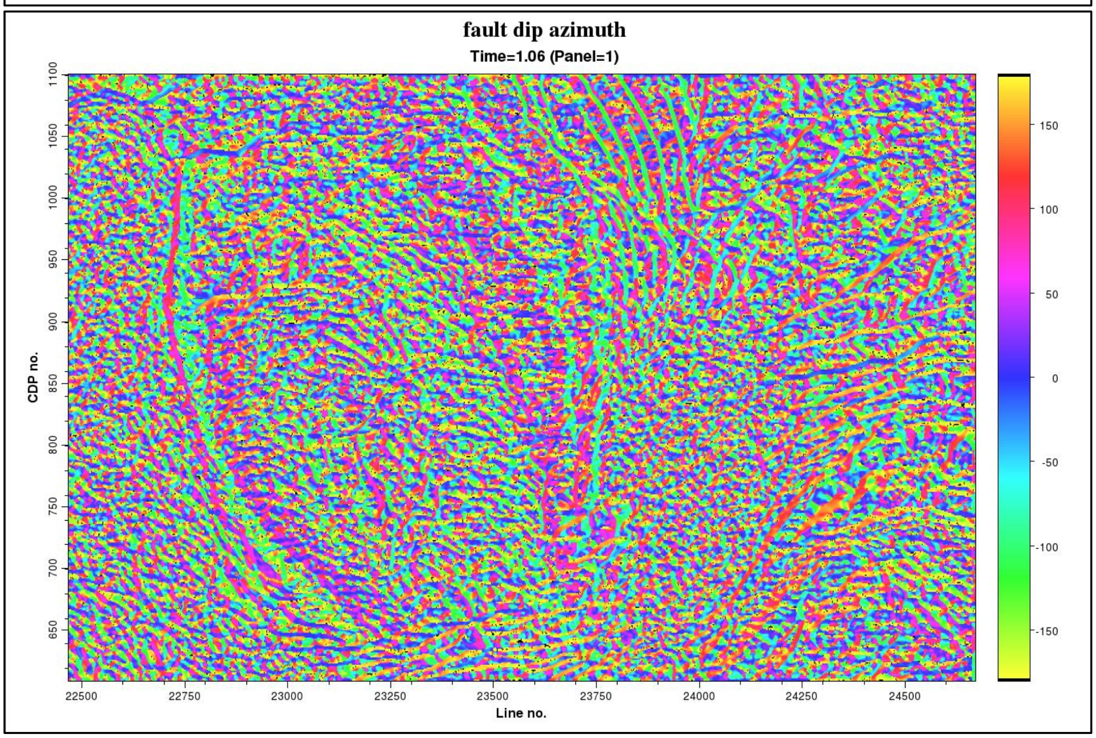
The following examples come from a different survey in the Gulf of Mexico and are discussed by Qi et al. (2017). The dataset within the inline and crossline spacing of 123.1ft * 39.4ft (37.5m * 12.5m), covers over 2723.27 ft^2 (253 km^2), and has been pre-stack time migrated. The data has been preconditioned through the structure-oriented filtering. In this case, the energy ratio similarity was computed from the 3rd iterated pc-filtered seismic amplitude data using a +/-0.020 s (+/- 5 sample) window.





Figures 27 (a) and (b)



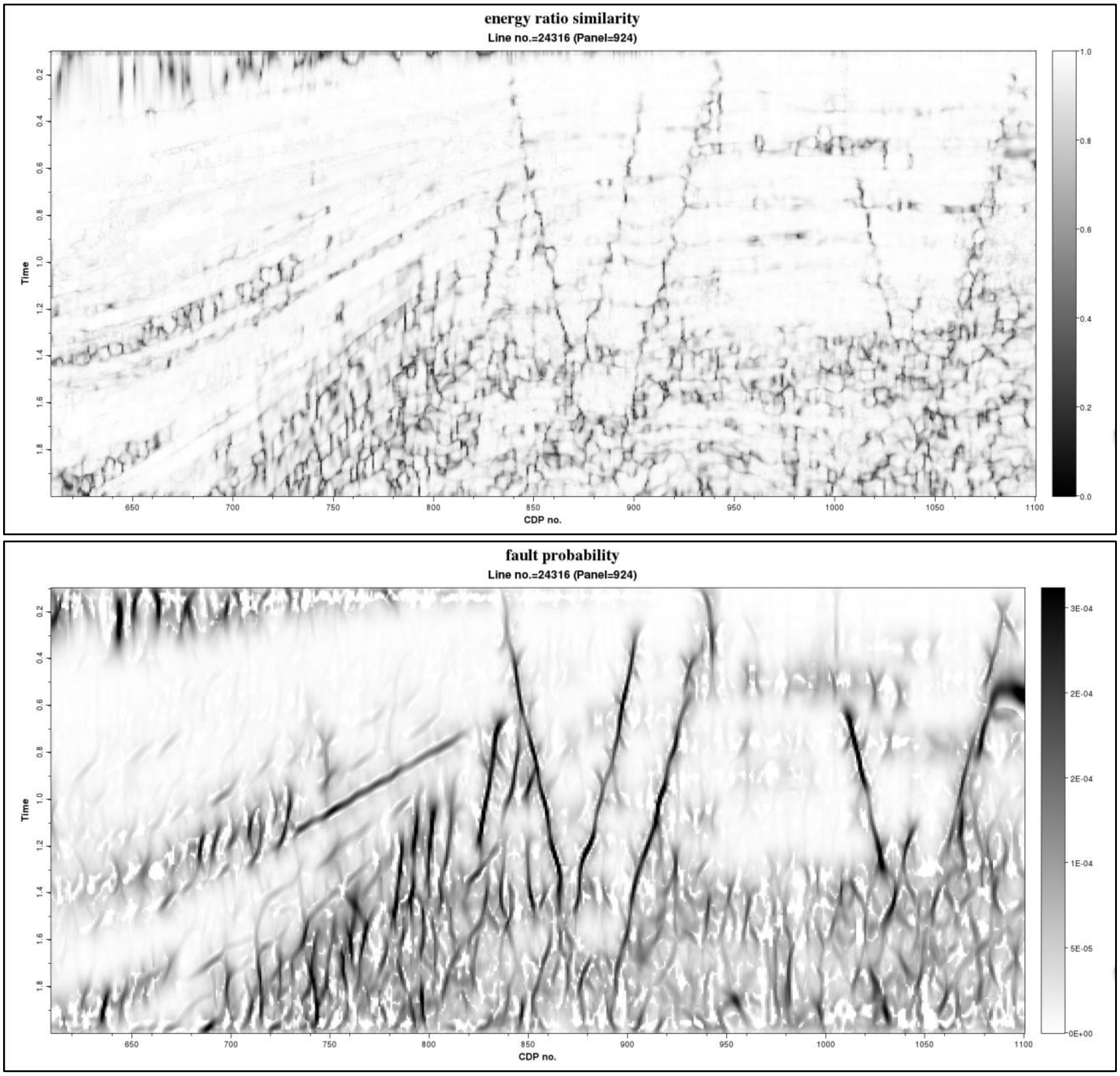


Figures 28 (a) and (b).

Fault anomalies are now more continuous, exhibit higher contrast, with reduced “stairstep” artifacts. Salt edges, MTC edges, and many subtle faults are enhanced in both time and vertical slices

Image Processing: Program **fault_enhancement**

Examples through vertical slices:



Figures 29 (a) and (b).

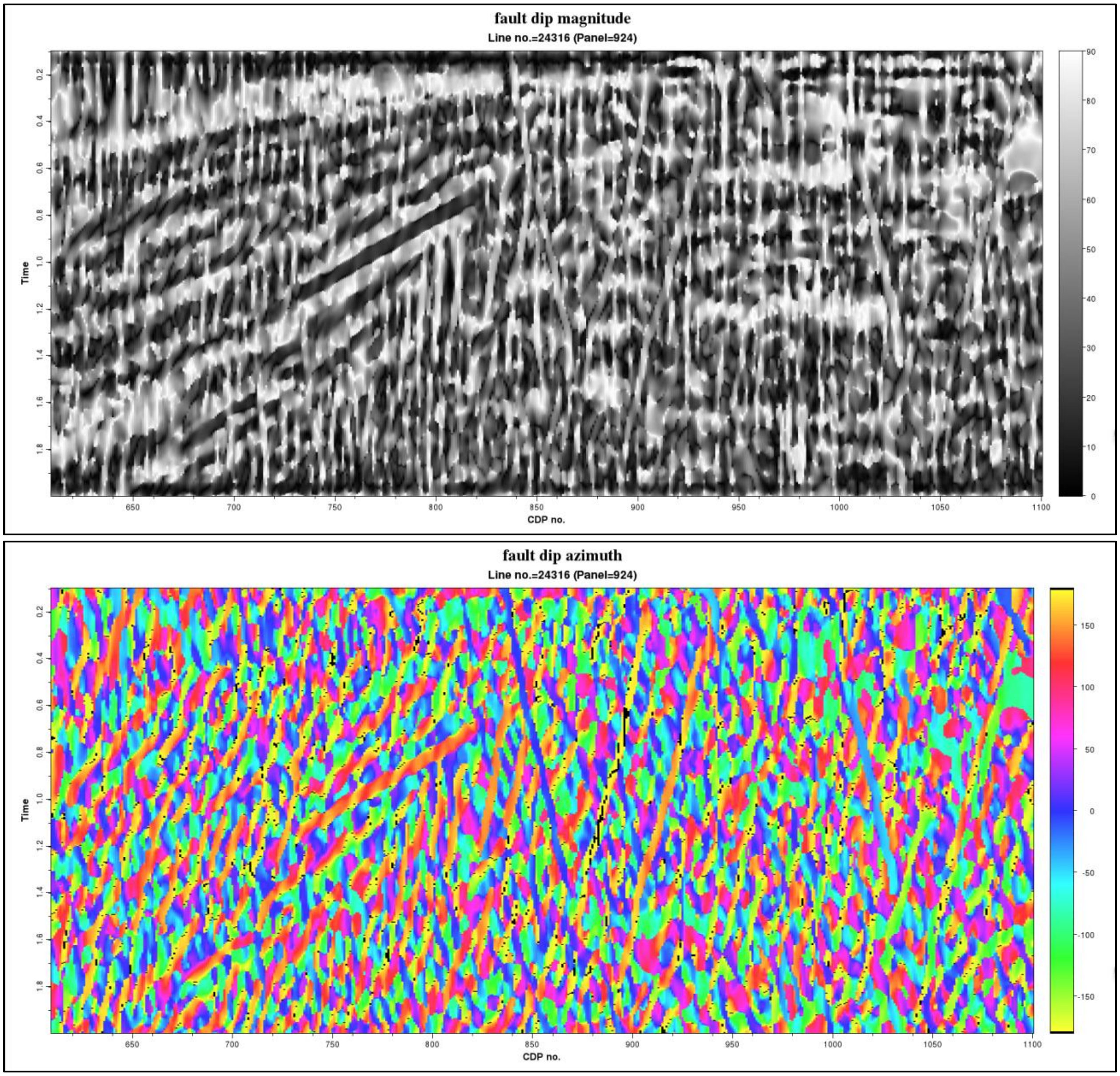
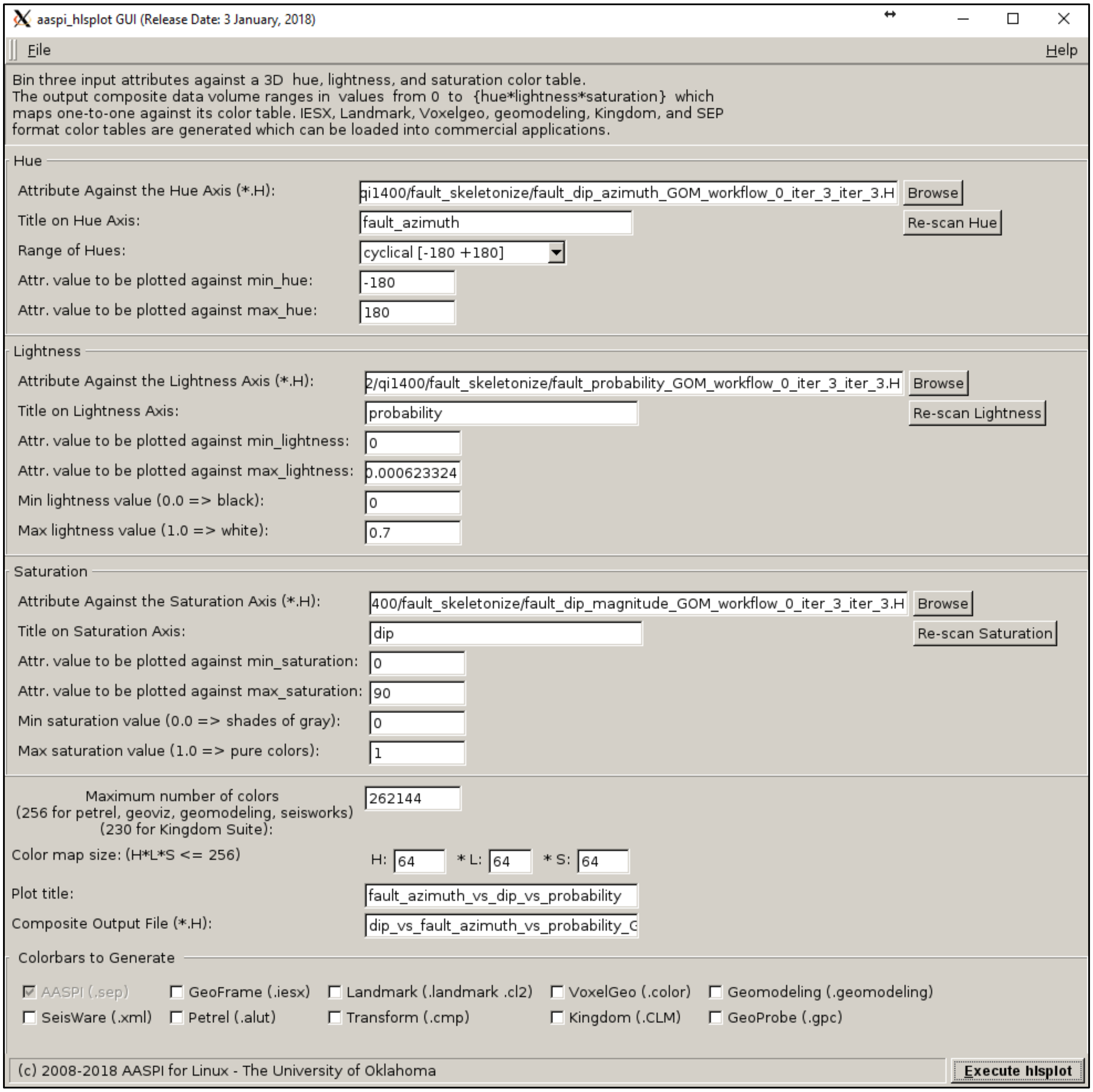


Figure 30 (a) and (b).

Finally, let's use AASPI program **hlsplot** to co-render the fault probability, fault dip magnitude, and fault dip azimuth. Hue-Lightness-Saturation (HLS) color model is used to co-render the fault dip magnitude (against S), the skeletonized fault probability (against L), and the fault dip azimuth (against H). In the figures below, after running the program, the fault orientation is readily seen. Numerical computation of fault probability and orientation at each voxel provide an easy way to identify fault sets, either visibly or through statistical analysis. Note that the coherent noise within the salt has been organized and should be interpreted as noise.


aaspi_hlsplot GUI (Release Date: 3 January, 2018)
File
Help
Bin three input attributes against a 3D hue, lightness, and saturation color table.
The output composite data volume ranges in values from 0 to {hue*lightness*saturation} which
maps one-to-one against its color table. IESX, Landmark, Voxelgeo, geomodeling, Kingdom, and SEP
format color tables are generated which can be loaded into commercial applications.
Hue
Attribute Against the Hue Axis (*.H):
qi1400/fault_skeletonize/fault_dip_azimuth_GOM_workflow_0_iter_3_iter_3.H
Browse
Title on Hue Axis:
fault_azimuth
Re-scan Hue
Range of Hues:
cyclical [-180 +180]
Attr. value to be plotted against min_hue:
-180
Attr. value to be plotted against max_hue:
180
Lightness
Attribute Against the Lightness Axis (*.H):
2/qi1400/fault_skeletonize/fault_probability_GOM_workflow_0_iter_3_iter_3.H
Browse
Title on Lightness Axis:
probability
Re-scan Lightness
Attr. value to be plotted against min_lightness:
0
Attr. value to be plotted against max_lightness:
0.000623324
Min lightness value (0.0 => black):
0
Max lightness value (1.0 => white):
0.7
Saturation
Attribute Against the Saturation Axis (*.H):
400/fault_skeletonize/fault_dip_magnitude_GOM_workflow_0_iter_3_iter_3.H
Browse
Title on Saturation Axis:
dip
Re-scan Saturation
Attr. value to be plotted against min_saturation:
0
Attr. value to be plotted against max_saturation:
90
Min saturation value (0.0 => shades of gray):
0
Max saturation value (1.0 => pure colors):
1
Maximum number of colors
(256 for petrel, geoviz, geomodeling, seisworks)
(230 for Kingdom Suite):
262144
Color map size: (H*L*S <= 256)
H: 64 * L: 64 * S: 64
Plot title:
fault_azimuth_vs_dip_vs_probability
Composite Output File (*.H):
dip_vs_fault_azimuth_vs_probability_G
Colorbars to Generate
AASPI (.sep)
GeoFrame (.iesx)
Landmark (.landmark .cl2)
VoxelGeo (.color)
Geomodeling (.geomodeling)
SeisWare (.xml)
Petrel (.alut)
Transform (.cmp)
Kingdom (.CLM)
GeoProbe (.gpc)
(c) 2008-2018 AASPI for Linux - The University of Oklahoma
Execute hlsplot

The time slice at t=1.06 s appears as follows

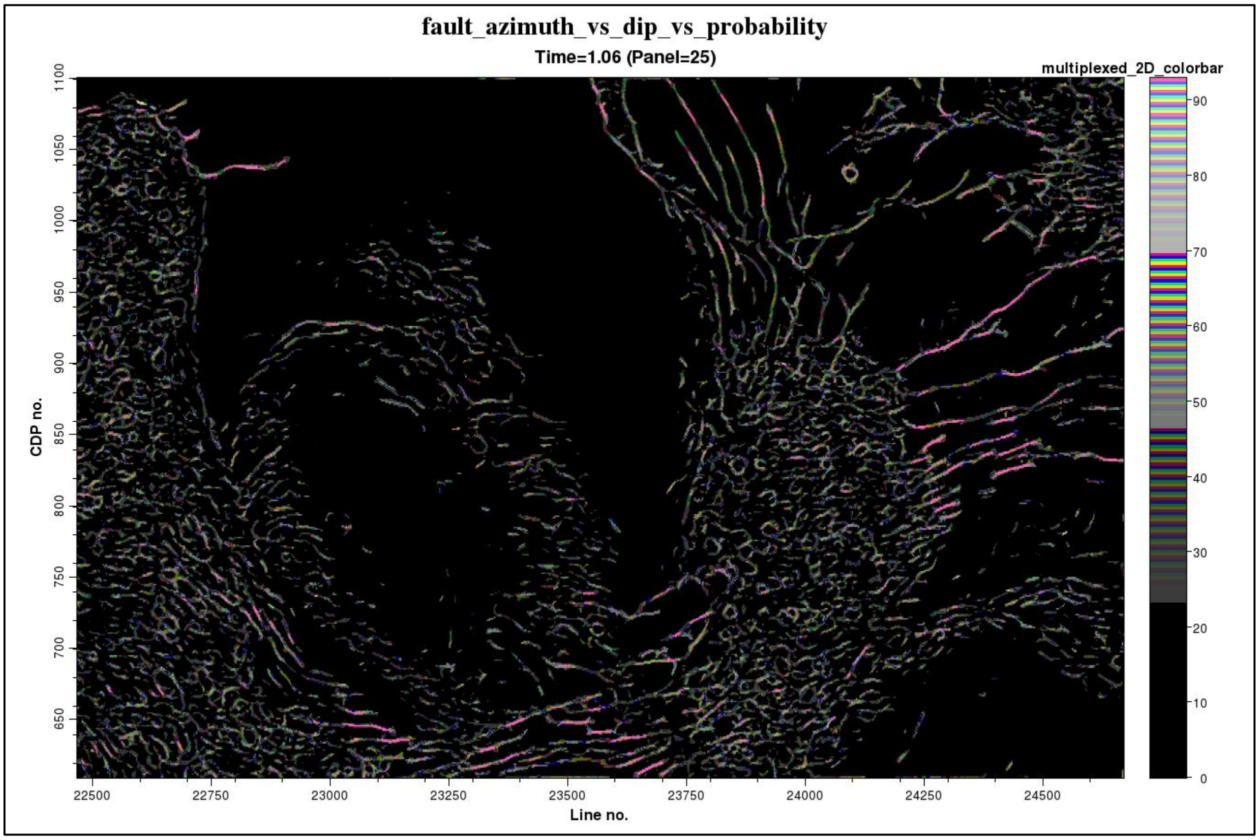


Figure 31.

A vertical slice through HLS example:

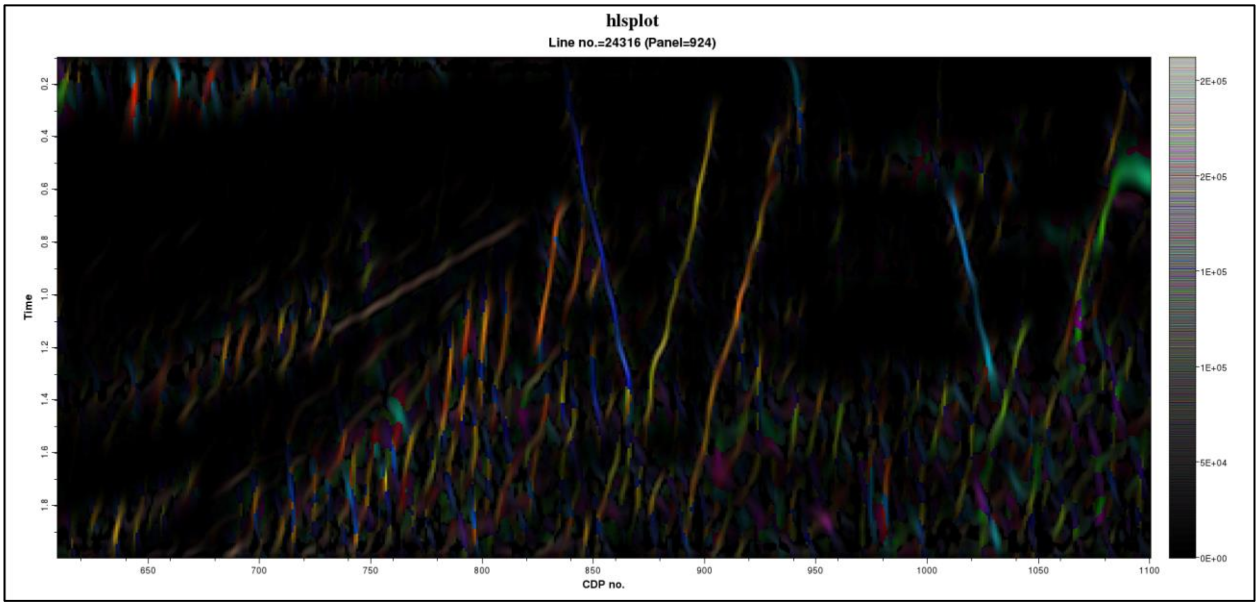


Figure 32.

References

Barnes, A. E., 2006, A filter to improve seismic discontinuity data for fault interpretation: Geophysics, **71**, P1-P4.

Dorn, G. and B. Kadlec, 2011, Automatic Fault Extraction in Hard and Soft -Rock Environments: The 31stAnnual Bob F. Perkins Research Conference31stAnnual Bob F. Perkins Research Conference on Seismic attributes –New views on seismic imaging: Their use in exploration and production, 587-619.

Machado, G., A. Alali, B. Hutchinson, O. Olorunsola, and K. J. Marfurt, 2016, Display and enhancement of volumetric fault images, Interpretation, **4**, xx-yy.

Millán M. S., and E. Valencia, 2006, Color image sharpening inspired by human vision models: Applied Optics, 45, Issue 29, 7684-7697 doi.org/10.1364/AO.45.007684

Qi, J., B. Lyu, A. Alali, G. Machado, Y. Hu, and K. J. Marfurt, 2019, Image processing of seismic attributes for automatic fault extraction: Geophysics, **84**, O25-O37. DCI 10.1190/GEO2018-0369.1

Qi, J., G. Machado, and K. J. Marfurt, 2017, A workflow to skeletonize faults and stratigraphic features, Geophysics, **82**, no. 4, O57-O70.

Qi, J., B. Lyu, X. Wu, and K. J. Marfurt, 2021, Comparing convolutional neural networking and image processing seismic fault detection workflows: Interpretation, **8**, in press.

Randen, T., S. I. Pedersen, and L. Sonneland, 2001, Automatic extraction of fault surfaces from three-dimensional seismic data: 71st Annual International Meeting of the SEG, Expanded Abstracts.