GEOPHYSICS, VOL. 78, NO. 2 (MARCH-APRIL 2013); P. O33–O43, 10 FIGS.
10.1190/GEO2012-0331.1

Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images

Dave Hale[1]

ABSTRACT

Fault interpretation enhances our understanding of complex geologic structures and stratigraphy apparent in 3D seismic images. Common steps in this interpretation include image processing to highlight faults, the construction of fault surfaces, and estimation of fault throws. Although all three of these steps have been automated to some extent by others, fault interpretation today typically requires significant manual effort, suggesting that further improvements in automatic methods are feasible and worthwhile. I first used an efficient algorithm to compute images of fault likelihoods, strikes, and dips from a 3D seismic image. From these three fault images, I then automatically extracted fault surfaces as meshes of quadrilaterals that coincide with ridges of fault likelihood. A quadrilateral mesh is a simple data structure alongside which one can easily gather samples of the 3D seismic image. I automatically estimated fault throws by minimizing differences in values of samples gathered from opposite sides of a fault, while constraining the variation of throw within a fault surface. I tested the fidelity of estimated fault throws by using them to undo faulting. After unfaulting, reflectors in 3D seismic images were more continuous than those in the original 3D seismic image. In one example, this unfaulting test supported the observation that some extracted fault surfaces have unusual conical shapes.

INTRODUCTION

Fault surfaces are an important aspect of subsurface geology that we can derive from seismic images. Fault displacements are important as well because they enable correlation across faults of subsurface properties.

In the context of exploration geophysics, fault throw, relative displacement up or down along the dip of a fault, is usually more significant than displacement along the strike of a fault. Moreover, fault throw vectors are usually more perpendicular to geologic layers, and therefore easier to estimate, than are strike-slip displacements.

Figure 1 illustrates the definition of fault throw as a vector component of displacement that is perpendicular to the strike of a fault surface. The orientation of that surface is described by a fault strike angle ϕ and fault dip angle θ, where the latter is here measured from vertical. Typically, fault orientation and fault throw vary within a fault surface.

Figure 2 shows examples of curviplanar fault surfaces and corresponding fault throws computed automatically from a 3D seismic image using new methods proposed in this paper. Because fault throws are vectors, they are difficult to visualize in displays of numerous faults. Therefore, while estimating all three components of fault throw vectors, I display in color only the vertical components. Because the vertical axis of the seismic image is time (not depth), vertical components of fault throw are here measured in milliseconds. Note that fault throws are nonnegative, which indicates that the faults shown here are normal faults.

As Luo and Hale (2012) describe, we can use estimated fault throw vectors to undo faulting. Figure 3 displays in color vertical components of fault throws on slices of a 3D seismic image, before and after this unfaulting process. Although only vertical components of fault throw are displayed, the unfaulting process uses all three components of fault throw vectors. After unfaulting, seismic reflections are more continuous across faults, suggesting that estimated fault throws are generally consistent with true fault displacements.

Fault surfaces and fault throws like those displayed in Figures 2 and 3 may be computed by numerous automatic or semiautomatic methods used today in the interpretation of 3D seismic images. Therefore, before describing the new methods proposed in this paper, I briefly review methods developed by others.

Manuscript received by the Editor 16 August 2012; revised manuscript received 24 November 2012; published online 25 February 2013.
[1]Colorado School of Mines, Golden, Colorado, USA. E-mail: dhale@mines.edu.

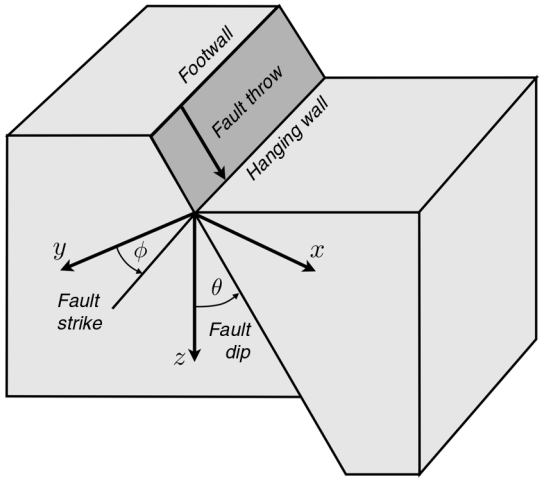

Figure 1. Fault throw is a vector representing one component of displacement of the hanging-wall side of a fault surface relative to the footwall side of the same surface. More generally, fault throw varies within a fault surface, which need not be planar; but fault throw remains everywhere perpendicular to fault strike. The other component, not shown here, corresponds to horizontal displacement in the strike direction.

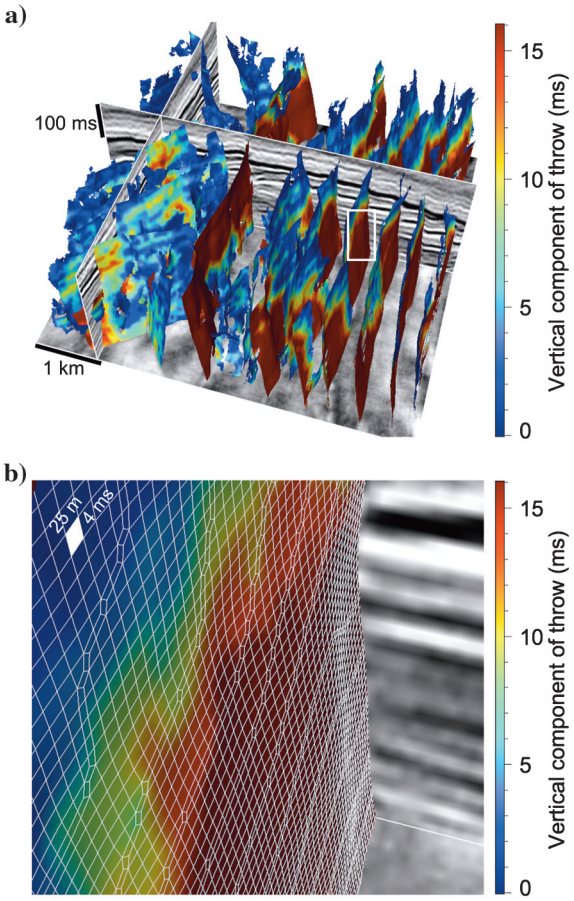

Figure 2. Fault surfaces and fault throws (a) computed automatically from a 3D seismic image. The close-up view (b) of the fault in the white rectangle in (a) shows the quadrilateral mesh used to represent each of the fault surfaces. Each quadrilateral intersects exactly one edge of the 4 ms × 25 m × 25 m image-sampling grid.

Fault images

Several methods for highlighting faults, that is, for computing 3D images of faults from 3D seismic images, are commonly used today. Some compute a measure of the continuity of seismic reflections, such as semblance (Marfurt et al., 1998) or other forms of coherence (Marfurt et al., 1999). Others compute a measure of discontinuity, such as variance (Randen et al., 2001; Van Bemmel and Pepper, 2011), entropy (Cohen et al., 2006), or gradient magnitude (Aqrawi and Boe, 2011). All of these methods are based on the observation that faults may exist where continuity in seismic reflections is low or, equivalently, where discontinuity is high.

However, in small regions within 3D seismic images, continuity may be low for reasons unrelated to faults. Stratigraphic features such as buried channels are well highlighted in seismic images by low continuity. Low continuity is also caused by incoherent noise that is stronger than weak seismic reflections. Even where a fault is present, seismic events may appear to be highly continuous where fault throws are approximately equal to the dominant period (or wavelength) of those events. Event continuity alone is insufficient to distinguish faults.

For these reasons, Gersztenkorn and Marfurt (1999) suggest that any measure of continuity or discontinuity should include some form of averaging within vertical windows that are longer when detecting faults than when detecting stratigraphic features. In effect, these averaging windows smooth together small regions of low continuity that are vertically aligned along faults with significant vertical extent. More recently, Aqrawi and Boe (2011) note that such vertical smoothing of image gradient magnitudes is desirable when highlighting faults.

However, faults are seldom vertical. When averaging any seismic attribute used to highlight faults, we should alter the orientation of this averaging to coincide with the strikes and dips of the faults. Neff et al. (2000) and Cohen et al. (2006) do this in their computation of fault images, as they scan over a range of fault orientations for each sample in a 3D seismic image. The computational cost of such scans can be high when, for each 3D image sample point and for each possible fault orientation, one must process many samples (over 1300 in the example of Cohen et al., 2006) within some neighborhood of the sample point.

One way to reduce this cost is to perform a sequence of scans within 2D image slices. For example, Crawford and Medwedeff (1999) scan over a range of fault azimuths using samples within small neighborhoods of each sample in horizontal 2D slices of a 3D seismic image. Dorn et al. (2012) describe a similar sequence of scans, first over fault azimuths in horizontal 2D image slices and then over fault dips, using for each 3D image sample the azimuth estimated in the first scan. A potential disadvantage of such methods is that fault azimuths may not be well resolved in single horizontal 2D slices of a 3D seismic image.

Fault surfaces

To extract fault surfaces such as those shown in Figure 2 from 3D images of faults requires additional processing, which again has been performed in various ways.

For example, Pedersen et al. (2002, 2003) and Pedersen (2007, 2011) develop the method of ant tracking to merge together small regions of low continuity in 3D fault images into larger fault surfaces.

Gibson et al. (2005) propose a multistage method of constructing larger fault surfaces by merging smaller ones, beginning with small surfaces that correspond to "local discontinuities" in 3D seismic images. Other methods for growing large fault surfaces from small initial surfaces are also proposed by Admasu et al. (2006), Kadlec et al. (2008), and Kadlec (2011). In such methods, seismic interpreters can specify seed points from which to begin growing fault surfaces.

In a more general context, Schultz et al. (2010) describe a direct method for extracting so-called crease surfaces from 3D images without seed points. In one example, they extract surfaces corresponding to ridges in a 3D image of fractional anisotropy, which is computed from 3D diffusion-tensor magnetic resonance images of the human brain. Their method of extracting surfaces works well for 3D images with ridges that are well defined and continuous.

Fault throws

Methods for computing fault surfaces lead naturally to the problem of estimating relative displacements of geologic layers alongside such surfaces. Solutions to this problem are not trivial, in part because of the sinusoidal character of seismic waveforms alongside faults, which can cause coincidental horizontal alignment of seismic events across faults even where fault throws are significant. Another difficulty is that fault throws typically vary within the spatial extent of any fault surface. Nevertheless, several authors have described solutions to the problem of estimating fault throws.

For example, Aurnhammer and Tönnies (2005) demonstrate the use of local crosscorrelations computed in rectangular windows and a genetic algorithm with geologic and geometric constraints to match horizons extracted from both sides of faults in 2D seismic images. Liang et al. (2010) also use local crosscorrelations to estimate fault throws, while simultaneously scanning over fault dips to determine the locations and orientations of faults in 2D seismic images.

Admasu (2008) addresses the problem of estimating fault throws from 3D images through a Bayesian matching of seismic horizons extracted alongside faults in vertical 2D image slices, with the matching for one 2D slice used as a guide for the matching in adjacent slices. This method requires that fault surfaces are approximately orthogonal to the 2D image slices used to compute the fault throws.

In a 3D solution to the problem, Borgos et al. (2003) correlate seismic horizons across faults by clustering into classes local extrema in various attributes computed from 3D seismic images. Carrillat et al. (2004) and Skov et al. (2004) show examples of analyzing fault displacements computed using this method. In another 3D solution, Bates et al. (2009) demonstrate a "geomodel time differential analysis method" for computing fault throws after automatic horizon tracking.

Proposed methods

This paper contributes to solutions of all three of the problems described above: (1) computing 3D fault images (2) extracting fault surfaces, and (3) estimating fault throws. The sequence of solutions proposed here was used to compute the fault surfaces and throws displayed in Figures 2 and 3. Although each of these three solutions was designed in conjunction with the others in the sequence, aspects of any one of them could be adapted to enhance other methods summarized above.

I first compute 3D fault images of an attribute I call fault likelihood. Much like Cohen et al. (2006), I scan over multiple fault strikes and dips to maximize this semblance-based attribute. However, the computational cost of the algorithm I use to perform this scan is independent of the number of samples used in the averaging

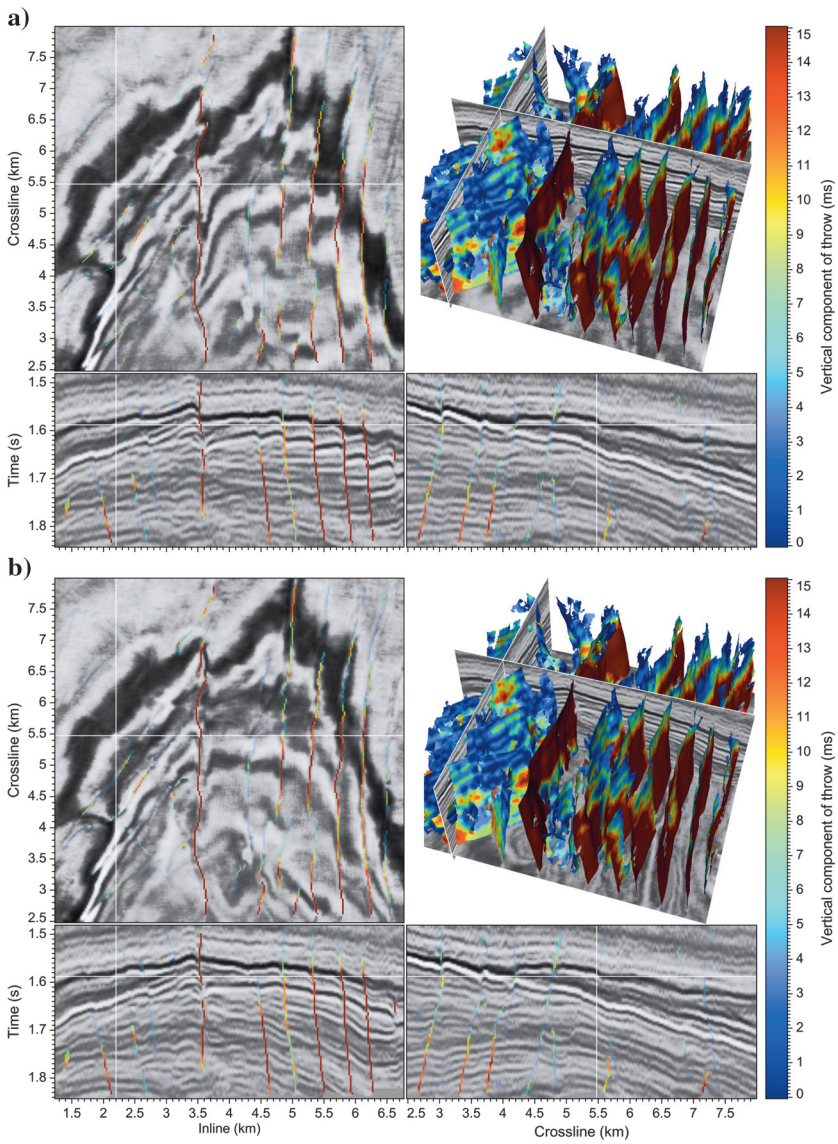

Figure 3. Fault surfaces and fault throws for a 3D seismic image before (a) and after (b) unfaulting. In all image slices, reflectors are more continuous after unfaulting.

performed for each fault orientation. In other words, I improve computational efficiency by eliminating the large factor equal to the number of samples in the windows that Cohen et al. (2006) describe.

I then use the resulting 3D images of fault likelihoods, dips and strikes to extract fault surfaces using a method that is similar to that of Schultz et al. (2010). The fault surfaces shown in Figures 2 and 3 are ridges in 3D images of fault likelihood, and are represented by meshes of quadrilaterals.

Finally, I compute fault throws from differences in values of samples extracted from 3D seismic images alongside fault surfaces. The algorithm I use to compute fault throws is derived from a classic dynamic programming solution (Sakoe and Chiba, 1978) to a problem in speech recognition. Today, that solution is often called dynamic time warping and is here extended to find a spatial warping that best aligns samples of 3D seismic images alongside faults, as illustrated in Figure 3.

FAULT IMAGES

Whereas seismic horizons appear in 3D seismic images as coherent events, a fault appears less prominently as a curviplanar surface across which seismic events are discontinuous, yet correlated, with some displacement, from one side of the fault to the other. Therefore, a useful first step in extracting fault surfaces and estimating fault throws is to first compute images in which faults are most prominent.

Fault-oriented semblance

The method I use for this first step is based on semblance (Taner and Koehler, 1969), and is therefore similar to methods that Marfurt et al. (1998) propose. Like Marfurt et al. (1999), I compute semblances from small numbers (3 in 2D, $3 \times 3 = 9$ in 3D) of adjacent seismic traces, after aligning those traces so that any coherent reflections are horizontal.

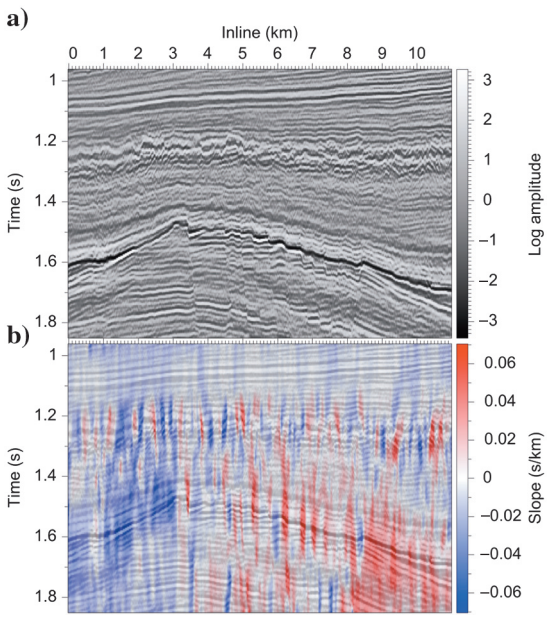

Figure 4. A 2D seismic image after logarithmic gain (a), with estimated reflection slopes (b) displayed in translucent colors.

That alignment requires prior estimates of reflection slopes, like those displayed in Figure 4b. I computed these slopes from structure tensors (e.g., van Vliet and Verbeek, 1995; Weickert, 1999) for the seismic image displayed in Figure 4a. The logarithmic gain applied to that image has merely increased the relative amplitudes of weaker reflections.

Given estimates of reflection slopes, we might define structure-oriented semblance as

$$\text{semblance} = \frac{\langle \text{image} \rangle_s^2}{\langle \text{image}^2 \rangle_s}, \tag{1}$$

where, for each semblance value, $\langle \cdot \rangle_s$ denotes a *structure-oriented* averaging of whatever is inside the brackets, 3 (2D) or $3 \times 3 = 9$ (3D) image sample values. However, semblance ratios computed in this way will vary wildly where the numerators and denominators are small.

This instability is one reason that we typically perform an additional smoothing of numerators and denominators before computing semblance ratios:

$$\text{semblance} = \frac{\langle \langle \text{image} \rangle_s^2 \rangle_f}{\langle \langle \text{image}^2 \rangle_s \rangle_f}, \tag{2}$$

where the subscript f implies that this additional smoothing $\langle \cdot \rangle_f$ should be fault oriented.

A second reason is that discontinuities in seismic reflections corresponding to faults are most significant for strong reflections that may be widely separated. The additional fault-oriented smoothing in equation 2 links together these localized discontinuities, where semblance numerators can be much smaller than semblance denominators.

It is for this second reason that Gersztenkorn and Marfurt (1999) recommend the use of long vertical smoothing windows when highlighting structural features, such as faults, and short windows when highlighting stratigraphic features such as channels. In proposing a different gradient-based measure of discontinuity, Aqrawi and Boe (2011) likewise use a vertical smoothing of that measure for the same reason.

Figure 5a shows semblances computed using a highly efficient two-sided-exponential filter to implement $\langle \cdot \rangle_f$ in equation 2 with vertical smoothing. The impulse response of this recursive filter is infinitely long, decays smoothly to zero, and in this example approximates a Gaussian filter with half-width $\sigma = 20$ samples. Significantly, the computational cost of the filter is independent of this half-width.

This vertical smoothing of semblance numerators and denominators accounts for the vertical extent of features with low semblance apparent in Figure 5a. To highlight these features, I define an attribute *fault likelihood* f by

$$f \equiv 1 - \text{semblance}^8. \tag{3}$$

The choice of power eight is somewhat arbitrary; it increases the contrast between samples with low and high fault likelihoods, as shown in Figure 5b, where lower likelihoods $f < 0.5$ are transparent, to enhance visibility of unfaulted seismic reflectors.

Although features in the semblance and fault likelihood images shown in Figure 5 have significant vertical extent, these features are

not well aligned with faults because those faults are not vertical. To improve the fault likelihood attribute f, we should instead smooth semblance numerators and denominators along the faults. Our problem is that we have not yet determined fault locations or orientations.

Scanning for fault orientations

This sort of problem is common in seismic data processing, for example, when we must perform normal moveout corrections without knowing normal moveout velocities. A common solution is to perform a scan for multiple velocities to find the velocities that maximize some (often semblance-based) measure of alignment. Here, I scan over negative and positive fault dip angles θ (where $\theta = 0$ corresponds to vertical faults) to find those dips that maximize fault likelihoods f.

Figure 6 illustrates the results of nonvertical smoothing for two different fault dips θ in this scan. These examples demonstrate that fault likelihoods tend to be largest when smoothing of semblance numerators and denominators is performed along the faults, which are not vertical.

To perform this nonvertical smoothing efficiently for each fault dip θ, I shear semblance numerator and denominator images horizontally to make faults with that dip appear to be vertical, apply the fast vertical smoothing filter described above, and unshear the smoothed images before computing their ratio. Horizontal shearing (and unshearing) of images requires a simple horizontal shift of each row of image pixels, where the amount of shift varies linearly with row index. After shearing, faults with some nonvertical (nonzero) dip θ will appear to be vertical, so that we may apply the recursive smoothing filter to each column of pixels in the sheared image, independently, before unshearing.

Recall that the computational cost of the recursive smoothing filter does not depend on the extent of smoothing, which is controlled by the half-width parameter σ. This recursive filter and the shearing and unshearing for nonvertical faults are largely responsible for reducing the computational cost of the scan over fault dips, relative to alternative scans as Neff et al. (2000) and Cohen et al. (2006) describe.

This cost reduction is especially significant when scanning over fault dips θ and strikes ϕ for 3D seismic images. In these scans, smoothing of semblance numerators and denominators should be 2D, within potential fault planes spanned by fault strike and dip vectors. For each fault strike angle ϕ in the scan, I rotate the semblance numerator and denominator images, to align the fault strike direction with the crossline image axis. I then smooth those rotated images once horizontally in the crossline direction before scanning over fault dips θ. The computational costs of rotation and crossline smoothing for each fault strike ϕ are therefore negligible compared to the cost of the scans over fault dips θ. The cost of an entire scan over fault strikes and dips for a 3D image is dominated by a sequence of independent scans over fault dips for multiple 2D images.

An alternative to the sequence of rotation, horizontal smoothing, shearing, vertical smoothing, and unshearing described above is to implement the fault-oriented smoothing filter $\langle\cdot\rangle_f$ with a 3D fast Fourier transform (FFT). However, my implementation of such an FFT-based smoothing filter is about three times slower than the scanning process used here. Rotation and shearing are common operations in image processing that can all be performed with computational cost less than that for FFTs.

Computational cost is also a factor in my choice of the semblance attribute, which requires smoothing of only the two (numerator and denominator) images in equation 2. Alternatives such as the normalized correlation coefficient (Rodgers and Nicewander, 1988) or the eigenstructure-based coherence that Gersztenkorn and Marfurt (1999) describe would require fault-oriented smoothing $\langle\cdot\rangle_f$ of more than two images for each fault strike and dip in the scan.

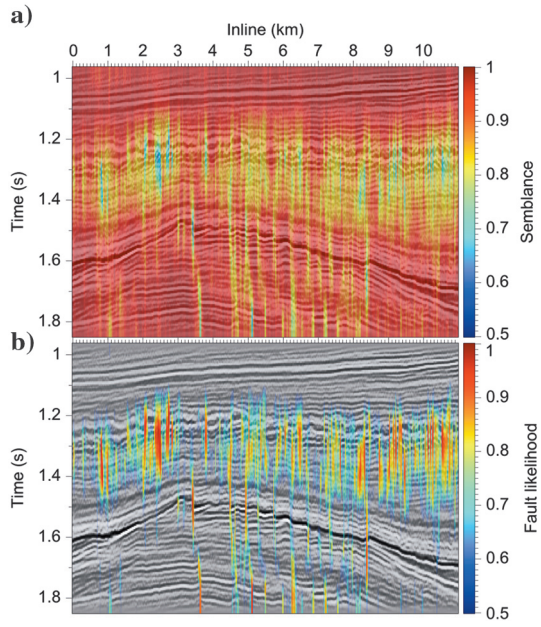

Figure 5. Semblances (a) and fault likelihoods (b) computed using vertical smoothing of semblance numerators and denominators.

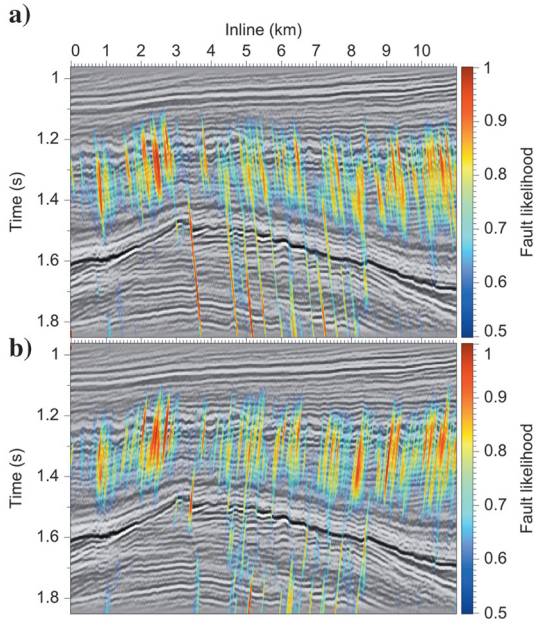

Figure 6. Fault likelihoods computed for two different fault dips θ, one positive (a) and the other negative (b), in a scan used to estimate fault dips. The positive dip is consistent with many of the major faults. The negative dip is consistent with antithetic faults.

Ridges of fault likelihood

The purpose of the scan over fault strikes and dips is to find, for each image sample, the angles ϕ and θ that maximize fault likelihood f. I begin with a fault likelihood image $f = 0$. Then, for each orientation (ϕ, θ) in the scan, where the fault likelihood $f_{(\phi,\theta)}$ exceeds the maximum likelihood stored in f, I update f and also save the corresponding strike ϕ and dip θ. When complete, the results of this scan are images of maximum fault likelihoods and corresponding fault strikes and dips.

Figure 7a shows fault likelihoods f computed in this way with a scan over fault dips for the 2D seismic image. Ridges of high fault likelihood generally coincide with faults apparent in the seismic image. These ridges can be extracted by scanning each row of the fault likelihood image, preserving only local maxima, and setting fault likelihoods elsewhere to zero. In effect, this process thins the fault likelihood image, reducing the number of image samples at which a fault might be considered to exist.

Figure 7b shows ridges extracted from the fault image of Figure 7a. Parts of some ridges, especially those with lower fault likelihoods, may not coincide with faults. At this stage, I do not suppress these parts, although one might easily remove some of them by thresholding fault likelihoods.

The seismic image displayed in Figure 7b is the result of structure-oriented filtering applied to the image shown in Figure 7a. This filtering is similar to that of Fehmers and Höcker (2003), but it is here constrained to not smooth across the thin ridges of fault likelihood. When estimating fault throws as described below, I use this filtering to attenuate noise and smooth coherent reflections toward, but not across, faults, much as human interpreters do visually.

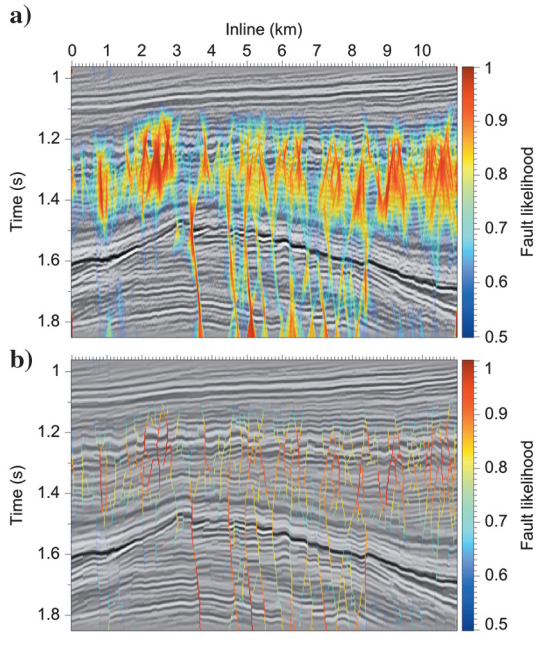

Figure 7. Fault likelihoods computed by scanning over fault dips θ, (a) before and (b) after thinning. To facilitate estimation of fault throws, the seismic image in (b) has been smoothed along structures, but not across faults.

FAULT SURFACES

We can easily imagine how to extract fault curves from 2D fault images like those shown in Figure 7. For example, we might somehow link together samples with nonzero fault likelihood in the thinned image of Figure 7b. We could then use samples of the seismic image on the left and right sides of the extracted fault curves to estimate fault throws.

It is more difficult to construct fault surfaces like those shown in Figures 2 and 3 from 3D fault images. One problem is how to best represent a fault surface, which need not be aligned with any axis of the sampling grid for the 3D seismic image. Many fault surfaces cannot be projected onto a plane, and they cannot be well represented by a single-valued function of coordinates within such a plane (e.g., Walsh et al., 1999).

Also, the resolution with which we sample fault surfaces will be important when we compute fault throws. In that computation, we must be able to efficiently iterate upward and downward along fault curves of constant strike as we analyze seismic image samples alongside fault surfaces. We must also be able to efficiently traverse horizontally along fault traces for which time (or depth) is constant.

For these reasons, I represent fault surfaces with unstructured meshes of quadrilaterals (quads) like those shown in Figure 2b.

Extracting quads from fault images

My first step in constructing quad meshes is to extract a set of quads, not yet connected, from the 3D image of fault likelihoods. That 3D image is analogous to the 2D image of fault likelihoods shown in Figure 7a.

As shown in Figure 8, each quad in a fault surface intersects exactly one edge of the 3D sampling grid of a fault image. I find these fault-edge intersections using a method similar to the one that

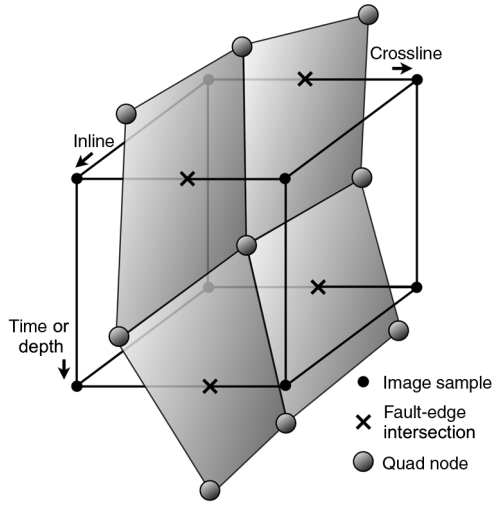

Figure 8. Four adjacent quadrilaterals in a fault surface share a node that lies within one cell of the 3D sampling grid of a fault image. Spatial coordinates of the quad node are averages of the coordinates of intersections of the fault surface and edges of the image sampling grid. Quads are linked to quad neighbors where two edges in any of the six faces of a cell are intersected by a fault, and this example shows four such links.

Schultz et al. (2010) develop. Specifically, I assume that fault surfaces are ridges in 3D images of fault likelihoods, analogous to the fault curves apparent in the 2D images shown in Figure 7.

Recall that curves apparent in Figure 7b are merely collections of pixels corresponding to local maxima within each row of pixels in the image of fault likelihoods shown in Figure 7a. If we walk up or down along a fault in Figure 7a, the fault likelihood may increase or decrease. If we instead walk horizontally across a fault in Figure 7, the fault likelihood will first increase, then decrease, as we cross a ridge of fault likelihood. Faults are most likely to exist at maxima, where horizontal derivatives of fault likelihood are zero.

A more precise location of ridges of fault likelihood can be obtained by first computing derivatives of fault likelihood in directions perpendicular to faults and then interpolating with subpixel precision the locations at maxima where these derivatives are zero. For each image sample, the perpendicular direction can be computed from the image of fault dips θ obtained in the scan to maximize fault likelihoods.

For 3D images, ridges of fault likelihood intersect edges of cells in the 3D sampling grid (see Figure 8), and can be found by considering all such edges, one at a time. Each edge is defined by two adjacent samples in the 3D image of fault likelihood. For each of those two image samples, we have estimates of fault likelihood f, strike ϕ, and dip θ. These attributes enable us to compute derivatives of fault likelihood in directions perpendicular to faults, and from those directional derivatives we can determine whether or not a ridge (a local maximum) of fault likelihood intersects the edge of the seismic sampling grid between the two adjacent samples.

As Schultz et al. (2010) describe, those same derivatives also enable us to interpolate the location between samples of the fault-edge intersection (where the directional derivative is zero), if one exists. The spatial coordinates of a quad node located within any cell of the sampling grid are the average of coordinates computed for all fault-edge intersections in that cell. This interpolation and averaging yields quads that coincide (with subvoxel precision) with ridges in the 3D image of fault likelihoods.

By analyzing all edges in the 3D sampling grid in this way, I extract quads that intersect edges where faults may exist. Most quads will share nodes with adjacent quads, and when displayed may appear to be parts of large fault surfaces. However, at this point in the process of fault surface extraction, quads have not yet been linked together to form a surface mesh. We have only a collection of quads, what is sometimes called "quad soup."

Linking quads

The next step in extracting fault surfaces is to link quads together to form a mesh. Each quad in such a mesh will be linked to no more than four quad neighbors, quads that share an edge between two of their nodes. A simple method to find such links is to again loop over all cells in the image sampling grid, like the cell shown in Figure 8. Each cell has six faces, and each face has four edges. For each face, we look for edges that are intersected by faults, edges for which we have already constructed quads. If fault-edge intersections occur for two edges in the face, we link the two quads constructed for those two edges. In Figure 8, each quad would be linked in this way to exactly two neighbors.

If fault-edge intersections occur for more than two edges in a single face of the image sampling grid cell, I link none of the corresponding quads that intersect those edges. This choice implies that two fault surfaces extracted in this way cannot intersect precisely, although they may be separated by only one grid sample.

After all links between quads and their neighbors have been found, I apply several filters to remove links that are inconsistent with geologically feasible fault surfaces. For example, I unlink two quads if either of them has no other neighbors or has only one neighbor on the opposite side. Such quads tend to appear as fins or bridges between two nearby fault surfaces, and by unlinking them I ensure that fault surfaces are nowhere too skinny, with a width or height of only one quad.

Another filter unlinks two quads if the dihedral angle between them (computed from their normal vectors) is less than 90°. This filter ensures that the orientation of a fault surface does not vary too rapidly from one quad to the next.

Let the orientation of a quad be defined by its normal vector, which points toward a viewer that sees the four quad nodes labeled in counterclockwise order. From the opposite side, those same nodes would appear to be labeled in clockwise order. We can then define the back and front sides of a quad such that the quad normal vector points toward a viewer looking at the front side.

At this stage in the process of fault surface extraction, the orientations of quad neighbors may be inconsistent. That is, when linking quads, the normal vectors of a quad and its quad neighbors may point in opposite directions.

Constructing oriented fault surfaces

After extracting quads and linking them to neighbors, the final step in extracting fault surfaces is to find collections of quads that are linked either directly as neighbors or recursively as neighbors of neighbors. These collections form quad meshes that represent fault surfaces.

I assume that these fault surfaces are orientable, that they have topologically distinct front and back sides, unlike the surfaces extracted from medical images by Schultz et al. (2010). In other words, I assume that fault normal vectors can be chosen consistently for every quad in the surface, so that the front side of every quad coincides with the front side of the surface. Therefore, when recursively collecting quads to form fault surfaces, I flip the orientations of quads as necessary to be consistent with their neighbors.

In the extraction of surfaces shown in Figure 3 I found 1922 surfaces, and all of them were orientable. However, some surfaces were quite small; this figure displays only the 20 largest fault surfaces, those with at least 2000 quads.

This selection of faults based on their size is just one example of the sort of filtering that is possible after constructing fault surfaces. We could also filter these surfaces based on their average strikes or dips (Pedersen et al., 2003; Pedersen, 2007, 2011), their fault likelihoods, or any combination of statistics derived from attributes computed for the quads that comprise the surfaces.

FAULT THROWS

Conceptually, the problem of estimating fault throws from 3D seismic images is a simple one. We must correlate seismic reflections on one side of the fault with those on the other side and compute vector displacements between corresponding reflections. In practice, this problem is difficult for several reasons.

Difficulties in estimating fault throws

One difficulty is that fault throws vary within a fault surface and within any local windows that we might use to correlate seismic reflections on opposite sides of the fault. To ease this difficulty, Aurnhammer and Tönnies (2005) use several geologic and geometric constraints in a genetic algorithm to estimate fault displacements. Still, inconsistency remains in estimating fault throw by crosscorrelation of windows of image samples in which that throw may vary significantly. This first difficulty is exacerbated by the fact that crosscorrelation windows must be at least as long as the longest fault throw vector to be estimated.

A second difficulty is related to the fact that we can best estimate fault throws from strong seismic reflections with high signal-to-noise ratios, but these reflections may be far apart, with weaker and noisier reflections in between. Constraints (e.g., Aurnhammer and Tönnies, 2005) are therefore needed to ensure continuity of fault throws estimated between strong reflections.

A third difficulty lies in constraining fault throws to vary smoothly in dip and strike directions within a fault surface. Imposing this constraint is difficult partly because fault surfaces often cannot be projected onto a plane (e.g., Walsh et al., 1999). This means that we cannot estimate fault throws from 3D seismic images by crosscorrelation of one 2D image extracted from the footwall side of a fault surface with another 2D image extracted from the hanging-wall side.

Dynamic warping

I address the difficulties summarized above with an extension of a classic method for estimating relative shifts between two acoustic signals in the problem of speech recognition (Sakoe and Chiba, 1978). This method is today widely known as dynamic time warping. For the problem of estimating fault throws, the most important aspect of this method is that it estimates time-varying (dynamic) shifts between two sampled functions of time, without any local correlation windows. Instead, dynamic time warping finds shifts that minimize alignment errors, squared differences between sample values computed for different shifts, while strictly constraining the rate at which those shifts can vary with time.

Hale (2013) describes an extension of the dynamic time warping solution to the problem of dynamic image warping, in which we seek to constrain estimates of relative shifts between two images to vary smoothly in all directions. The extension is an alternating sequence of vertical (top-down, bottom-up) and horizontal (left-right, right-left) smoothings of alignment errors, followed by the classic dynamic time warping algorithm. The vertical and horizontal smoothings are nonlinear but computationally efficient.

As mentioned above, we often cannot reduce the problem of estimating fault throws to that of finding an optimal warping between footwall and hanging-wall images. We can, however, adapt the image warping solution to the problem of estimating fault throws, by extending this solution to quad meshes like that displayed in Figure 2b.

These quad meshes provide the links necessary to iterate upward and downward along fault curves of constant strike and to thereby compute alignment errors from seismic image sample values located on opposite sides of a fault surface. Quad links also enable us to smooth those alignment errors vertically and horizontally, just as we would in dynamic image warping, where such links are implied by a uniform sampling grid.

In this way, I estimate vertical components of throw vectors from the footwall side to the hanging-wall side of a fault. The vertical components of throw shown in Figures 2 and 3 are smooth because of the constraints imposed in dynamic warping. They are nonnegative because the

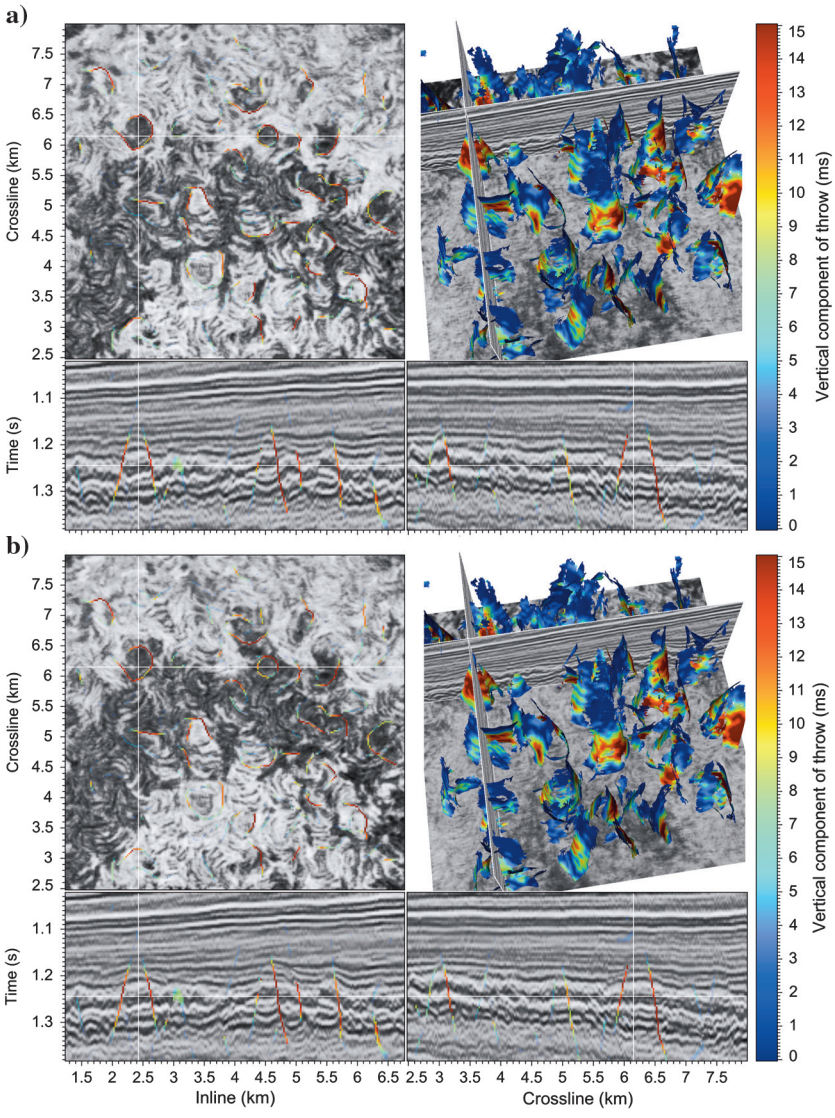

Figure 9. Fault surfaces and fault throws for a 3D seismic image (a) before and (b) after unfaulting. The shape for many of these faults is roughly conical, and the two vertical sections intersect near the symmetry axis of one of these conical faults.

faults in this example are normal faults. For each quad, the vertical component of throw tells us how far upward or downward along a fault curve we must iterate within the quad mesh to determine the horizontal inline and crossline components (not shown) of the throw vector.

In summary, the dynamic warping process proposed here computes fault throws that minimize the sum of squared differences between image sample values on opposite sides of a fault, while constraining those fault throws to vary smoothly within the fault surface.

Unfaulted images

A good test of the fidelity of estimated fault throws is to use them to undo faulting apparent in the seismic images from which they were derived. Luo and Hale (2012) describe this unfaulting process in detail. Here, I use this process only to illustrate the accuracy of fault surfaces and fault throws computed using the methods proposed in this paper.

Figure 3 provides one example in which fault surfaces have roughly planar shapes. Faulting and unfaulting are most apparent in the inline sections because faults with large throws have strike vectors that are approximately aligned with the orthogonal crossline direction. Throws for these faults vary smoothly while generally increasing with depth.

Visual comparison of the continuity of reflections before and after unfaulting suggests that estimated fault throw vectors are generally accurate. One location where estimated fault throw appears to be incorrect (for unknown reasons) is located at about 1.7 s and 4.5 km in the crossline section.

Figures 9 and 10 show faults extracted from the same seismic image for a shallower portion of the subsurface with a more chaotic structure. Many of the faults extracted from this portion have roughly conical shapes, in which the apex of each cone lies above its base. In Figure 9, the two vertical image slices intersect near the symmetry axis of one of these conical fault surfaces.

These conical shapes became apparent to me only after extracting fault surfaces, partly because I had never before seen faults with such shapes, and so I did not recognize their appearance in horizontal and vertical slices of the 3D seismic image. This experience highlighted an important benefit in using an automated process to extract information from 3D seismic images. The process used here cannot exclude such shapes merely because they are unexpected.

After recognizing the conical shapes of these faults, they are easily seen in 3D seismic images, even without the color overlays shown in Figures 9 and 10. In vertical sections, conical faults appear to be hyperbolic because a vertical slice (conic section) of a cone is a hyperbola. When interactively moving a vertical image slice through the 3D image, one can clearly see these hyperbolic discontinuities rising and falling as the image slice moves through the faults.

Moreover, even in this relatively chaotic part of the seismic image, reflections are more continuous in the unfaulted image than in the original image. Some discontinuities that remain may be due to using only the 30 largest fault surfaces, those with at least 2000 quads.

Vertical exaggeration in all of these figures makes faults appear to be more vertical than they really are; an approximate time-to-depth conversion (with 1 s equivalent to 1 km) suggests that the dips of most faults are roughly 45° from vertical.

All of the seismic images shown in this paper are subsets of a 3D seismic image made publicly available by dGB Earth Sciences (1987). I therefore encourage others to study similar subsets, to look for conical faults apparent as circular arcs in horizontal slices and hyperbolic curves in vertical sections, before assuming that

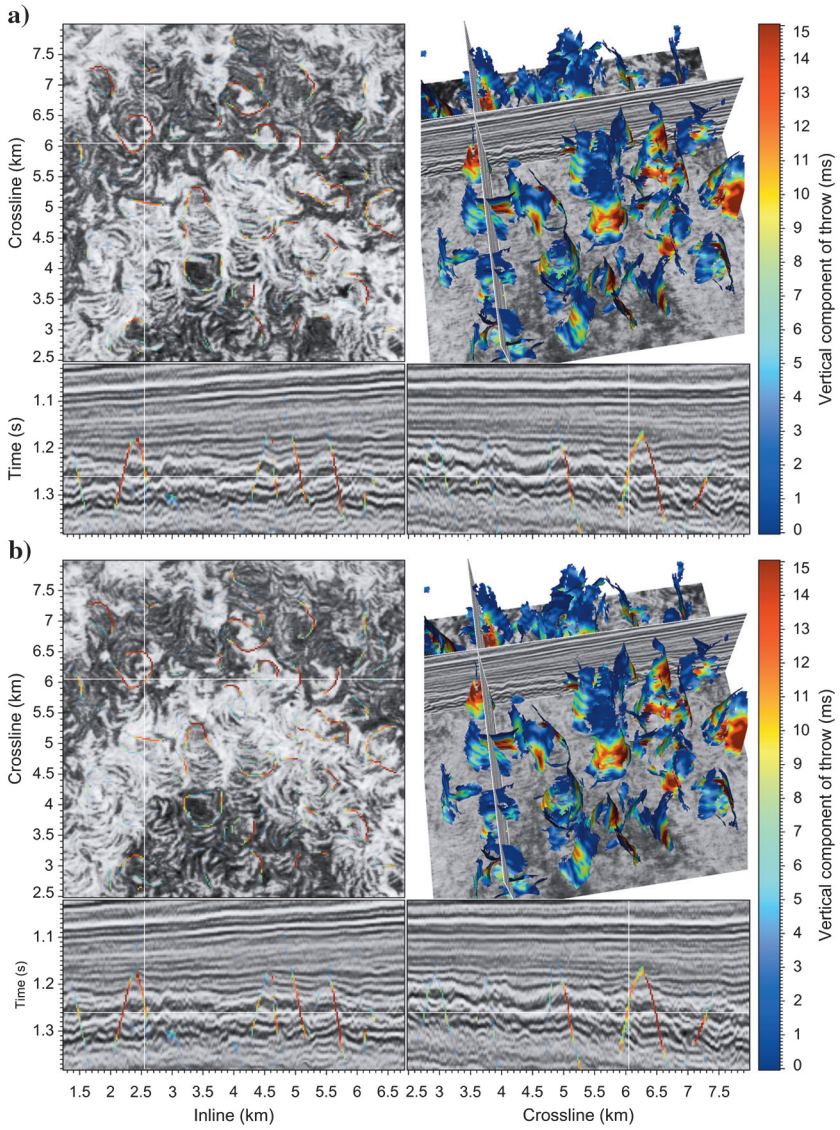

Figure 10. Different image slices for the same fault surfaces and fault throws displayed in Figure 9, (a) before and (b) after unfaulting. Conical faults here appear as fragments of hyperbolas in vertical seismic sections.

these faults have polygonal or other shapes that may be more familiar.

CONCLUSION

The methods proposed in this paper comprise a three-step process to (1) compute fault images of likelihood, strike, and dip (2) extract fault surfaces, and (3) estimate fault throws. I once hoped to bypass the first two steps, to directly compute fault throws everywhere and then let faults be defined by locations where fault throws are significant. However, I have been unable to find an implementation of this conceptually simpler one-step process that is computationally feasible for 3D seismic images.

It is significant that the scan in the first step yields images of fault strikes and dips for which the fault likelihood is maximized. These estimates of fault orientations are especially useful in the second step used to extract fault surfaces from ridges in fault likelihood.

I chose the quad-mesh representation for fault surfaces in part to facilitate the third step of estimating fault throws. Because throw vectors connect samples on one side of a fault to those on the other side, it is especially convenient that quads in the fault surface lie between two adjacent samples of the seismic image at the same time or depth. In addition, the quad mesh also provides up-down and left-right connectivity needed to implement the dynamic warping algorithm used to estimate vertical throws.

Most of the computation time in this three-step process is spent in the first step, which currently requires a scan over all possible fault orientations. I improve the computational efficiency of this scan by using fast recursive smoothing filters within each potential fault plane, but further improvements may be worthwhile. My implementation of this scan for about 500 different fault orientations requires less than 2 h to process a 3D image of 1000^3 samples on a modern workstation.

Perhaps the biggest current limitation in this process is in its handling of intersecting faults. As discussed above, to simplify the extraction of fault surfaces and estimation of fault throws, I do not yet permit two fault surfaces to intersect precisely, requiring instead that they be separated by at least one sample of the seismic image sampling grid. Further work is required to extend these two steps to properly account for fault intersections.

ACKNOWLEDGMENTS

In much of the research described in this paper, I benefited greatly from discussions with others, including L. Liang, M. Maučec, B. Howard, D. Witte, and A. Mironova. I very much appreciate suggestions by D. Gao and two anonymous reviewers that led to significant revision of this paper. The 3D seismic image used in this study was graciously provided by dGB Earth Sciences.

REFERENCES

Admasu, F., 2008, A stochastic method for automated matching of horizons across a fault in 3D seismic data: Ph.D. thesis, Otto-von-Guericke University Magdeburg.

Admasu, F., S. Back, and K. Toennies, 2006, Autotracking of faults on 3D seismic data: Geophysics, **71**, no. 6, A49–A53, doi: 10.1190/1.2358399.

Aqrawi, A., and T. Boe, 2011, Improved fault segmentation using a dip-guided and modified 3D Sobel filter: 81st Annual International Meeting, SEG, Expanded Abstracts, 999–1003.

Aurnhammer, M., and K. Tönnies, 2005, A genetic algorithm for automated horizon correlation across faults in seismic images: IEEE Transactions on Evolutionary Computation, **9**, 201–210, doi: 10.1109/TEVC.2004.841307.

Bates, K., T. Cheret, F. Pauget, and S. Lacaze, 2009, Vertical displacement on faults extracted from seismic, offshore Nigeria case study: 71st Annual International Conference and Exhibition, EAGE, Extended Abstracts, 5800.

Borgos, H., T. Skov, T. Randen, and L. Sonneland, 2003, Automated geometry extraction from 3D seismic data: 73rd Annual International Meeting, SEG, Expanded Abstracts, 1541–1544.

Carrillat, A., H. Borgos, T. Randen, L. Sonneland, L. Kvamme, and K. Hansch, 2004, Fault system analysis with automatic fault displacement estimates — A case study: 66th Annual International Conference and Exhibition, EAGE, Extended Abstracts, B037.

Cohen, I., N. Coult, and A. Vassiliou, 2006, Detection and extraction of fault surfaces in 3D seismic data: Geophysics, **71**, no. 4, P21–P27, doi: 10.1190/1.2215357.

Crawford, M., and D. Medwedeff, 1999, Automated extraction of fault surfaces from 3-D seismic prospecting data: U. S. Patent 5,987,388.

dGB Earth Sciences, 1987, 3D seismic image, F3 Block, offshore Netherlands, http://www.opendtect.org/osr/, accessed 13 November 2012.

Dorn, G., B. Kadlec, and P. Murtha, 2012, Imaging faults in 3D seismic volumes: 82nd Annual International Meeting, SEG, Expanded Abstracts, 1–5.

Fehmers, G., and C. Höcker, 2003, Fast structural interpretation with structure-oriented filtering: Geophysics, **68**, 1286–1293, doi: 10.1190/1.1598121.

Gersztenkorn, A., and K. Marfurt, 1999, Eigenstructure-based coherence computations as an aid to 3-D structural and stratigraphic mapping: Geophysics, **64**, 1468–1479, doi: 10.1190/1.1444651.

Gibson, D., M. Spann, J. Turner, and T. Wright, 2005, Fault surface detection in 3-D seismic data: IEEE Transactions on Geoscience and Remote Sensing, **43**, 2094–2102, doi: 10.1109/TGRS.2005.852769.

Hale, D., 2013, Dynamic warping of seismic images: Geophysics, **78**, this issue, doi: 10.1190/geo2012-0327.1.

Kadlec, B., 2011, Visualization of geologic features using data representations thereof: U. S. Patent Application 2011/0115787.

Kadlec, B., G. Dorn, H. Tufo, and D. Yuen, 2008, Interactive 3-D computation of fault surfaces using level sets: Visual Geosciences, **13**, 133–138, doi: 10.1007/s10069-008-0016-9.

Liang, L., D. Hale, and M. Maučec, 2010, Estimating fault displacements in seismic images: 80th Annual International Meeting, SEG, Expanded Abstracts, 1357–1361.

Luo, S., and D. Hale, 2012, Unfaulting and unfolding 3D seismic images: 82nd Annual International Meeting, SEG, Expanded Abstracts, 1–6.

Marfurt, K., R. Kirlin, S. Farmer, and M. Bahorich, 1998, 3-D seismic attributes using a semblance-based coherency algorithm: Geophysics, **63**, 1150–1165, doi: 10.1190/1.1444415.

Marfurt, K., V. Sudhaker, A. Gersztenkorn, K. Crawford, and S. Nissen, 1999, Coherence calculations in the presence of structural dip: Geophysics, **64**, 104–111, doi: 10.1190/1.1444508.

Neff, D., J. Grismore, and W. Lucas, 2000, Automated seismic fault detection and picking: U. S. Patent 6,018,498.

Pedersen, S., 2007, Image feature extraction: U. S. Patent 7,203,342.

Pedersen, S., 2011, Image feature extraction: U. S. Patent 8,055,026.

Pedersen, S., T. Randen, L. Sonneland, and O. Steen, 2002, Automatic 3D fault interpretation by artificial ants: 72nd Annual International Meeting, SEG, Expanded Abstracts, 512–515.

Pedersen, S., T. Skov, A. Hetlelid, P. Fayemendy, T. Randen, and L. Sonneland, 2003, New paradigm of fault interpretation: 73rd Annual International Meeting, SEG, Expanded Abstracts, 350–353.

Randen, T., S. Pedersen, and L. Sonneland, 2001, Automatic extraction of fault surfaces from three-dimensional seismic data: 71st Annual International Meeting, SEG, Expanded Abstracts, 551–554.

Rodgers, J., and W. Nicewander, 1988, Thirteen ways to look at the correlation coefficient: The American Statistician, **42**, 59–66, doi: 10.2307/2685263.

Sakoe, H., and S. Chiba, 1978, Dynamic programming algorithm optimization for spoken word recognition: IEEE Transactions on Acoustics, Speech, and Signal Processing, **26**, 43–49, doi: 10.1109/TASSP.1978.1163055.

Schultz, T., H. Theisel, and H.-P. Seidel, 2010, Crease surfaces: from theory to extraction and application to diffusion tensor MRI: IEEE Transactions on Visualization and Computer Graphics, **16**, 109–119, doi: 10.1109/TVCG.2009.44.

Skov, T., M. Oygaren, H. Borgos, M. Nickel, and L. Sonneland, 2004, Analysis from 3D fault displacement extracted from seismic data: 66th Annual International Conference and Exhibition, EAGE, Extended Abstracts, C009.

Taner, M., and F. Koehler, 1969, Velocity spectra — Digital computer derivation and applications: Geophysics, **34**, 859–881, doi: 10.1190/1.1440058.

Van Bemmel, P., and R. Pepper, 2011, Seismic signal processing method and apparatus for generating a cube of variance values: U. S. Patent 6,151,555.

van Vliet, L., and P. Verbeek, 1995, Estimators for orientation and anisotropy in digitized images, *in* J. van Katjweik, ed., Proceedings of the first annual conference of the Advanced School for Computing and Imaging: ASCI, 442–450.

Walsh, J., J. Watterson, W. Bailey, and C. Childs, 1999, Fault relays, bends and branchlines: Journal of Structural Geology, **21**, 1019–1026, doi: 10.1016/S0191-8141(99)00026-7.

Weickert, J., 1999, Coherence-enhancing diffusion filtering: International Journal of Computer Vision, **31**, 111–127, doi: 10.1023/A:1008009714131.