GEOPHYSICS, VOL. 83, NO. 5 (SEPTEMBER-OCTOBER 2018); P. O67–O82, 16 FIGS.
10.1190/GEO2018-0115.1

Automatic fault interpretation with optimal surface voting

Xinming Wu[1] and Sergey Fomel[1]

ABSTRACT

Numerous types of fault attributes have been proposed to detect faults by measuring reflection continuities or discontinuities. However, these attributes can be sensitive to other seismic discontinuities, such as noise and stratigraphic features. In addition, fault features within a fault attribute image often cannot be continuously tracked. We have developed an optimal surface-voting method to enhance a fault attribute image so that the noisy features (unrelated to faults) are suppressed whereas the fault features become cleaner and more continuous. In this method, we first automatically pick seed points from the input attribute image and use these seeds as control points to compute optimal surface patches that pass through the control points and follow globally maximum fault attribute values. Then, we consider all the computed surfaces as voters and define voting scores for each voter by using fault attribute values that are smoothed along the surface voter. We further collect voting scores of all the voters to compute a voting score map as a new fault attribute image, in which fault features (with high scores) are much cleaner, sharper, and more continuous than those in the input attribute image. With the optimal surface voters, we can also accurately estimate fault orientations (strikes and dips) by computing weighted averages of the surface voter orientations. From a voting score map with clean and continuous fault features, fault surfaces can be extracted by tacking the fault features along the estimated fault orientations. The computational cost of the method depends on the number of seed points, not the size of the seismic volume, which makes the method highly efficient. With an four-core computer, our parallel implementation can process more than 1000 seeds in 1 s to compute the corresponding optimal voting surfaces and a final voting score map.

INTRODUCTION

Fault interpretation from seismic images is important for subsurface structural interpretation because faults themselves are important structural surfaces and because faults can be used as an important boundary control for interpreting seismic horizons (Luo and Hale, 2013; Wu and Hale, 2016b; Wu et al., 2016). In addition, fault interpretation is important for reservoir exploration and reducing risk in well placement. Therefore, fault interpretation remains a key step of seismic interpretation.

A seismic image contains rich structural and stratigraphic features such as reflections (horizons), faults, and channels. Reflections are the most dominant features within a seismic image, whereas faults and channels are typically apparent as reflection discontinuities. Therefore, fault interpretation from a seismic image often requires first computing a second fault attribute image, in which only the fault features are dominant while reflections and other features are removed. Numerous seismic attributes have been proposed to detect faults by measuring seismic reflection continuity such as semblance (Marfurt et al., 1998) and coherency (Marfurt et al., 1999; Li and Lu, 2014; Wu, 2017), or reflection discontinuity such as variance (Van Bemmel and Pepper, 2000; Randen et al., 2001), curvature (Roberts, 2001; Al-Dossary and Marfurt, 2006; Di and Gao, 2016), and gradient magnitude (Aqrawi and Boe, 2011). However, these seismic attributes can be sensitive to noise and stratigraphic features, which are also apparent as reflection discontinuities within a seismic image. In addition, fault features often cannot be continuously tracked in such seismic attribute images of reflection continuity or discontinuity. Therefore, seismic reflection continuity of discontinuity alone is insufficient to detect faults (Hale, 2013b).

To better distinguish faults from other features in a seismic image, Gersztenkorn and Marfurt (1999) suggest to use vertically elongated windows in computing seismic coherence to enhance fault features while suppressing stratigraphic features by assuming that faults are more vertically aligned than stratigraphic features. Similarly, some other authors (Bakker, 2002; Hale, 2009; Wu,

Manuscript received by the Editor 14 February 2018; revised manuscript received 8 May 2018; published ahead of production 09 June 2018; published online 21 August 2018.
[1]The University of Texas at Austin, Bureau of Economic Geology, Austin, Texas, USA. E-mail: xinming.wu@beg.utexas.edu; sergey.fomel@beg.utexas.edu.

2017) suggest applying smoothing in directions perpendicular to seismic reflections in computing coherence or semblance by assuming that faults are typically normal to reflections.

However, faults are seldom vertical and are not necessarily perpendicular to seismic reflections. Therefore, Pedersen et al. (2002, 2003) and Pedersen (2007, 2011) propose an ant-tracking method to enhance fault features along paths of "artificial ants" and the paths are assumed to follow faults. Some other authors (Neff et al., 2000; Cohen et al., 2006; Wu and Zhu, 2017) suggest smoothing along fault strikes and dips to enhance fault features by scanning over all possible combinations of fault strikes and dips. Similarly, Hale (2013b) and Wu and Hale (2016a) compute fault likelihood or fault-oriented semblance by smoothing the numerator and denominator of the semblance along fault strikes and dips.

In this paper, we first discuss methods of optimal path (2D) and surface (3D) picking by using the dynamic programming algorithm (e.g., Cormen et al., 2001; Hale, 2013a). By solving for global maximization, the optimal path and surface picking methods are robust to compute smooth and continuous paths or surfaces from highly noisy or discontinuous features. Based on the optimal path and surface-picking method, we then propose a novel method of optimal path or surface voting to efficiently compute a new fault attribute map of voting scores from an input fault attribute image, so that the fault features in the voting score map are much cleaner, sharper, and more continuous than those in the input image.

In the proposed optimal surface-voting method, we first automatically pick sparse seed points from an input fault attribute image, which can be any seismic attribute image that highlights faults with relatively high values. For each seed point, we then efficiently pick an optimal surface patch that passes through the seed point and follows globally maximum fault attribute values. We define all these picked optimal surfaces as voters, and we define voting scores on each surface voter with fault attribute values that are smoothed along the surface voter. Finally, we collect the voting scores of all the surface voters to obtain a voting score map, a new fault attribute image, in which noisy features unrelated to faults are suppressed, whereas the fault features (with high voting scores) are much cleaner and more continuous than those in the input attribute image. With the surface voters, we can also accurately estimate fault strikes and dips by averaging (weighted by voting scores) the orientations of the overlapping surface voters. From the voting score map with clean and continuous fault features, fault surfaces can be extracted by following the fault features along the estimated fault strikes and dips. The computational cost of the optimal surface-voting method depends on the number of seed points, and our parallel implementation is highly efficient. We demonstrate the efficiency and effectiveness of the method by using multiple examples that are complicated by intersecting faults and heavy noise.

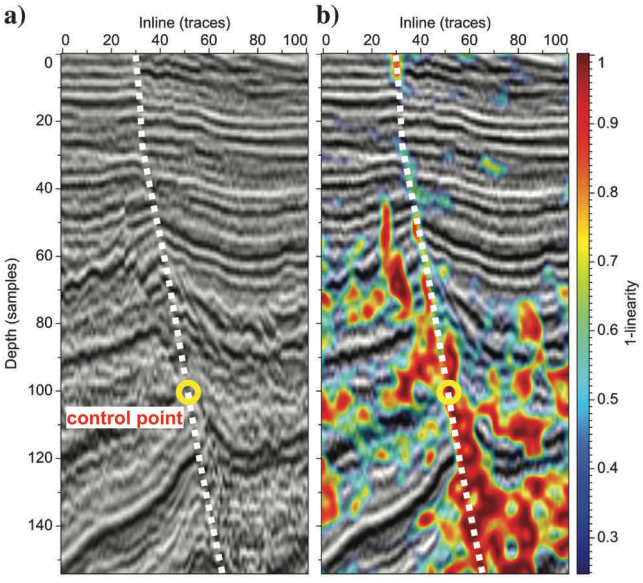

Figure 1. (a) In a 2D seismic image, a fault curve passing through the control point (yellow circle) is manually interpreted. (b) In a fault attribute (1-linearity) image computed from the seismic image, most part of the fault is highlighted out by relatively high values (colored red). However, automatic extraction of the fault curve from this attribute image is not straightforward because the fault features are noisy and discontinuous.

OPTIMAL PATH AND SURFACE PICKING

In a fault attribute image, faults are typically recognized by relatively low (e.g., semblance, Marfurt et al., 1998; coherence, Marfurt et al., 1999) or high values (e.g., fault likelihoods, Hale, 2013b; Wu and Hale, 2016a) and 1-linearity (Hale, 2009; Wu, 2017; shown in Figure 1b). Therefore, we may consider fault picking from an attribute image as a problem of searching for an optimal path (two dimensions) or surface (three dimensions) with globally minimum or maximum values. We use the dynamic programming algorithm (e.g., Cormen et al., 2001; Hale, 2013a) to efficiently pick such optimal paths or surfaces.

Optimal path picking

To explain how to pick the optimal fault passing through globally maximum fault attribute values in Figure 1b, we transpose the fault attribute image (Figure 1b) into a new space (Figure 2a), where the vertical axis represents the inline direction and the horizontal axis represents depth. In this new image in Figure 2a, the fault passing through the control point (the yellow circle) is an optimal path with maximum fault attribute values from left to right.

If we label the vertical and horizontal axes with j ($j = 0, 1, \ldots, M-1$) and i ($i = 0, 1, \ldots, N-1$), respectively, as in Figure 2b, then, a path from left to right can be represented as $j[i]$, and we solve the following maximization problem to find the optimal path:

$$\arg\max_{j[i]} \sum_{i=1}^{N} g[i, j[i]], \tag{1}$$

where $g[i, j]$ represents the transposed fault attribute image shown in Figure 2b.

In practice, a fault is typically a vertically dipping curve as denoted by the dashed white curve in Figure 1. Therefore, in the transposed image in Figure 2, a fault should be a slightly dipping curve with small slopes. To pick such an optimal path with small and slowly varying slopes, we solve the following constrained maximization problem:

$$\arg\max_{j[i]} \sum_{i=1}^{N} g[i, j[i]],$$
$$\text{subject to } |j[i+1] - j[i]| \le \varepsilon (0 < \varepsilon \le 1), \quad (2)$$

where the second equation represents the slope constraints and we use a small upper slope bound $\varepsilon = 0.25$. We solve the above constrained maximization and find the optimal path by using the dynamic programming algorithm with three steps of nonlinear smoothing, forward accumulation, and backward tracking, as discussed by Hale (2013a).

Nonlinear smoothing

The first step of smoothing is applied to enhance the features related to the fault or path to be picked and to attenuate noise features unrelated to the fault. As discussed by Hale (2013a), such a smoothing filter is implemented by applying nonlinear accumulation to the input fault attribute image in the forward (right) and reverse (left) directions.

The forward accumulation (from left to right) of the input image $g[i, j]$ with the slope constraints $|j[i+1] - j[i]| \le \varepsilon$ can be implemented as follows:

$$f[0, j] = g[0, j],$$
$$f[i, j] = g[i, j] + \max \begin{cases} f[i-d, j-1] + \sum_{k=i-d+1}^{i-1} g[k, j-1] \\ f[i-1, j] \\ f[i-d, j+1] + \sum_{k=i-d+1}^{i-1} g[k, j+1] \end{cases},$$
$$\text{for } i = 1, 2, \ldots, N-1, \quad (3)$$

where $d = \lfloor 1/\varepsilon \rfloor$ is the integer nearest to $1/\varepsilon$.The term $g[i, j]$ is an input fault attribute image, whereas $f[i, j]$ is the accumulated image in the forward direction. Such a forward accumulation can be considered as a one-sided smoothing filter, which is nonlinear because of the min function used in the above equation.

Similarly, the backward accumulation (from right to left) of the input image $g[i, j]$ with the slope constraints $|j[i+1] - j[i]| \le \varepsilon$ can be implemented as follows:

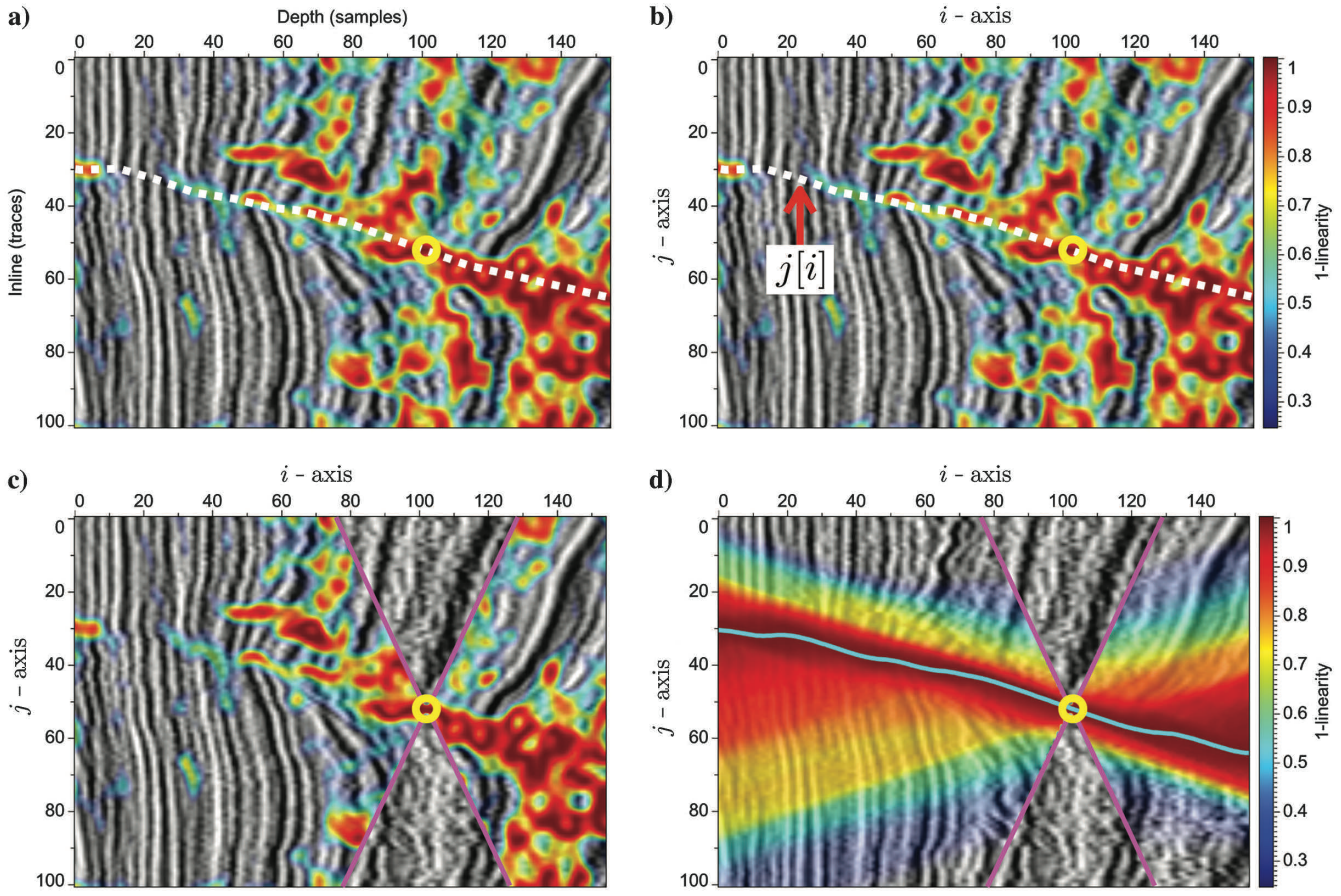

Figure 2. To pick the fault curve in Figure 1b, we first transpose the seismic fault attribute image (Figure 1b) into a new space shown in (a), where the vertical and horizontal axes are inline and depth, respectively. The fault curve to be picked in this new space is a slightly dipping path extending from left to right as denoted by the dashed white curve in (b). Picking such a fault curve from the transposed fault attribute image can be considered as a problem of searching for an optimal path that passes through the control point (yellow cycle) and follows globally maximum attribute values from left to right. To enforce the maximum path passing through the control point, we set zeros (minimum values) in areas above and below the point and we set ones (maximum values) near the control point. To compute an optimal path following globally maximum values, we use a modified dynamic programming algorithm in which we first apply a nonlinear smoothing filter to (c) the original fault attribute image to obtain (d) a smoothed image with more continuous fault features and then pick the maximum path (cyan curve in [d]) from the smoothed image.

$$b[N-1,j]=g[N-1,j],$$
$$b[i,j]=g[i,j]+\max\begin{cases} b[i+d,j-1]+\sum_{k=i+1}^{i+d-1} g[k,j-1] \\ b[i+1,j] \\ b[i+d,j+1]+\sum_{k=i+1}^{i+d-1} g[k,j+1] \end{cases},$$
$$\text{for } i=N-2,N-3,\ldots,0, \tag{4}$$

where, again, $d=\lfloor 1/\varepsilon \rfloor$ is the integer nearest to $1/\varepsilon$, and $b[i,j]$ is the accumulated image in the backward direction. Such a backward accumulation can be considered as a one-sided smoothing filter applied to the input image in the reverse direction (from right to left).

The two-sided nonlinear smoothing is then defined as the combination of forward and backward accumulations:

$$s[i,j]=f[i,j]+b[i,j]-g[i,j], \tag{5}$$

where we subtract $g[i,j]$ because this value is counted in $f[i,j]$ and $b[i,j]$. With slope-constrained forward and backward accumulations, the low slope constraints are imposed in the nonlinear smoothing to yield smooth and continuous features in the smoothed image.

Figure 2d shows the normalized output image computed by applying the nonlinear smoothing (equation 5) to the input image in Figure 2c. After the smoothing, the noise features are attenuated, whereas image features related to the fault are much more obvious and continuous in middle of the output image, in which the fault or maximum path is much easier to pick.

Note that we have set zero values (minimum values) in the areas above and below the control point and ones (maximum values) near the control point (yellow circle) in the image before (Figure 2c) and after (Figure 2d) smoothing. By doing this, we can ensure that the next step of maximum path picking will find the path passing through the control point.

Forward accumulation and backtracking

From the smoothed image (Figure 2d), we pick the maximum path with the following two steps of forward accumulation and backtracking. The forward accumulation is exactly the same as in equation 3, but the input is the smoothed image $s[i,j]$:

$$a[0,j]=s[0,j],$$
$$a[i,j]=s[i,j]+\max\begin{cases} a[i-d,j-1]+\sum_{k=i-d+1}^{i-1} s[k,j-1] \\ a[i-1,j] \\ a[i-d,j+1]+\sum_{k=i-d+1}^{i-1} s[k,j+1] \end{cases},$$
$$\text{for } i=1,2,\ldots,N-1. \tag{6}$$

From the forward accumulated image $a[i,j]$, the maximum path $j[i]$ is computed by backtracking, which begins with the last position and ending with the first position of the path:

$$j[N-1]=\arg\max_j a[N-1,j],$$
$$l=j[i],$$
$$j[i-1]=\arg\max_{l-1,l,l+1}\begin{cases} a[i-d,l-1]+\sum_{k=i-d+1}^{i-1} s[k,l-1] \\ a[i-1,l] \\ a[i-d,l+1]+\sum_{k=i-d+1}^{i-1} s[k,l+1] \end{cases},$$
$$\text{for } i=N-1,N-2,\ldots,1. \tag{7}$$

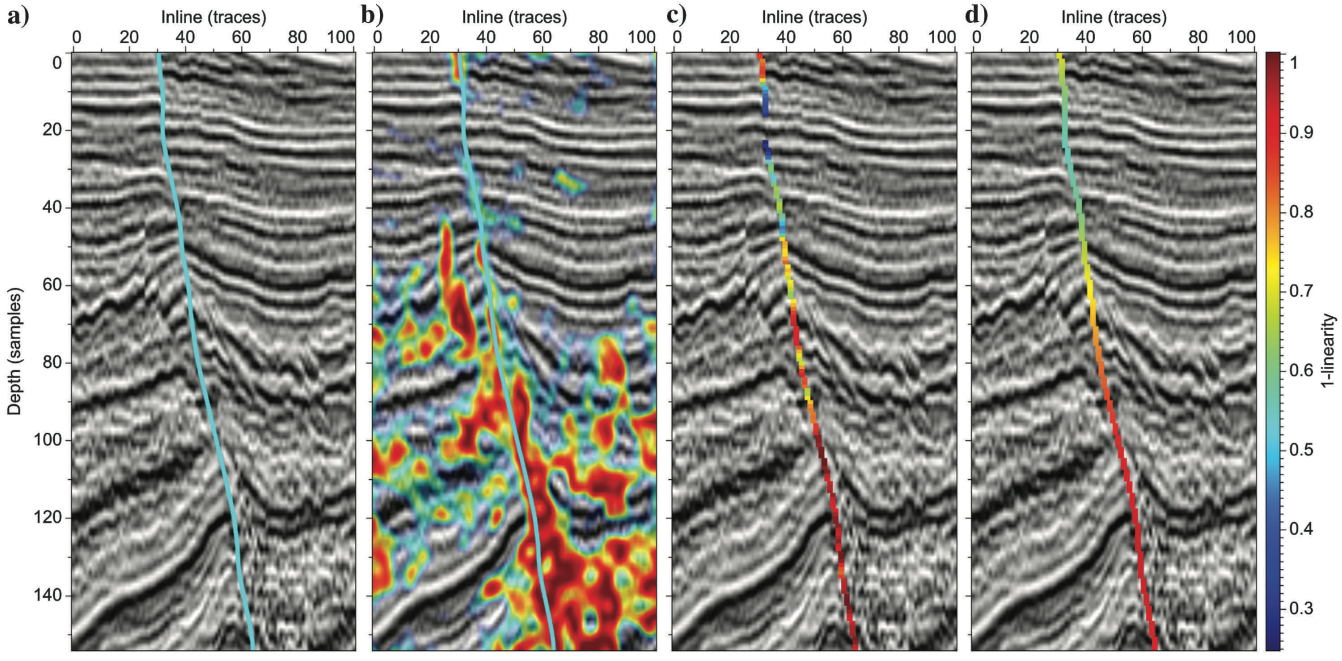

Figure 3. The picked optimal path (Figure 2d) is mapped back to the original space and is displayed with (a) the 2D seismic image and (b) fault attribute image, where the picked path matches with the fault to be extracted. This picked path is further colored by fault attribute values (c) before and (d) after smoothing along the path.

The backtracking step begins with finding the last position of the path by searching (over the vertical index j) for the maximum value of the last vertical trace ($a[N-1,j]$) in the forward accumulated image $a[i,j]$. Then, we recursively find the previous position $j[i-1]$ of the path by comparing the three values $a[i-d,l-1]+\sum_{k=i-d+1}^{i-1} s[k,l-1]$, $a[i-1,j]$, and $a[i-d,l+1]+\sum_{k=i-d+1}^{i-1} s[k,l+1]$, where $l=j[i]$ is the current path position that is already found.

The cyan curve in Figure 2d shows the picked optimal path, which passes through the control point and correctly follows the high fault attribute values in the smoothed fault attribute image (Figure 2d). Note that this picked path is already smoothed by a Gaussian filter. In Figure 3a and 3b, we display the picked path with the original seismic and fault attribute images, in which we observe that the path correctly follows the fault that we want to extract. Figure 3c shows the fault attribute values extracted on the picked path, and these values are discontinuous along the path. Therefore, we smooth the fault attribute values along the path to obtain more continuous values on the path, as shown in Figure 3d.

Optimal surface picking

In a 3D seismic image or fault attribute image, a fault is a locally planar surface oriented by fault strikes and dips. To simplify the 3D fault interpretation, some methods are proposed to image faults within a sequence of 2D image slices (Crawford and Medwedeff, 1999; Dorn et al., 2012) or pick fault curves within 3D images (Pedersen et al., 2002, 2003; Pedersen, 2007, 2011). However, to compute 3D faults or fault attributes that are spatially consistent in fault strike and dip directions, we really need to compute 3D faults as surfaces instead of sets of curves. We use a modified dynamic programming algorithm (Hale, 2013a) to efficiently extract 3D fault surfaces as globally maximum surfaces from a 3D fault attribute image.

Figure 4a shows a small 3D seismic image cube (155 (depth) ×101 (inline) ×101 (crossline) samples) with a fault surface intersecting the seismic sections at the dashed yellow curves. The fault surface is approximately parallel to the depth-crossline plane because the fault strikes and dips are closely parallel to the depth and crossline axes, respectively. Figure 4b shows a seismic attribute image of reflection discontinuities (1-planarity, Hale, 2009; Wu, 2017). This attribute image detects most of the fault positions but also highlights noise such as the thick red areas unrelated to faults. In addition, the fault features in this attribute image cannot be continuously tracked. Therefore, automatic fault surface extraction from such an attribute image is not straightforward.

To explain how to extract the optimal fault surface from the fault attribute image (Figure 4b), we first transpose this image into a new space where the vertical axis is inline and the horizontal axes are depth and crossline, as shown in Figure 5a. In this new space, the fault surface to be extracted is a slightly dipping surface that laterally passes throughout the whole 3D attribute image because the fault surface is approximately parallel to the depth-crossline plane. Therefore, extracting the fault surface in this new space can be considered as a problem of picking a slightly dipping surface that laterally passes through globally maximum attribute values. The control point (yellow circle in Figure 5) indicates the position at which we expect the fault surface to pass through. Note that we have set zero values (the minimum values) in the conical areas above and below the control point, and we set ones (the maximum values) near the control point. By doing this, we are able to impose constraints of the control point to enforce the maximum surface passing through the control point.

Smoothing and picking

Because the transposition of the attribute image does not actually improve the fault features within the image, extracting a smooth and continuous fault surface from this transposed image (Figure 5a) is still not straightforward. Therefore, we apply the nonlinear smoothing discussed in the previous section to the attribute image before picking the fault surface.

In this 3D example, we first apply the slope-constrained nonlinear smoothing (equations 3–5) to each inline-depth slice and obtain a smoothed image shown in Figure 5b. With low slope constraints, the nonlinear smoothing imposes smoothness and continuity on the smoothed attribute features while attenuating noise. After the smoothing, the fault features, especially in the inline-depth slice, are much more obvious and continuous in Figure 5b than those in original transposed image (Figure 5a). With this smoothed image as an input, we further apply the slope-constrained nonlinear smoothing in each inline-crossline slice to obtain a better smoothed image shown in Figure 5c, in which the fault features are smooth and continuous in the inline-depth and inline-crossline slices. Note that all of these smoothed attribute images (Figure 5b and 5c) are normalized so that the image values are in the range of $[0,1]$.

The next step of picking the fault surface in the smoothed image (Figure 5c) is now more straightforward than in the original image in Figure 5a. A fault surface can be extracted as a set of maximum curves from either the 2D inline-depth or the inline-crossline slices of the smoothed image (Figure 5c). These curves will be spatially consistently aligned to form a smooth and continuous fault surface because the smoothness of the fault surface has been imposed by the previous slope-constrained nonlinear smoothing in computing

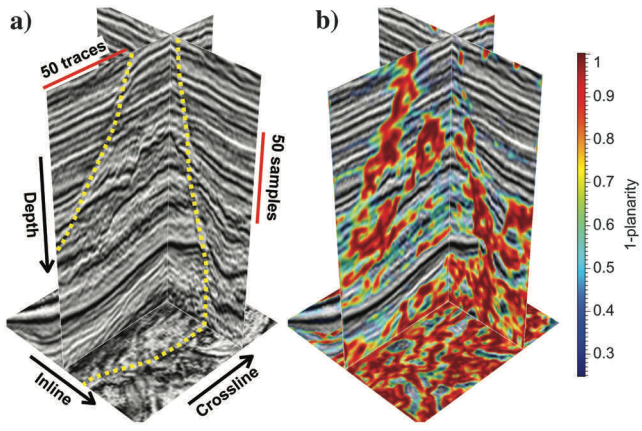

Figure 4. (a) In a small 3D seismic image, a manually interpreted fault surface intersects with the seismic sections at the dashed yellow curves. (b) In a fault attribute (1-planarity) image computed from the seismic image, some parts of the fault are highlighted out by relatively high values (colored red). However, automatic extraction of the fault surface from this attribute image is not straightforward because the fault features are noisy and discontinuous.

the smoothed image (Figure 5c). Each maximum curve in each 2D slice can be picked by the two steps of forward accumulation and backtracking shown in equations 6 and 7, respectively.

The cyan surface in Figure 5d shows the extracted smooth and continuous fault surface, which passes through the control point and correctly follows the high values in the smoothed fault attribute image (Figure 5d). In Figure 6a, we display the extracted fault surface overlaid with the original seismic and fault attribute images. In Figure 6b, the fault surface is colored by the original fault attribute values, which are discontinuous along the surface. Then, we apply a Gaussian filter to smooth the fault attribute values along the surface to obtain more continuous values on the surface, as shown in Figure 6c. The curves (colored by the smoothed attribute values) in Figure 6d show the intersections of the extracted fault surface and the seismic sections. These intersecting fault curves coincide well with the manually interpreted fault positions denoted by the dashed yellow curves in Figure 4a.

SEISMIC FAULT INTERPRETATION

We have discussed automatically extracting a single fault curve (two dimensions) or surface (three dimensions) from a fault attribute image by picking an optimal path or surface using the dynamic programming method. By solving for a global maximization, this method is robust to extract a reliable fault from a fault attribute image with even highly noisy and discontinuous fault features. In picking a single fault curve or surface, we have assumed that there is only one fault corresponding to one control point and the fault is a slightly dipping curve or surface passing throughout the whole fault attribute image. However, these assumptions are not necessarily true in a practical 2D or 3D seismic fault attribute image, which often contains multiple faults with different orientations and spatial extensions. In this section, we will discuss how to adapt the optimal path and surface picking algorithms to enhance a fault attribute image for automatic interpretation of all the faults with different orientations and extensions.

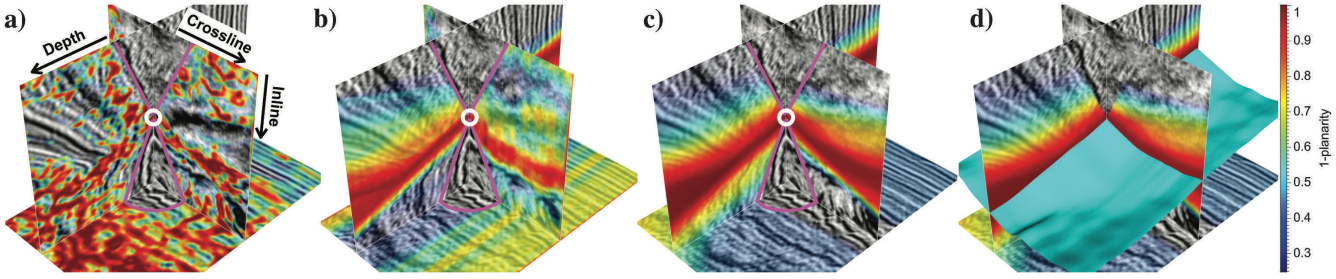

Figure 5. To pick the fault surface from the fault attribute image (Figure 4b), we first transpose the attribute image into a new space shown in (a), where the vertical axis represents the inline and horizontal axes represent the depth and crossline. The fault surface to be picked in this new space is a slightly dipping surface that laterally extends throughout the whole attribute image. Picking such a fault surface from the transposed attribute image can be considered as a problem of searching for an optimal surface that passes through the control point (white circle) and follows globally maximum attribute values. To enforce the maximum surface passing through the control point, we set zeros (minimum values) in the conical areas above and below the point and set ones (maximum values) near the control point. To compute an optimal surface following globally maximum values, we use a modified dynamic programming algorithm, in which we first smooth every depth-inline slice of (b) the attribute image and further smooth every crossline-inline slice to obtain (c) a smoothed image with more continuous fault features. (d) The maximum surface is finally extracted from the smoothed attribute image.

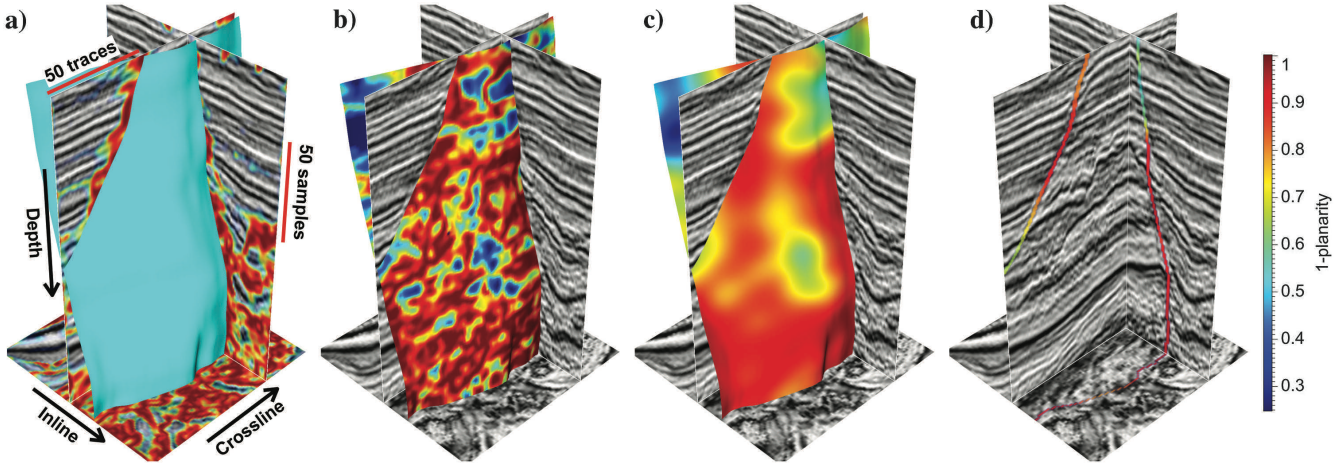

Figure 6. The picked optimal surface (Figure 5d) is mapped back to the original space and is displayed with (a) the overlaid seismic and fault attribute images. This picked surface is further colored by fault attribute values (b) before and (c) after smoothing along the surface. The intersection curves (colored by smoothed attribute values in [d]) of the picked surface and the seismic sections match well with the manually interpreted fault curves in Figure 4a.

2D faults with optimal path voting

Figure 7a shows a time slice of a 3D seismic image, and this slice is cut by many faults. Figure 7b shows the same time slice of the corresponding 3D seismic planarity (Hale, 2009; Wu, 2017) image, which highlights reflection discontinuities with relatively low values. Figure 7c shows a fault attribute map computed by 1-planarity, where most of the faults and other reflection discontinuities are indicated by relatively high values (colored white). Although this fault attribute map highlights most of the fault positions, automatic tracking faults from such an attribute map is still challenging because the fault features cannot be continuously tracked and noise unrelated to faults are also apparent in the map. We propose the following optimal path-voting method with three steps to suppress the noisy features, while enhancing the fault features in a 2D fault attribute image.

Step 1: Fault orientation scanning

In the dynamic programming of optimal path picking algorithm, an optimal path can be more reliably picked if it is dipping with smaller slopes. Therefore, the first step of our optimal path-voting method is to approximately estimate fault orientations, so that the fault curves can be reliably picked along the estimated fault orientations because the faults will be slightly dipping along the fault orientations. The fault orientations do not need to be perfectly estimated at this step; we only need to find the general fault trend, along which we can pick the true fault curve and further estimate more accurate fault orientations from the picked curve.

The structure-tensor method (Van Vliet and Verbeek, 1995; Weickert, 1997; Fehmers and Höcker, 2003; Wu and Janson, 2017) is commonly used to estimate orientations of image features, but it is not a good choice to estimate orientations of faults in a fault attribute image (Figure 7c), in which most of the image values are nearly zeros and multi-oriented intersecting faults are apparent. Some authors (e.g., Tingdahl and Hemstra, 2003; Barnes, 2006) propose to use principal component analysis-based methods to better estimate fault orientations. In this paper, we use a matched filtering or scanning method (e.g., Marfurt, 2006; Hale, 2013b; Wu and Zhu, 2017) to estimate fault orientations. In this method, we smooth the fault attribute image along all possible fault orientations to find the maximum smoothing response at each image sample. The orientation that yields the maximum response at each image sample is recorded as the estimated dominant fault orientation at that sample. We implement the scanning method with an efficient scheme discussed by Hale (2013b).

The computational cost of the scanning method depends on the number of fault orientations that are scanned. Because we do not need a perfect estimation of fault orientations at this step, we can scan for only sparsely spaced fault angles. For example, in a time slice of fault map in Figure 7c, the fault strikes can be oriented in a range between 0° and 180°. To save time, we scan for only nine different angles, one for every 20°, as denoted by the yellow and magenta lines in Figure 7c. The magenta line represents the dominant fault orientation estimated at the center point denoted by the red dot.

Step 2: Optimal path picking

With the estimated fault orientations, our next step is to automatically compute seed points and pick the optimal fault path at each seed point along the fault orientation estimated at that point.

In computing the seed points, we first compute a thinned fault attribute image by applying non-maximum suppression to the original fault attribute image (Figure 7c). In this non-maximum suppression, we keep only the attribute values on the ridges (in directions perpendicular to the estimated fault orientations) of the fault attribute image and we set zeros elsewhere to obtain a thinned attribute image. Then, we select seed candidates from the thinned attribute image by collecting all the image samples with attribute values that are larger than some threshold (0.3 for the example in Figure 7d). Finally, we check in the true seeds from all the candidate points in the order from the one with highest attribute value to the one with the lowest value. In checking in the seeds, we compute the distances between the current candidate to all the previously checked-in seeds and check the current candidate as a true seed only if the minimum distance is larger than some predefined radius ($r = 4$ for the example in Figure 7d). By doing this, we can greatly reduce the number of seed points and therefore save computational time for the followed optimal path picking at the seed points. A smaller radius will yield more seed points, which would be better for the next step of voting, but this requires more computational time of computing the optimal voting paths. From our experience, the radius $r = 4$ is suggested for most cases. The white dots in Figure 7d show the computed seed points, which are sparsely spaced, and the minimum distance of the two closest seeds is larger than the predefined radius $r = 4$.

For each seed point, we define a local rectangular window that is centered at the seed and is aligned in the $u - v$ space, as denoted by the yellow box in Figure 7d. The u- and v-axes, respectively, are perpendicular and parallel to the estimated fault orientation (denoted by the magenta line in Figure 7c). Because the fault to be picked is expected to mostly extend along the v-axis, the rectangular window is typically defined wider in the v direction than in the u-direction. We use the seed as a control point in the dynamic programming algorithm to pick the optimal path from the original fault attribute patch located within the rectangular window. The red curve in Figure 7d shows the optimal fault path picked within the local window, which generally extends along the v-axis as expected.

Figure 8a shows some local paths picked at a subset of the seed points. Most of these picked paths correctly follow the fault attribute features, and some of them have been overlaid with each other to visually form longer faults. Figure 8b shows all the local paths picked at all the seed points. Most of these picked paths are visually thicker because these paths are traversed by multiple times, which typically indicate that these paths are likely to follow true faults. Those thinner paths, which are traversed by fewer times or only once, are probably picked at noisy seed points and are less likely to follow true faults. Based on these observations, we consider each path as a voter and we collect the voting scores of all these voters to obtain a voting score map, where the true fault features are voted in and the noisy features are voted out.

Step 3: Optimal path voting

For each voting path in Figure 8b, we compute the voting scores on the path by smoothing (as in Figure 3d) the original fault attribute values along the path. Because the attribute values (Figure 7c) indicate the fault probabilities or likelihoods, the voting paths with higher smoothed attribute values will provide higher scores in the followed voting process to compute a voting score map.

We represent any kth voting path parametrically as $\mathbf{v}_k(p) = (i_k(p), j_k(p))$, where $p = 0, 1, \ldots, N-1$. The voting scores $s_k(p)$ on the $_k$th voting path are then defined as follows:

$$s_k(p) = \langle g(\mathbf{v}_k(p)) \rangle, \tag{8}$$

where $g(i, j)$ represents the original 2D fault attribute image (Figure 7c) and $g(\mathbf{v}_k(p))$ denotes the attribute values on the path. $\langle \cdot \rangle$

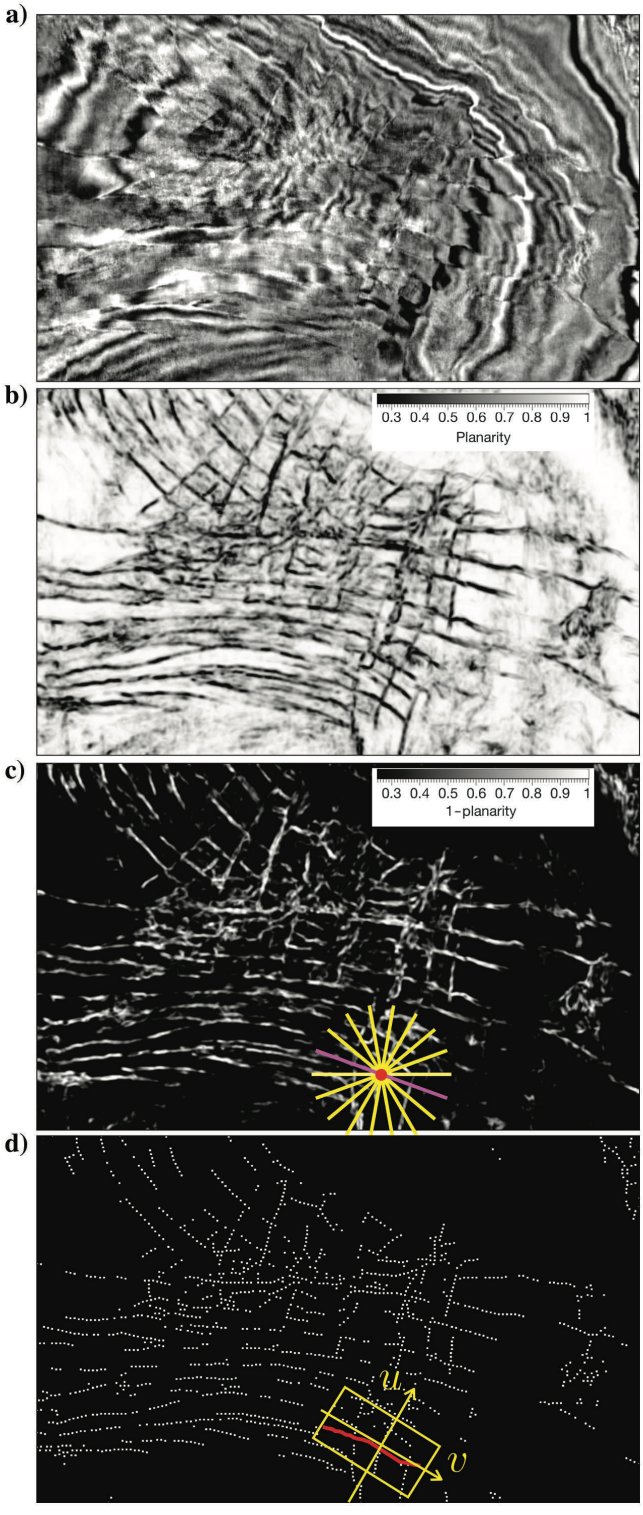

Figure 7. (a) A time slice of a 3D seismic image. (b) The corresponding time slice of seismic planarity. (c) A fault map computed by 1-planarity. (d) An optimal path, passing through high fault energies (1-planarity), is automatically picked in the u-v space.

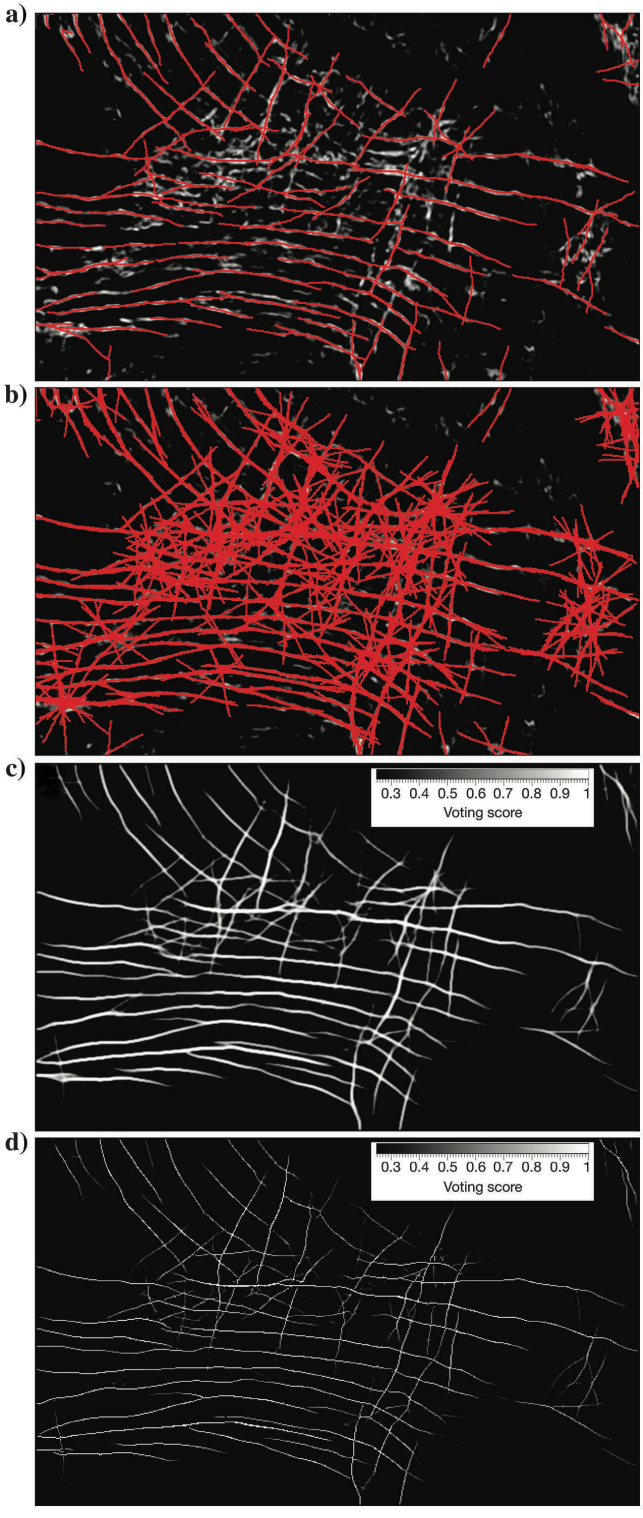

Figure 8. Optimal paths are automatically picked at (a) a subset of the seed points and (b) at all seed points. (b) All the picked paths are combined to compute for (c) a voting score map and a corresponding thinned map, where the noises are suppressed and fault features are enhanced.

represents Gaussian smoothing of the attribute values along the path.

The voting process of computing a voting score map $m(i,j)$ is to collect all the voting scores on all the M voting paths

$$m(\mathbf{v}_k(p)) += s_k(p), \quad \text{for } k = 0, 1, 2, \ldots, M-1. \quad (9)$$

In this way, the paths with higher voting scores and that are traversed by multiple times will generate higher values in the accumulated voting score map $m(i,j)$, whereas the noisy paths with lower voting scores and that are traversed by fewer times will be suppressed in the accumulated map. We further normalize the voting score map by shifting and scaling as $[m(i,j) - m_{\min}]/m_{\max}$, where $m_{\min}$ and $m_{\max}$ represent the minimum and maximum values of the accumulated voting score map, respectively.

Figure 8c shows the normalized voting score map computed from the voting paths in Figure 8b. In this voting score map, the noisy paths (the thin paths) have been suppressed whereas the paths following true faults are preserved and enhanced. Compared with the original fault attribute image in Figure 7c, the noisy features are removed, whereas the fault features are more obvious and continuous in this voting score map, where the fault curves are much easier to track. Figure 8d shows the thinned voting score map computed by applying nonmaximum suppression to the above voting score map (Figure 8c).

Using the voting paths in Figure 8b, we are also able to accurately estimate fault orientations (strikes) at the same time while computing the voting score map. We estimate fault strikes by computing a weighted averaging (weighted by voting scores) of strikes at overlapping voting paths. The estimated map of fault strikes is not shown here, but it would be helpful for fault population analysis and automatically picking fault curves from the voting score map.

More 2D examples

To demonstrate the effectiveness and efficiency of the optimal path-voting method in enhancing 2D fault attribute images, we apply this method to two more examples in Figures 9 and 10 and compare the voting score maps with fault likelihood images (Hale, 2013b; Wu and Hale, 2016a).

Figure 9a shows a simple 2D example with clean reflections and obvious conjugate faults. However, these faults are closely aligned with each other, which make the fault likelihood scanning method (Hale, 2013b; Wu and Hale, 2016a) fail to correctly detect the faults. The smoothing in the fault likelihood method is helpful to bridge fault gaps to compute continuous fault features, but it also generates fake fault features across closely aligned faults, as shown in Figure 9b. Figure 9c shows another fault attribute image computed by 1-linearity, which contains fewer continuous fault features and more noisy features (especially these thick red areas) unrelated to faults. To further enhance this fault attribute image (Figure 9c), we apply the optimal path-voting method to this image and compute a voting score map shown in Figure 9d, in which the fault features are much more continuous, cleaner, and sharper, whereas the noisy features are suppressed.

Figure 10a shows a more complicated 2D example, in which noise is apparent and the reflections and faults are not clearly imaged. Again, the fault likelihood scanning method fails to correctly detect the faults within this example as shown in the fault likelihood

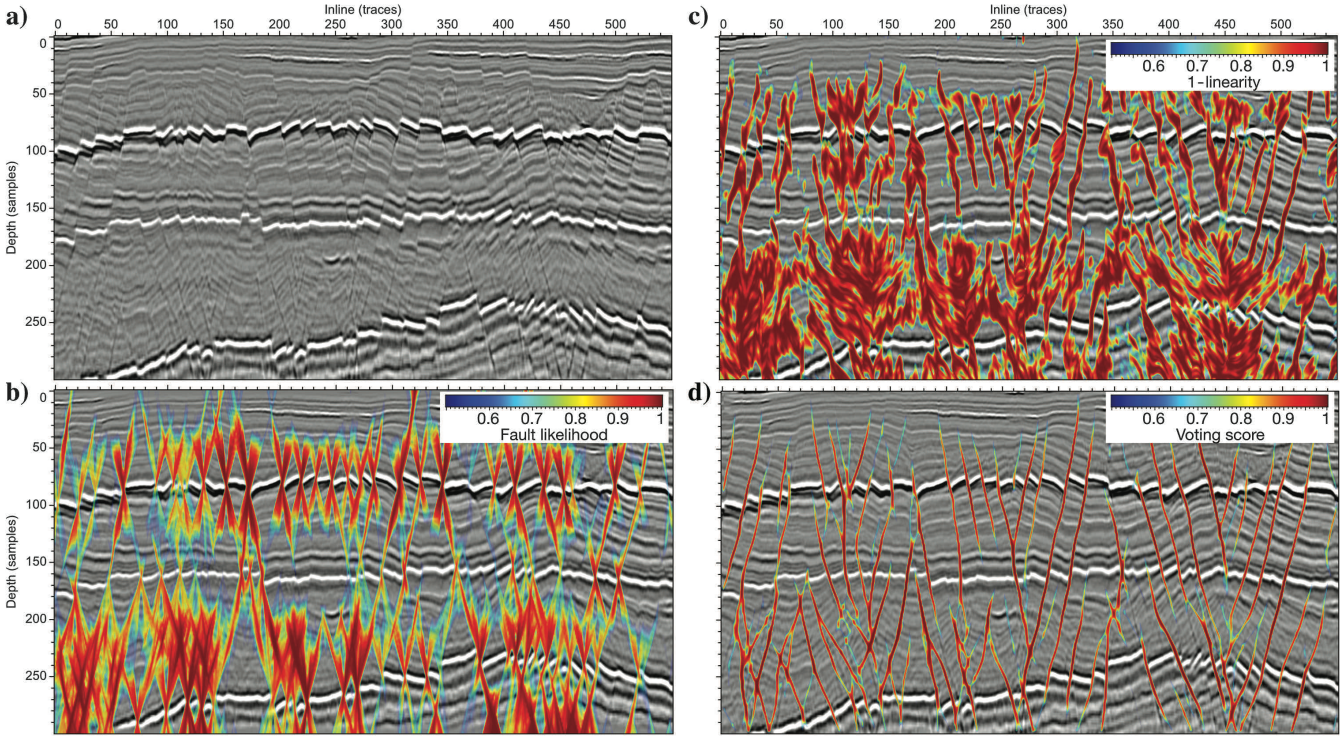

Figure 9. (a) A 2D seismic image is displayed with (b) a fault likelihood image, where the fault features are continuous, but a lot of fake fault features are generated. (c) Another fault attribute image computed by 1-linearity can detect some of the faults, but the fault features are noisy and discontinuous. (c) Using the fault attribute image as an input, the proposed optimal path-voting method can generate a voting score map that is a better fault attribute image with cleaner, sharper, and continuous fault features.

images before (Figure 10b) and after (Figure 10c) thinning. Figure 10d shows the seismic attribute of 1-linearity, where the features are noisy and faults are difficult to track from this attribute image. After applying the optimal path-voting method to this attribute image (Figure 10d), we obtain the voting score map shown in Figure 10e, where, again, the fault features are much more continuous and cleaner whereas the noise features are removed. Figure 10f shows a thinned voting score map, where most of the faults are correctly detected. The whole process of computing the voting paths and the voting score map can be parallelly implemented, and our implementation takes less than 1 s for each of these two examples by using an four-core computer.

3D faults with optimal surface voting

To enhance a 3D fault attribute image (Figure 11b), we propose an optimal surface-voting method with similar steps of (1) scanning for fault orientations (dips and strikes in 3D), (2) picking optimal voting surfaces, and (3) optimal surface voting to compute a voting score map. The fault features in the voting score map will be cleaner and more continuous than the original attribute image, and constructing fault surfaces from the score map will be more straightforward.

Step 1: Fault dip and strike scanning

In 2D cases, we need to scan for only fault strikes in a 2D time (or depth) slice or apparent fault dips in a 2D inline (or crossline) section. In 3D cases, faults are oriented by fault dips and strikes, and we need to estimate both of them to determine the fault orientations.

We estimate the fault dips and strikes by using the fault orientation scanning algorithm discussed by Hale (2013b) and Wu and Zhu (2017). Using this method, we smooth the fault attribute image (Figure 11b) along different planes defined by all possible combinations of strike and dip angles to find the combinations of fault strikes and dips that yield the maximum smoothing values. By doing this, we assume that a fault surface is locally planar and that the 2D smoothing filter (oriented by a combination of strike and dip angles) will generate the maximum smoothing response if the smoothing plane coincides with a local fault plane.

The computational cost of this method depends on the number of strike and dip combinations that are scanned. Again, because the fault orientations do not need to be perfectly estimated at this step, we typically need to scan over only a limited number of strike and dip combinations to find an approximate estimation of fault orientations. Assuming the possible fault strikes and dips are in the ranges of $[0^\circ, 180^\circ]$ and $[65^\circ, 90^\circ] \cup [-90^\circ, -65^\circ]$, respectively, then,

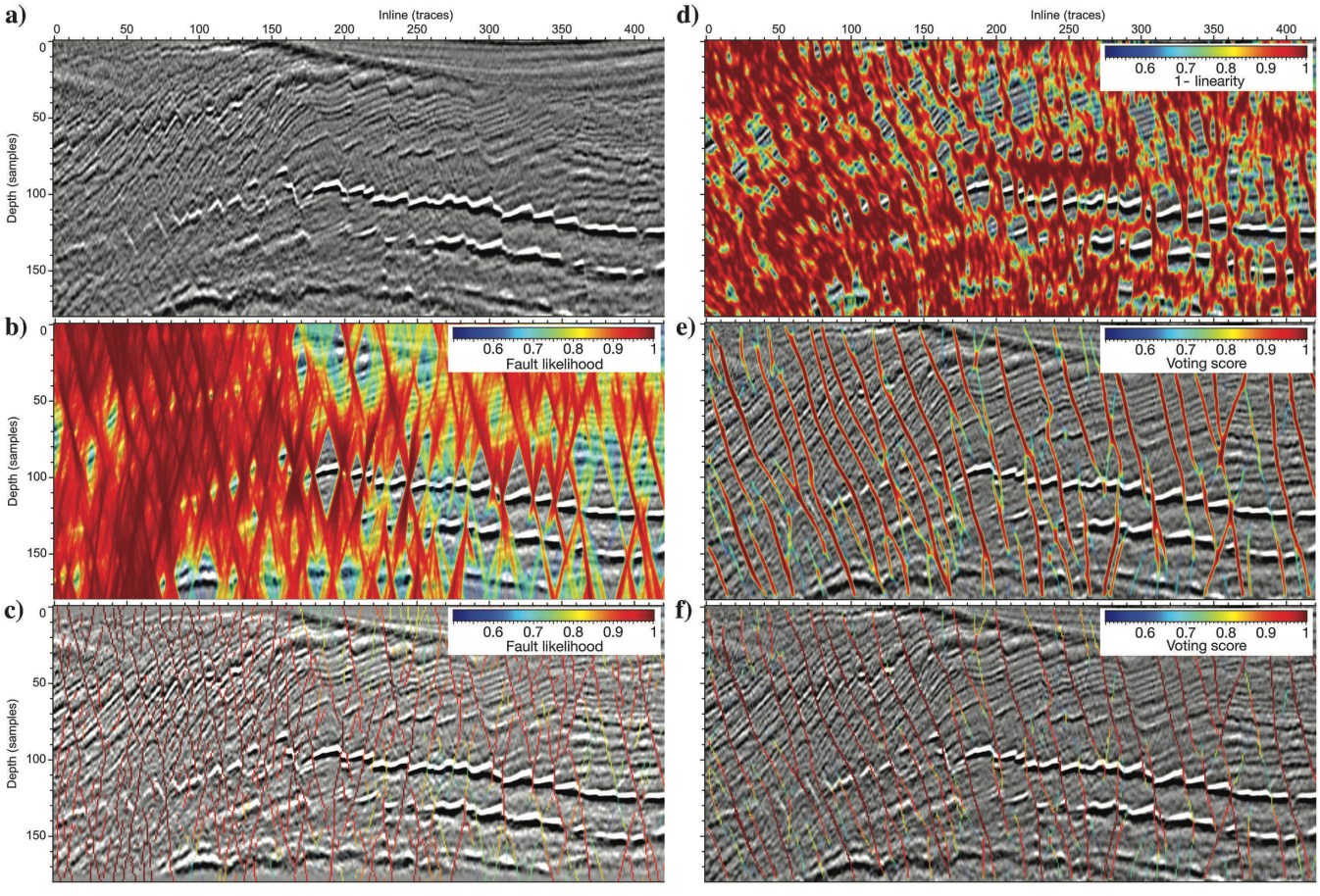

Figure 10. (a) A 2D seismic image is displayed with a fault likelihood image (b) before and (c) after thinning, where a lot of fake faults are detected. (d) Another attribute image computed by 1-linearity can detect some of the faults, but the fault features are noisy and discontinuous. (d) Using the fault attribute image as an input, the proposed optimal path-voting method can generate a voting score map (before [e] and after [f] thinning) that is a better fault attribute image with cleaner, sharper, and continuous fault features.

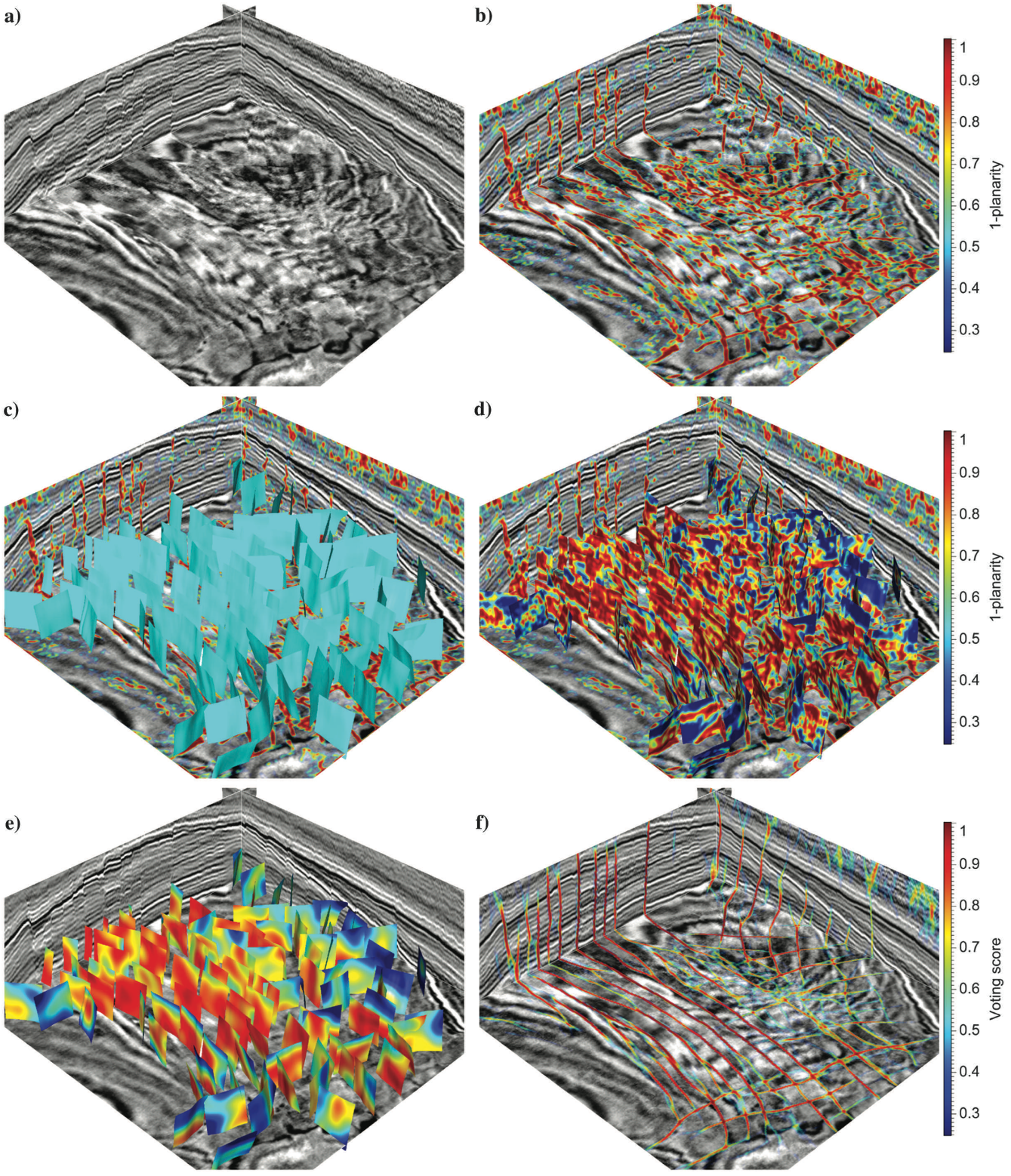

Figure 11. (a) A 3D seismic image. (b) A fault attribute (1-planarity) image computed from the seismic image. (c) A subset of voting surfaces extracted from (b) the fault attribute image. (d) Voting surfaces are colored by the fault attribute values. (e) Voting scores of the voting surfaces are computed by smoothing the fault attribute values on the surfaces. (f) A 3D voting score map is computed by stacking the scores of all the voting surfaces.

we often scan over only 18×10 combinations of fault strike and dip angles; that is, we scan for one strike angle for every 10° and one dip angle for every 5°. The estimated approximate fault orientations provide the general trend of fault extension for guiding the next step of picking optimal voting surfaces. More accurate fault orientations can be estimated by using the picked optimal voting surfaces.

Step 2: Optimal surface picking

In the second step of computing optimal voting surfaces, we first automatically find seed points using the same way of choosing 2D seed points. For each seed point, we then define a local 3D box window that is centered at the point and spatially aligned with the fault strike and dip estimated at the point. Similar to the 2D yellow box in Figure 7d, this 3D box window is also defined to extend more widely in the estimated fault strike and dip directions than in the direction perpendicular to the fault plane of the strike and dip. Within this 3D box oriented by the approximately estimated strike and dip, the true fault passing through the seed point should be a surface patch that is slightly dipping along the approximate strike and dip. We extract such a surface patch from the fault attribute cube within the 3D box by using the optimal surface picking algorithm, as illustrated in Figure 5.

In this way, we extract optimal surface patches at all the seed points. The cyan surface patches in Figure 11c show only some optimal surfaces extracted at a subset of seed points. These surface patches are colored by the original fault attribute values in Figure 11d. The surface patches colored red (high attribute values) are more likely to correctly follow true faults than those colored blue (low attribute values). Because the attribute values on the surface patches are discontinuous, we apply a Gaussian smoothing filter to the values on each of the surfaces, as shown in Figure 12e. These smoothed values (on the surface patches) are defined as voting scores for the next step of computing a 3D voting score map. As shown in Figure 11e, the scores can be spatially varying on each voting surface and the higher scores (red) will contribute more in computing the voting score map.

Step 3: Optimal surface voting

In this step, we define the picked surface patches as voting surfaces (such as those in Figure 11e) and we define the smoothed attribute values (on the surface patches) as voting scores. In computing a 3D voting score map, we collect voting scores on all the voting surfaces as what we did for computing a 2D score map in equation 9. We also normalize the voting score map to obtain a final map with values in the range of $[0, 1]$ as shown in Figure 11f. The fault features in this voting score map are much cleaner, sharper, and continuous than those in the original fault attribute image (Figure 11b). Note that no thinning processing is applied to the voting score map, but the fault features in this core map is already very sharp to clearly resolve the detected faults.

With the optimal voting surfaces, we can also vote for fault dips and strikes. We first estimate fault dips and strikes for each voting surface patch. Then, we compute a weighted average (weighted by voting scores) of the dips and strikes on overlapping voting surface patches to obtain 3D maps of fault dips and strikes. The estimated fault dip and strike maps are not shown here, but they are used in the next step to construct fault surfaces.

Construct fault surfaces

Constructing fault surfaces from the voting score map (Figure 11f) is more straightforward than the original fault attribute image (Figure 11b). The voted fault dip and strike maps are also helpful for tracking fault surfaces.

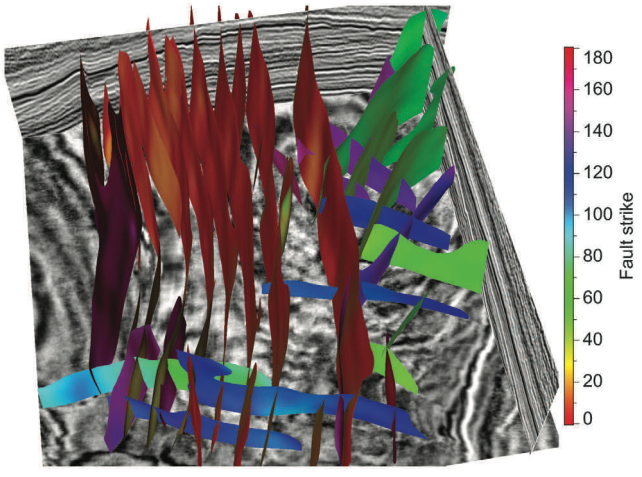

Figure 12. Fault surfaces (colored by fault strikes) are automatically extracted from the voting score map (Figure 11f).

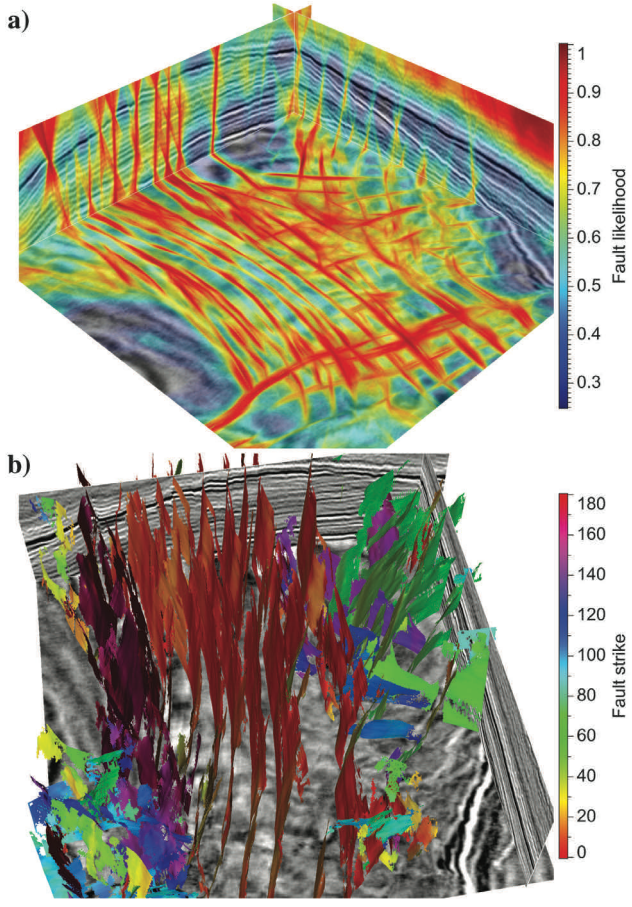

Figure 13. As a comparison, (a) a fault likelihood image is computed from the same 3D seismic image (Figure 11a) to automatically construct the fault surfaces colored by fault strikes in (b).

With the three maps of voting scores, fault dips, and strikes, we automatically construct the fault surfaces shown in Figure 12 by using a surface growing method similar to the one discussed by Wu and Hale (2016a). In this method, we first compute oriented fault samples (Wu and Hale, 2016a) from the three maps of voting scores, fault dips, and strikes. Then, we grow fault surfaces by linking nearby fault samples with consistent scores, dips, and strikes. Different from the method discussed by Wu and Hale (2016a), we typically do not need to interpolate fault samples to fill holes to construct complete fault surfaces because the fault features are often continuous in the voting score map. As shown in Figure 12, the

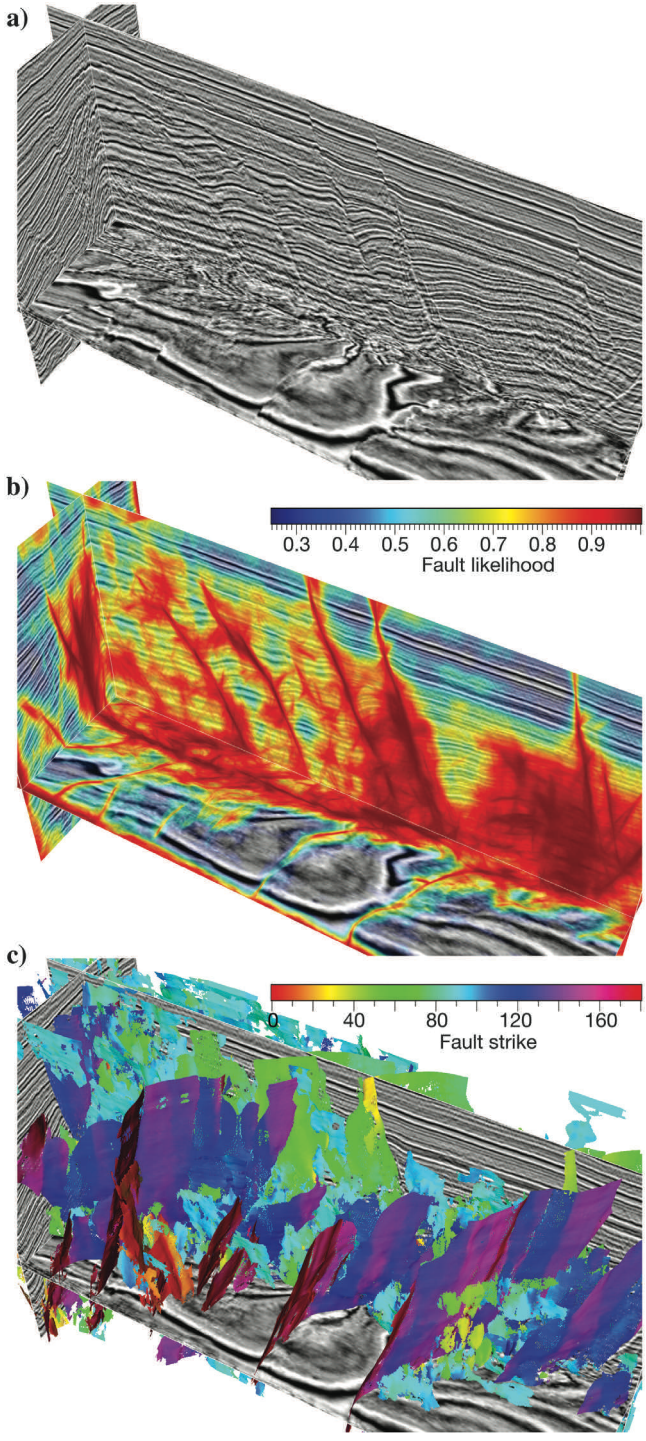

Figure 14. (a) A 3D seismic image is displayed with (b) a fault likelihood image and (c) fault surfaces automatically extracted from the likelihood image. The fault surfaces are colored by fault strikes.

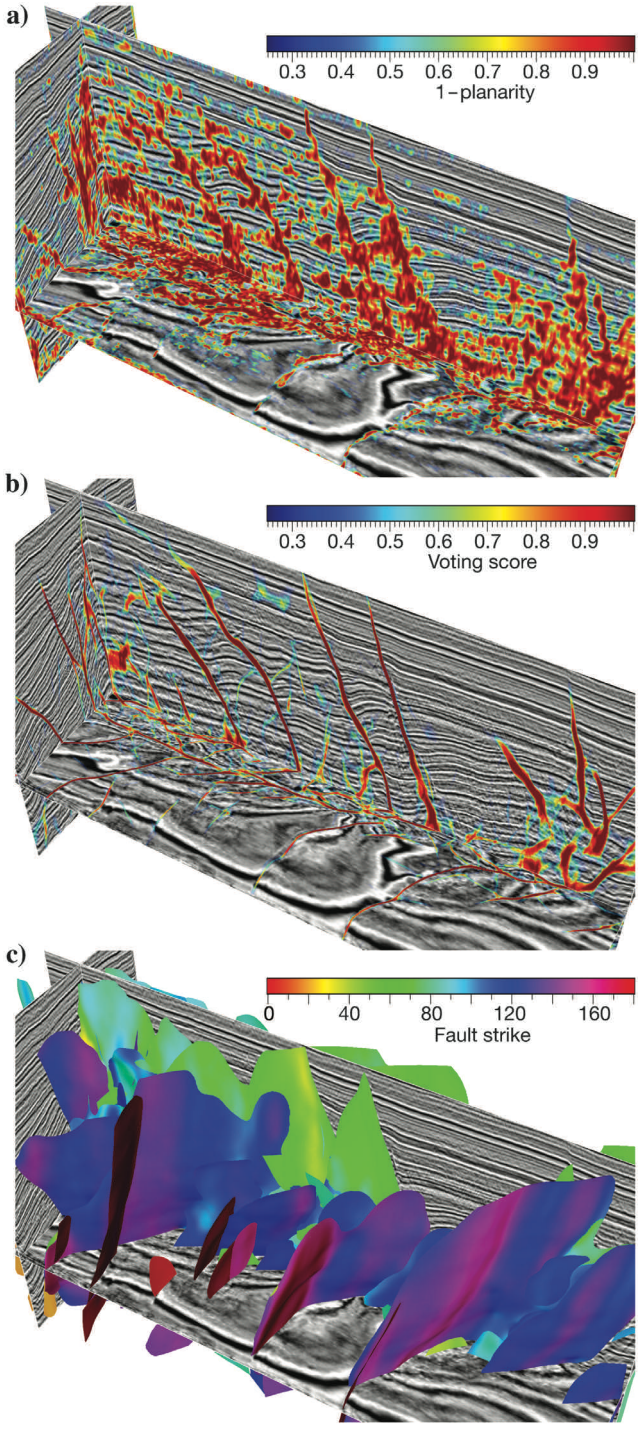

Figure 15. (a) A 3D seismic image is displayed with (b) a voting score map and (c) fault surfaces automatically extracted from the score map. The fault surfaces are colored by fault strikes.

automatically extracted fault surfaces, colored by strikes, correctly follow the fault positions apparent in the 3D background seismic image and each of the surface is complete (no holes are observed).

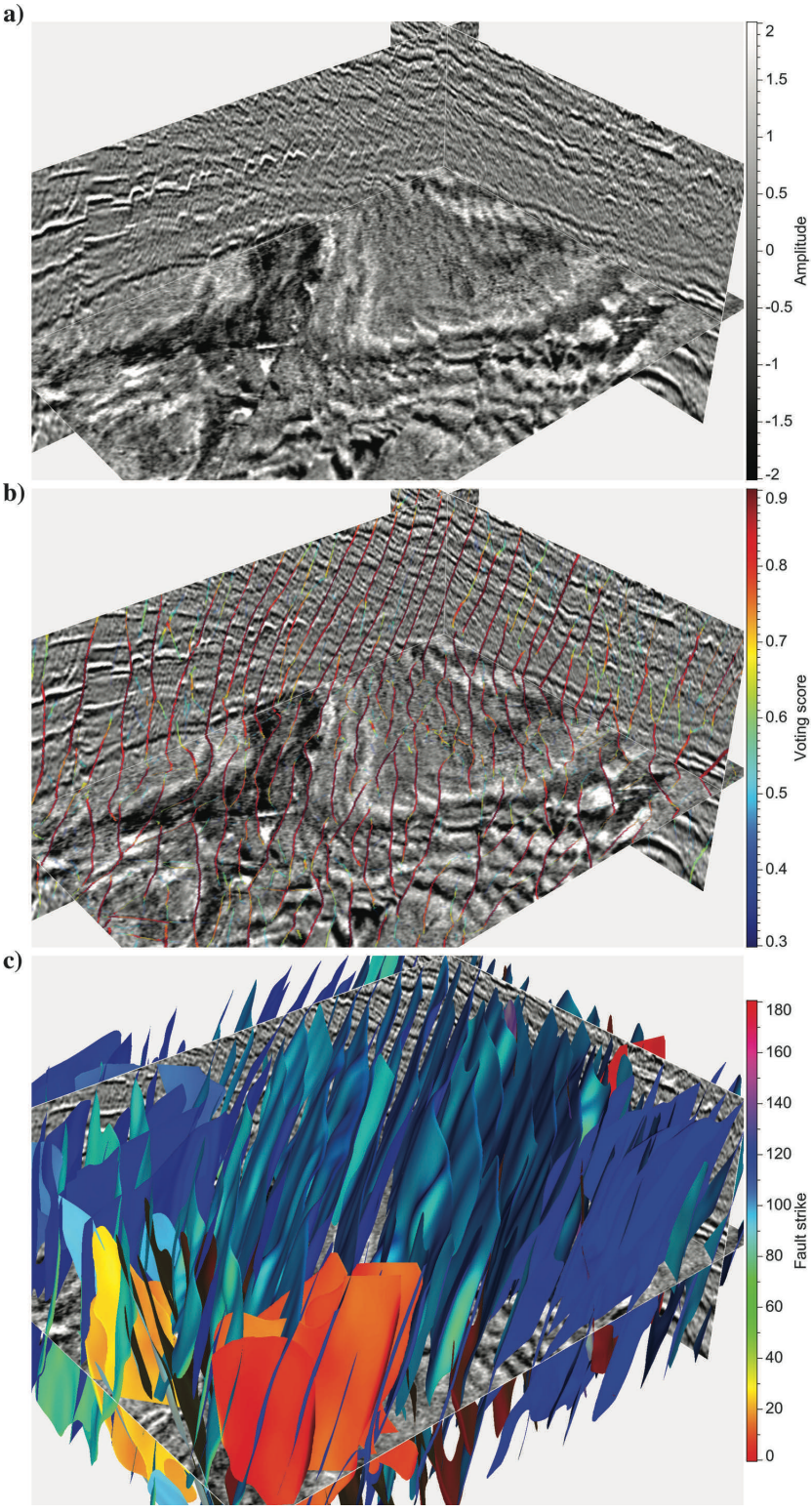

Figure 16. (a) A 3D view of a seismic image displayed with (b) a thinned voting score map and (c) fault surfaces automatically extracted from the score map. The fault surfaces are colored by fault strikes.

As a comparison, we also computed a fault likelihood image and constructed fault surfaces (Wu and Hale, 2016a) for this example, as shown in Figure 13a and 13b, respectively. Although most of the faults are highlighted in the fault likelihood image (Figure 13a), the fault features are more noisy, thicker, and less continuous than those in the voting score map (Figure 11f). The constructed fault surfaces (Figure 13b) are also noisier and less complete than those (Figure 12) constructed from the voting score map.

High efficiency is another advantage of the optimal surface-voting method. One reason for the high efficiency is that the computational cost of this method depends on the number of seed points, not the size of the seismic volume. This means that there will be no computational cost in areas without faults, and most parts of a 3D seismic volume are nonfaulting areas. Using an four-core computer, our parallel implementation of the method can process more than 1000 seeds in 1 s to compute the corresponding voting surfaces and a voting score map. For the 3D example in Figure 11, we automatically pick 37,900 seed points and we compute the voting score map in Figure 11f by using only 29 s, compared with 774 s to compute the fault likelihood image (Figure 13a).

More 3D examples

Figure 14a shows another 3D seismic image, and the fault likelihood image in Figure 14b provides a pretty good detection of the fault discontinuities from the seismic image. However, some noisy features are also highlighted in this fault likelihood image and the thick red areas blur the fault detection. From the fault likelihood image, most of the large fault surfaces, as shown in Figure 14c, are successfully extracted by linking nearby fault samples. However, many noisy fault surfaces, especially those small ones, are also generated and small holes are apparent on some of the extracted large fault surfaces.

Figure 15a shows another fault attribute image (1-planarity), in which the fault features are much more noisy and discontinuous than the fault likelihood image (Figure 15a). Directly extracting fault surfaces from such a fault attribute image would be highly difficult. Therefore, we use this attribute image as an input for the optimal surface-voting method and compute an output voting score map, which is a much better fault attribute image as shown in Figure 15b. The fault features in this voting score map are much cleaner, sharper, and more continuous than those in the input attribute image (Figure 15a) and the fault likelihood image (Figure 14b). Again, we did not apply any thinning processing to the voting score map, but the fault features in this map are much thinner than those in the other two fault

attribute images and they provide sharper detection of the faults. From the voting score map, the automatically extracted fault surfaces (Figure 15c) are also cleaner and more complete than those in Figure 14c. The fault surfaces in Figures 14c and 15c are colored by fault strikes. In computing the voting score map in Figure 15b, we automatically picked 161,838 seed points. The whole processing of picking optimal surfaces for all the seed points and computing a voting score map took only 141.08 s by using an four-core computer. With the same computer, calculating the fault likelihood image (Figure 14b) took approximately 2500 s.

The 3D example shown in Figure 16 is much more complicated than the previous one because the seismic reflections and faults are not clearly imaged and a lot of noise are apparent in the seismic image (Figure 16a). We tried to compute a fault likelihood image for this 3D example, but it failed to correctly detect the faults as in the 2D example in Figure 10c. However, by using our optimal surface-voting method, we are still able to compute a pretty good fault detection in the thinned voting score map shown in Figure 16b. The detected faults are clean and continuous, and most of them correctly follow true faults apparent in the 3D seismic image. Figure 16c shows the fault surfaces (colored by fault strikes) that are automatically extracted from the voting score map (Figure 16b). These fault surfaces are continuous and complete, and no holes are observed.

CONCLUSION

We have discussed an efficient slope-constrained dynamic programming algorithm for picking globally optimal paths (2D) and surfaces (3D). By solving for a global maximization with low slope constraints, this algorithm is robust to pick a smooth and stable fault path or surface from an attribute image with even highly noisy and discontinuous fault features. With the following two assumptions, this algorithm can be directly applied to robustly pick single fault curves or surfaces from a fault attribute image. The first assumption is that the fault should be approximately linear or planar within an area where the fault is picked. This assumption is true for most cases, but it may fail for some special faults such as conical faults (Hale and Groshong, 2014) with the full range of fault strike angles. The second assumption is that the fault should pass through the whole area where the global maximization is defined for the optimal path or surface picking. This assumption is difficult to satisfy because we often do not know the spatial range of a fault before we pick it.

However, these two assumptions can be well-satisfied by choosing only local windows for picking short fault segments or small fault patches because a fault should be locally linear or planar and should have some spatial extensions. Therefore, we use the slope-constrained dynamic programming algorithm to pick local paths or surfaces from a fault attribute image within local windows and then further use these picked paths or surfaces as voters to vote for a new fault attribute image in which the full faults can be continuously tracked. In the method of optimal path or surface voting, we first automatically pick sparse seed points from an input fault attribute image. Then, we use these seeds as control points to pick local paths or surfaces that pass through the points and follow globally maximum attribute values within local windows. We smooth the attribute values along the voter (a path or surface) to suppress noisy features and fill in the gaps among discontinuous fault features. We further define each picked path (two dimensions) or surface (three dimensions 3D) as a voter and define voting scores on the voter by using the smoothed attribute values, so that the voter passing through higher attribute values will be assigned with higher voting scores. We collect the voting scores of all the voters to compute a final voting score map and normalize the map to obtain a new fault attribute image with clean, sharp, and continuous fault features. We also average the orientations of the overlapping voters to compute maps of fault strikes and dips. In the new fault attribute image, we finally track the continuous fault features along estimated fault strikes and dips to automatically extract full fault curves or surfaces.

The optimal path or surface-voting method is highly efficient because the optimal path or surface-picking algorithm is efficient and the total computational cost depends on the number of seed points, which are sparsely picked at fault positions. Our parallel implementation of the method can process more than 1000 seed points in 1 s by using an four-core computer.

REFERENCES

Al-Dossary, S., and K. J. Marfurt, 2006, 3D volumetric multispectral estimates of reflector curvature and rotation: Geophysics, **71**, no. 5, P41–P51, doi: 10.1190/1.2242449.

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

Bakker, P., 2002, Image structure analysis for seismic interpretation: Ph.D. thesis, Delft University of Technology.

Barnes, A. E., 2006, A filter to improve seismic discontinuity data for fault interpretation: Geophysics, **71**, no. 3, P1–P4, doi: 10.1190/1.2195988.

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

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein, 2001, Introduction to algorithms, 2nd ed.: MIT Press.

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

Di, H., and D. Gao, 2016, Efficient volumetric extraction of most positive/negative curvature and flexure for fracture characterization from 3D seismic data: Geophysical Prospecting, **64**, 1454–1468, doi: 10.1111/gpr.2016.64.

Dorn, G., B. Kadlec, and P. Murtha, 2012, Imaging faults in 3D seismic volumes: 82nd Annual International Meeting, SEG, Expanded Abstracts, doi: 10.1190/segam2012-1538.1.

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

Gersztenkorn, A., and K. J. 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.

Hale, D., 2009, Structure-oriented smoothing and semblance: CWP Report 635.

Hale, D., 2013a, Dynamic warping of seismic images: Geophysics, **78**, no. 2, S105–S115, doi: 10.1190/geo2012-0327.1.

Hale, D., 2013b, Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images: Geophysics, **78**, no. 2, O33–O43, doi: 10.1190/geo2012-0331.1.

Hale, D., and R. H. Groshong, 2014, Conical faults apparent in a 3D seismic image: Interpretation, **2**, no. 1, T1–T11, doi: 10.1190/INT-2013-0121.1.

Li, F., and W. Lu, 2014, Coherence attribute at different spectral scales: Interpretation, **2**, no. 1, SA99–SA106, doi: 10.1190/INT-2013-0089.1.

Luo, S., and D. Hale, 2013, Unfaulting and unfolding 3D seismic images: Geophysics, **78**, no. 4, O45–O56, doi: 10.1190/geo2012-0350.1.

Marfurt, K. J., 2006, Robust estimates of 3D reflector dip and azimuth: Geophysics, **71**, no. 4, P29–P40, doi: 10.1190/1.2213049.

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

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

Neff, D. B., J. R. Grismore, and W. A. 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. I., T. Randen, L. Sonneland, and Ø. Steen, 2002, Automatic fault extraction using artificial ants: 72nd Annual International Meeting, SEG, Expanded Abstracts, 512–515.

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

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

Roberts, A., 2001, Curvature attributes and their application to 3D interpreted horizons: First Break, **19**, 85–100, doi: 10.1046/j.0263-5046.2001.00142.x.

Tingdahl, K., and N. Hemstra, 2003, Estimating fault-attribute orientation with gradient analysis, principal component analysis and the localized Hough-transform: 73rd Annual International Meeting, SEG, Expanded Abstracts, 358–361.

Van Bemmel, P. P., and R. E. Pepper, 2000, Seismic signal processing method and apparatus for generating a cube of variance values: U.S. Patent 6,151,555.

Van Vliet, L. J., and P. W. Verbeek, 1995, Estimators for orientation and anisotropy in digitized images: Proceedings of the First Annual Conference of the Advanced School for Computing and Imaging, 442–450.

Weickert, J., 1997, A review of nonlinear diffusion filtering, *in* B. ter Haar Romeny, L. Florack, J. Koenderink, and M. Viergever, eds., Scale-space theory in computer vision: Springer, 1–28.

Wu, X., 2017, Directional structure-tensor based coherence to detect seismic faults and channels: Geophysics, **82**, no. 2, A13–A17, doi: 10.1190/geo2016-0473.1.

Wu, X., and D. Hale, 2016a, 3D seismic image processing for faults: Geophysics, **81**, no. 2, IM1–IM11, doi: 10.1190/geo2015-0380.1.

Wu, X., and D. Hale, 2016b, Automatically interpreting all faults, unconformities, and horizons from 3D seismic images: Interpretation, **4**, no. 2, T227–T237, doi: 10.1190/INT-2015-0160.1.

Wu, X., and X. Janson, 2017, Directional structure tensors in estimating seismic structural and stratigraphic orientations: Geophysical Journal International, **210**, 534–548, doi: 10.1093/gji/ggx194.

Wu, X., S. Luo, and D. Hale, 2016, Moving faults while unfaulting 3D seismic images: Geophysics, **81**, no. 2, IM25–IM33, doi: 10.1190/geo2015-0381.1.

Wu, X., and Z. Zhu, 2017, Methods to enhance seismic faults and construct fault surfaces: Computers & Geosciences, **107**, 37–48, doi: 10.1016/j.cageo.2017.06.015.