Semblance, coherence, and other discontinuity attributes

Joe Kington[1]

Discontinuity calculations, also called coherence or semblance, are some of the most commonly used seismic attributes. They measure the amount of similarity between adjacent seismic traces. However, there are several types of discontinuity estimates and a plethora of names for similar attributes. As a result, it can be difficult to understand the differences among them. Although the discontinuity algorithm that any particular software package uses might be unknown, most are based on one of a few published methods. Understanding published discontinuity attributes gives insight into the trade-offs among different proprietary implementations. Therefore, in this tutorial and the accompanying Jupyter notebook, we will explore simplified Python implementations of a few widely used discontinuity algorithms.

Introduction

Discontinuity attributes are a variety of poststack calculations that measure the degree of local similarity or dissimilarity between seismic traces. Discontinuities in a seismic volume, such as faults or mass-transport deposits, are apparent as regions of low similarity. Therefore, discontinuity attributes most often are applied to highlight faults in a seismic volume but are also useful in detecting subtle stratigraphic features in map view. Semblance-based coherence (Marfurt et al., 1998), eigenstructure-based coherence (Gersztenkorn and Marfurt, 1999), and other discontinuity attributes are among the most widely applied seismic attributes today.

The terminology used for discontinuity attributes can be confusing. Terms such as *coherence*, *semblance*, *similarity*, *discontinuity*, and many others are used to refer to both the general class of attribute and to specific algorithms. For these reasons, in this tutorial, we will use the term *discontinuity* to refer to the general class of attribute. Specific algorithms always will include a reference in addition to the name the original author used to avoid confusion.

In practice, most discontinuity attributes are calculated and applied in proprietary seismic-interpretation packages. The exact algorithm used by a particular package is in some cases considered a trade secret or might be undocumented. For that reason, we will not refer directly to or compare commercially available implementations in this tutorial. However, most proprietary discontinuity attributes are based on or similar to published algorithms. Therefore, understanding widely cited published algorithms for discontinuity helps us to understand the trade-offs among different commercial implementations.

We will focus on demonstrating a few of the most cited discontinuity algorithms in the literature with short Python code snippets. The data we will work with are a small subset of the Penobscot 3D seismic data set from offshore Nova Scotia, Canada, owned by the Nova Scotia Department of Energy and distributed by dGB Earth Sciences under a CC-BY-SA license (Nova Scotia Department of Energy, 1992) (Figure 1a). These examples eschew practical considerations (e.g., memory usage and speed) and focus on demonstrating the algorithm using high-level concepts. The accompanying Jupyter notebook expands on these examples and gives runnable versions of these short snippets.

Early algorithms — Crosscorrelation

The earliest discontinuity algorithm was developed by Bahorich and Farmer (1995) and used the maximum crosscorrelation value of three traces. Each trace is correlated with a "moving-window" subset of two neighboring traces (Figures 1b and 2b). A complete implementation is given in the accompanying notebook, but we will skip it here for brevity. Bahorich and Farmer (1995) coin the term *coherence* for the attribute, based on its conceptual similarity to prestack methods for estimating stacking velocities (e.g., Taner and Koehler, 1969). Although this exact approach is computationally expensive and is not used widely today, it provides a starting point to understand later algorithms.

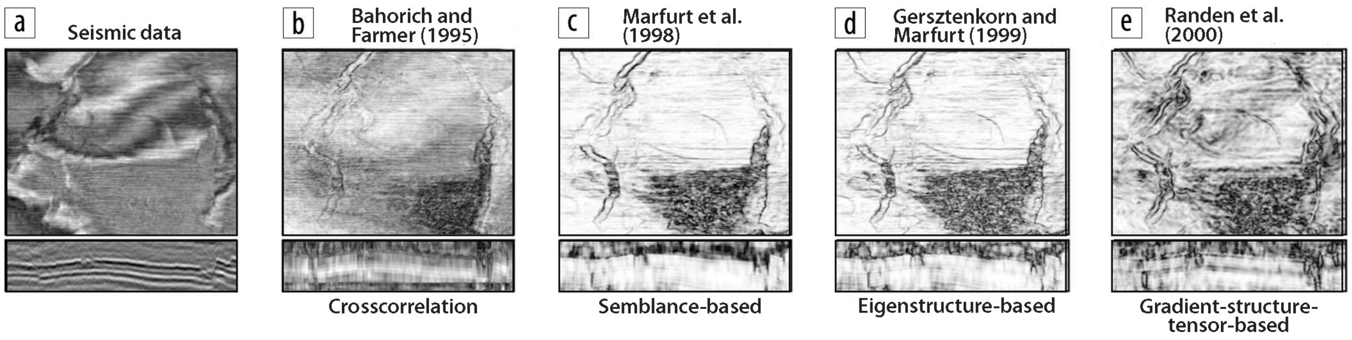

Figure 1. Comparison of the discontinuity algorithms discussed using the Penobscot 3D seismic data set (Nova Scotia Department of Energy, 1992).

[1]Chevron.

http://dx.doi.org/10.1190/tle34121510.1.

Generalization to an arbitrary number of traces

The approach of Bahorich and Farmer (1995) was successful, but it is sensitive to noise because only three traces are used. Marfurt et al. (1998) generalize Bahorich and Farmer's crosscorrelation approach to an arbitrary number of input traces, referred to by Marfurt et al. (1998) as *semblance-based coherence* (Figure 1c). As an example:

```
def moving_window(data, func, window):
   wrapped = lambda x: func(x.reshape(window))
   return scipy.ndimage.generic_filter(data,
                                       wrapped,
                                       window)


def marfurt_semblance(region):
   # Stack traces in 3D region into 2D array
   region = region.reshape(-1, region.shape[-1])
   ntraces, nsamples = region.shape
   square_sums = np.sum(region, axis=0)**2
   sum_squares = np.sum(region**2, axis=0)
   c = square_sums.sum() / sum_squares.sum()
   return c / ntraces


result = moving_window(seismic_data,
                       marfurt_semblance,
                       (3, 3, 9))
```

Conceptually, the algorithm of Marfurt et al. (1998) treats each seismic trace within the moving window as a separate dimension and measures how close the resulting point cloud is to a hyperplane with a slope of 1 in all directions and an intercept of 0. It is easiest to visualize for the case of two traces, as shown in Figure 2c. In that case, semblance-based coherence computes how close the points fall to a line with a slope of 1 and an intercept of 0.

Removing amplitude sensitivity

One caveat to the methods of Bahorich and Farmer (1995) and Marfurt et al. (1998) is that they are sensitive to lateral differences in amplitude and in phase. Although this is desirable for detecting stratigraphic features, differences caused by amplitude can obscure subtle structural features. Gersztenkorn and Marfurt (1999) propose an implementation that is sensitive only to lateral changes in phase of the input waveforms — *eigenstructure-based coherence* (Figures 1d and 2d).

Eigenstructure-based coherence (Gersztenkorn and Marfurt, 1999) computes the covariance matrix of the input region, similar to the previous example. However, it uses the ratio of the largest eigenvalue of the covariance matrix to the sum of the eigenvalues:

```
def gersztenkorn(region):
   region = region.reshape(-1, region.shape[-1])

   # Calculate eigenvalues of covariance matrix
   cov = region.dot(region.T)
   vals = np.linalg.eigvalsh(cov)
   return vals.max() / vals.sum()

result = moving_window(seismic_data,
                       gersztenkorn,
                       (3, 3, 9))
```

Conceptually, this is similar to treating each seismic trace within the moving window as a separate dimension and calculating how well the resulting point cloud is fit by a plane. To contrast with the method of Marfurt et al. (1998), for the case of two traces, this measures the scatter about the best-fit line instead of a line with a slope of 1, as shown in Figure 2d.

Gradient changes instead of trace similarity

An entirely different approach than any we have discussed so far is to apply a structure tensor to seismic attributes. The structure tensor measures how the gradient of each dimension covaries locally. Randen et al. (2000) were the first to propose this approach in their gradient-structure-tensor (GST) attributes. Several useful attributes can be computed from the ratios of the eigenvalues of the structure tensor, one of which is a measure of how planar the data are locally, which we will refer to as GST coherence (Randen et al., 2000) (Figure 1e).

```
from scipy.ndimage import gaussian_filter1d

def gradients(seismic, sigma):
   grads = []
   for axis in range(3):
     grad = gaussian_filter1d(seismic, sigma,
                              axis=axis, order=1)
```

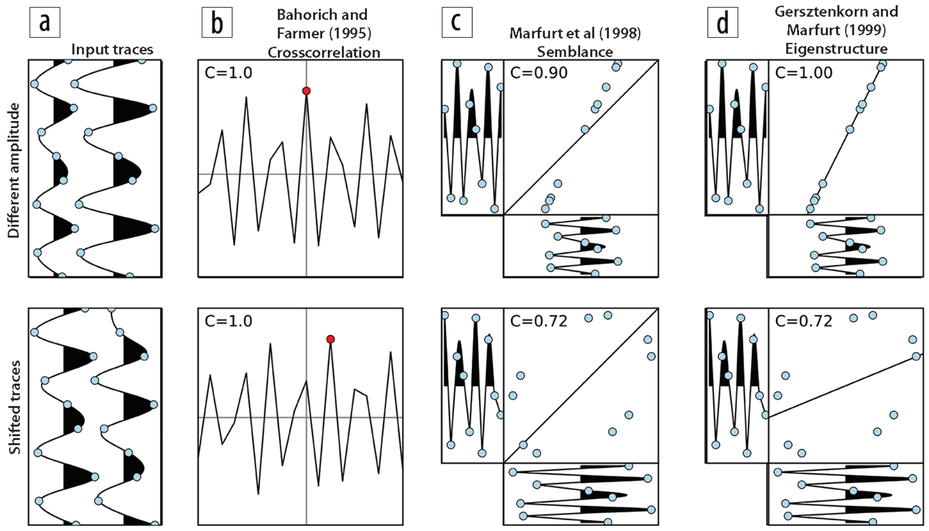
a
Input traces
b
Bahorich and Farmer (1995) Crosscorrelation
c
Marfurt et al (1998) Semblance
d
Gersztenkorn and Marfurt (1999) Eigenstructure
Different amplitude
Shifted traces
C=1.0
C=0.90
C=1.00
C=1.0
C=0.72
C=0.72

Figure 2. Visual explanation of the differences among the discontinuity algorithms discussed using only two traces. Letter C indicates the calculated coherence value.

```
        grads.append(grad[..., np.newaxis])
    return np.concatenate(grads, axis=3)

def gst_coherence_calc(region):
    region = region.reshape(-1, 3)
    gst = region.T.dot(region)
    eigs = np.sort(np.linalg.eigvalsh(gst))[::-1]
    return (eigs[0]-eigs[1]) / (eigs[0]+eigs[1])

def gst_coherence(seismic, window, sigma=1):
    grad = gradients(seismic, sigma)
    return moving_window4d(grad_array,
                           window,
                           gst_coherence_calc)
```

Because GST coherence measures change in the local slope of the data, it is related more closely to curvature attributes than to the other discontinuity attributes we have discussed (Randen et al., 2000; Chopra and Marfurt, 2007). Therefore, GST coherence can reveal different features than discontinuity attributes that compare waveform similarity, but it tends to show thicker regions of discontinuity around faults (Chopra and Marfurt, 2007) (Figure 1e).

Final thoughts

Discontinuity attributes are a fundamental part of an interpreter's tool kit when working with 3D data. Although most interpreters are familiar with using discontinuity attributes, few are familiar with how they are implemented or the differences among similar attributes. It is hoped that the accompanying Jupyter notebook and this brief tutorial will help to clear up some implementation details and will inspire you to attempt new variants on well-known discontinuity attributes. TLE

Acknowledgments

This article benefited greatly from reviews by Steve Purves and Matt Hall and suggestions from JoAnn Gage, Jonathan Fry, and Brendan Sullivan.

Corresponding author: joe.kington@chevron.com

References

Bahorich, M., and S. Farmer, 1995, 3-D seismic discontinuity for faults and stratigraphic features: The coherence cube: The Leading Edge, **14**, no. 10, 1053–1058, http://dx.doi.org/10.1190/1.1437077.
Chopra, S., and K. J. Marfurt, 2007, Seismic attributes for prospect identification and reservoir characterization: SEG Geophysical Development Series No. 11, http://dx.doi.org/10.1190/1.9781560801900.
Gersztenkorn, A., and K. J. Marfurt, 1999, Eigenstructure-based coherence computations as an aid to 3-D structural and stratigraphic mapping: Geophysics, **64**, no. 5, 1468–1479, http://dx.doi.org/10.1190/1.1444651.
Marfurt, K., R. L. Kirlin, S. L. Farmer, and M. S. Bahorich, 1998, 3-D seismic attributes using a semblance-based coherency algorithm: Geophysics, **63**, no. 4, 1150–1165, http://dx.doi.org/10.1190/1.1444415.
Nova Scotia Department of Energy, 1992, Penobscot 3D survey, data set, https://opendtect.org/osr/pmwiki.php/Main/PENOBSCOT3DSABLEISLAND, accessed 19 October 2015.
Randen, T., E. Monsen, C. Signer, A. Abrahamsen, J. O. Hansen, T. Sæter, and J. Schlaf, 2000, Three-dimensional texture attributes for seismic data analysis: 70th Annual International Meeting, SEG, Expanded Abstracts, 668–671, http://dx.doi.org/10.1190/1.1816155.
Taner, M. T., and F. Koehler, 1969, Velocity spectra — Digital computer derivation applications of velocity functions: Geophysics, **34**, no. 6, 859–881, http://dx.doi.org/10.1190/1.1440058.