# A method for estimating apparent displacement vectors from time-lapse seismic images

Dave Hale
*Center for Wave Phenomena, Colorado School of Mines, Golden CO 80401, USA*

**ABSTRACT**
Reliable estimates of vertical, inline and crossline components of apparent displacements in time-lapse seismic images are difficult to obtain for two reasons. First, features in 3-D seismic images tend to be locally planar, and components of displacement within the planes of such features are poorly resolved. Second, searching directly for peaks in 3-D cross-correlations is less robust, more complicated, and computationally more costly than searching for peaks of 1-D cross-correlations.
We estimate all three components of displacement with a process designed to mitigate these two problems. We address the first problem by computing for each image sample a local phase-correlation instead of a local cross-correlation. We address the second problem with a cyclic sequence of searches for peaks of correlations computed for lags constrained to one of the three axes of our images.

**Key words:** time-lapse seismic image processing

## 1 INTRODUCTION

Tiny displacements we observe in 3-D time-lapse seismic images are vectors, with three - vertical, inline, and crossline - components. These *apparent displacements* can be caused by reservoir compaction and are especially sensitive to related changes in strains and seismic wave velocities above reservoirs.

By "tiny", we mean displacements that may be only a fraction of a sampling interval. Figures 1–4 show an example from time-lapse seismic imaging of a high-pressure high-temperature reservoir in the North Sea. Here we estimated vertical apparent displacements roughly equal to the time sampling interval of 4 ms. In the inline and crossline directions, we estimated horizontal apparent displacements of approximately 5 m, which is much less than the 25 m inline and crossline sampling intervals.

Though small, the most significant inline and crossline displacements appear to be correlated with the geometry of the target reservoir. The point of intersection of the three orthogonal slices in each of Figures 1–4 lies just beneath that reservoir.

Apparent vertical (time) displacements like those shown in Figure 2 tend to be downward (positive), even when physical reservoir boundaries are displaced upwards. This difference between physical and apparent vertical displacements has been observed and explained by Hatchell and Bourne (2005).

Apparent horizontal displacements are less well understood, although these too have been measured here and by others (e.g., Hall, 2006). Figures 3 and 4 show apparent displacements that are generally smaller in the inline direction than in the crossline direction.

Figure 4 implies that, near the reservoir, 3-D seismic images are pulling apart in the crossline direction as fluids are extracted. This apparent horizontal stretching is the opposite of the compaction that we might expect if we interpreted such displacements as physical movements of reservoir rocks. However, the apparent stretching we observe here is reasonable if we consider the effect of a mild low-velocity lens above the reservoir induced by compaction. If not accounted for in seismic migration (as it was not here), such a change in seismic velocity could explain these apparent crossline displacements.

Such speculation notwithstanding, our understanding of apparent vector displacements today remains incomplete and beyond the scope of this paper. Our goal

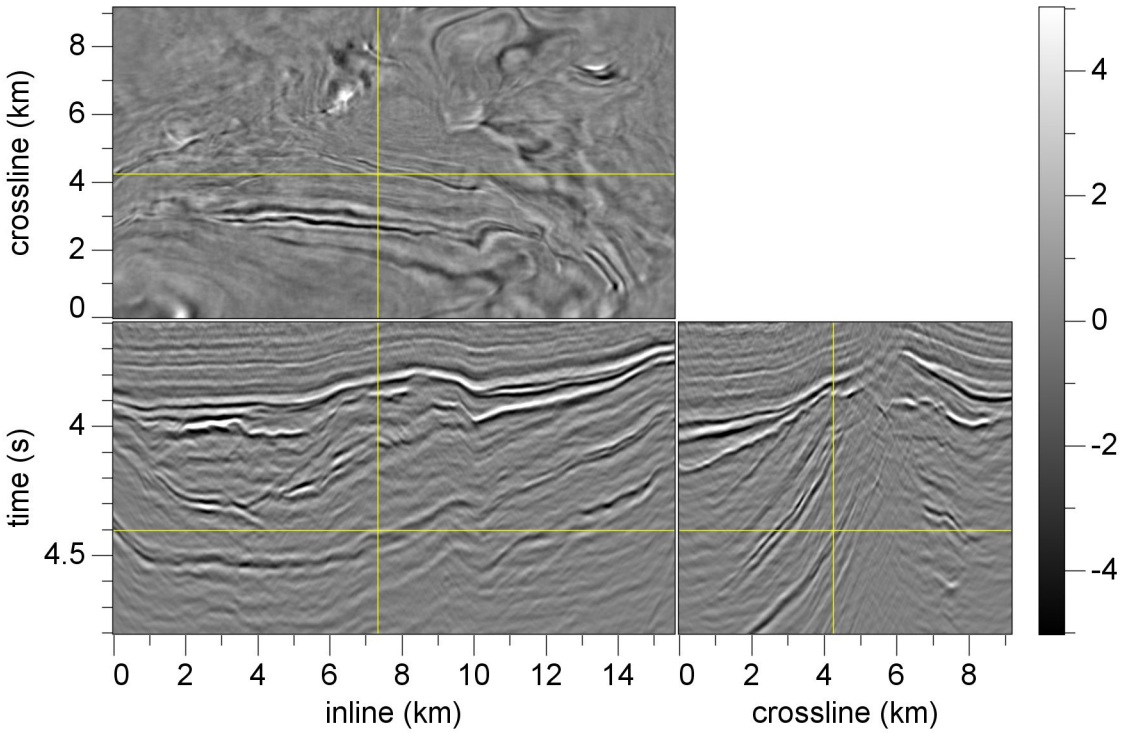



**Figure 1.** Three orthogonal slices of a 3-D seismic image recorded in 2002. A second image (not shown) was recorded in 2004. Crosshairs in each slice show the locations of the other two slices.

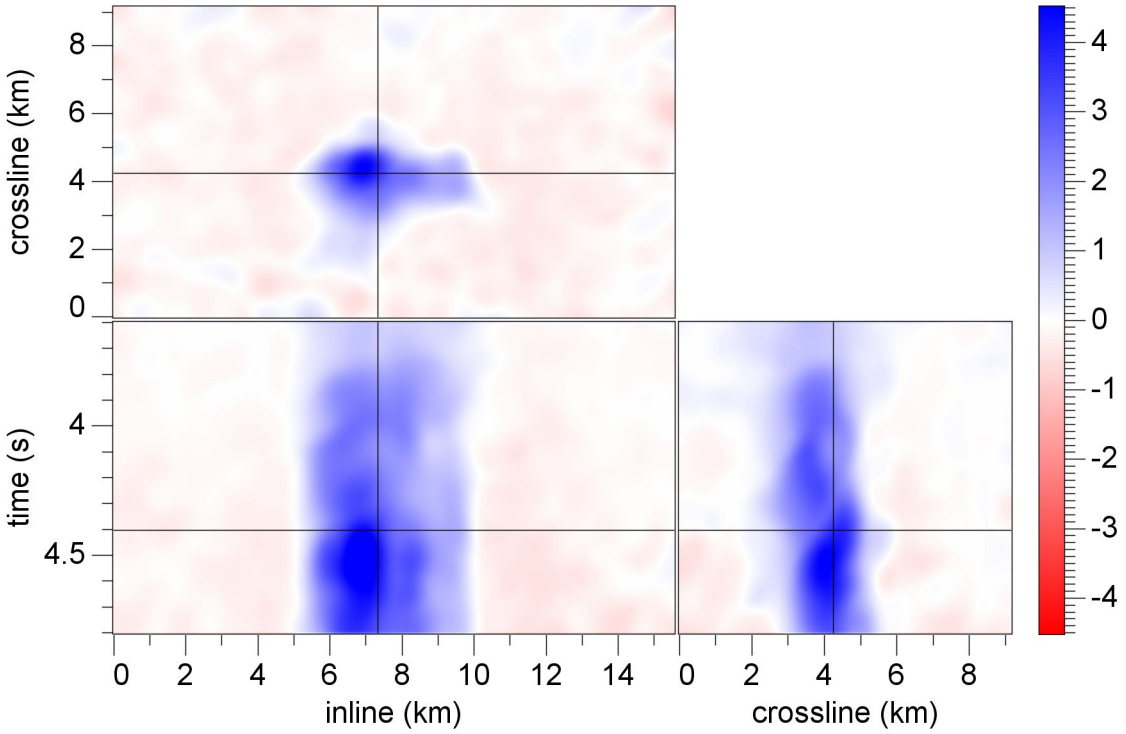



**Figure 2.** Vertical components of apparent displacement measured in ms.

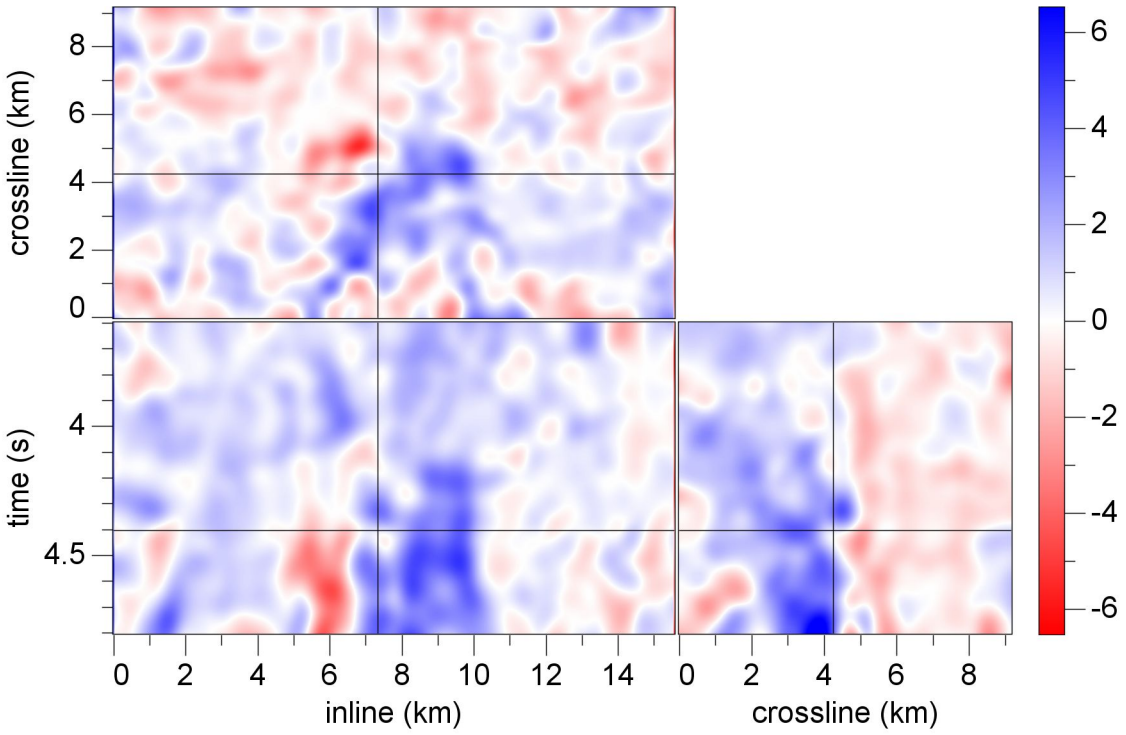



**Figure 3.** Inline components of apparent displacement measured in m.

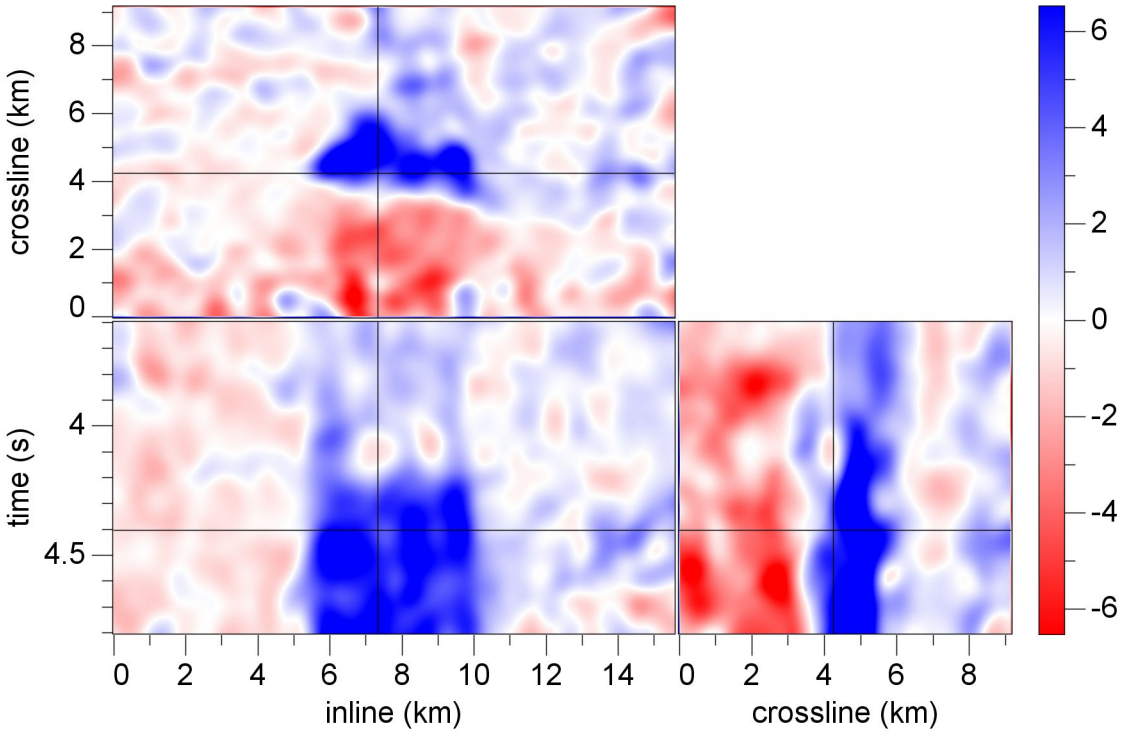
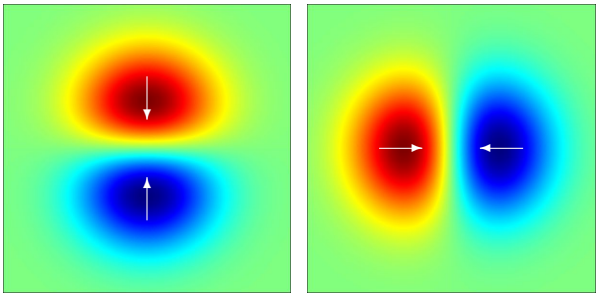



**Figure 4.** Crossline components of apparent displacement measured in m.



(a) (b)

**Figure 5.** Vertical (a) and horizontal (b) components of a synthetic vector displacement field representing compaction of an image towards its center. Dark red denotes three samples of vertical displacement (a) downward or (b) toward the right. Dark blue denotes three samples of vertical displacement (a) upward or (b) toward the left.

here is to describe the process by which we obtained these estimates of apparent vector displacements from time-lapse seismic images.

Estimation of all three components of displacements is difficult. One difficulty is that displacements of image features are poorly resolved in directions parallel to those features. Therefore, in seismic images where features are often more or less horizontal, we tend to estimate only the vertical component of displacement, because only that component is well resolved.

A second difficulty is that some processing techniques used to estimate only a single vertical component of displacement do not extend easily to estimation of all three components. For example, estimating the locations of peaks of cross-correlations of images is straightforward when those correlations are functions of only vertical lag. A simple quadratic interpolation of correlation values near a peak may suffice. An extension of this processing to finding peaks in correlations that are a function of two or three components of lag is more complicated and less robust, partly because of the resolution problem described above.

Finally, estimation of three components of displacement requires more computation, and the increase in cost can be significant when estimating a complete field of displacement vectors for every sample in 3-D images.

In this paper we illustrate these difficulties and describe a process that addresses them.

## 2 LOCAL CROSS-CORRELATIONS

Consider first only the two components of displacement shown in Figure 5. The displacement vectors in this example correspond to compaction or squeezing of an image towards its center.

### 2.1 Displacements between images

Using the synthetic displacement vector field shown in Figure 5, we can warp one seismic image to obtain another. Specifically, let sampled functions $f[j_1,j_2]$ and $g[j_1,j_2]$ denote two images related by

$$f[j_1,j_2] = g\big(j_1+u_1[j_1,j_2],\ j_2+u_2[j_1,j_2]\big), \tag{1}$$

where $u_1[j_1,j_2]$ and $u_2[j_1,j_2]$ represent the vertical and horizontal components of the vector displacement field $\mathbf{u}[j_1,j_2]$.

Throughout this paper we adopt the convention that $f[j_1,j_2]$ (with square brackets) is an image obtained by uniformly sampling a continuous function $f(x_1,x_2)$ (with parentheses) for integer pixel indices $j_1$ and $j_2$. We also assume that $f(x_1,x_2)$ is bandlimited and that $f[j_1,j_2]$ is not aliased, so that sinc interpolation can reconstruct the continuous function $f(x_1,x_2)$ with any required precision.

Because components of displacement $u_1$ and $u_2$ need not be integer values, the warping operation described by equation 1 implies interpolation of the sampled image $g[j_1,j_2]$ to compute $f[j_1,j_2]$.

Assume that we have two images $f$ and $g$ related by the synthetic displacement vector field of Figure 5. Can we recover the known displacement vectors $\mathbf{u}$ from the images?

### 2.2 Local cross-correlations

Figure 6 illustrates an attempt to estimate the displacement vector field $\mathbf{u}$ displayed in Figure 5 from two images $f$ and $g$. The images are displayed in Figures 6a and 6b and at this scale appear to be identical, because maximum displacements are less than three samples in both vertical and horizontal directions.

To estimate displacement vectors $\mathbf{u}$, we search for locations of peaks of local cross-correlations. We define local cross-correlation of two images $f$ and $g$ by

$$c_{fg}[k_1,k_2;l_1,l_2] \equiv \sum_{j_1,j_2} f[j_1,j_2]\ g[j_1+l_1,j_2+l_2] \times w[k_1-j_1,k_2-j_2], \tag{2}$$

where $w[k_1,k_2]$ is a 2-D Gaussian window defined by

$$w[k_1,k_2] \equiv \mathrm{e}^{-(k_1^2+k_2^2)/2\sigma^2} \tag{3}$$

for a specified radius $\sigma$. (In the example of Figure 6, we used $\sigma=12$ samples.) Summation indices $j_1$ and $j_2$ are limited by finite image bounds.

Equation 2 implies that for each pair of lag indices $[l_1,l_2]$ we compute a local cross-correlation value for every image sample indexed by $[k_1,k_2]$. In other words, we compute a correlation image $c_{fg}$ as large as $f$ and $g$ for each $[l_1,l_2]$. When we do this for many lags, the resulting $c_{fg}[k_1,k_2;l_1,l_2]$ could consume large amounts of storage. However, when estimating displacements, we

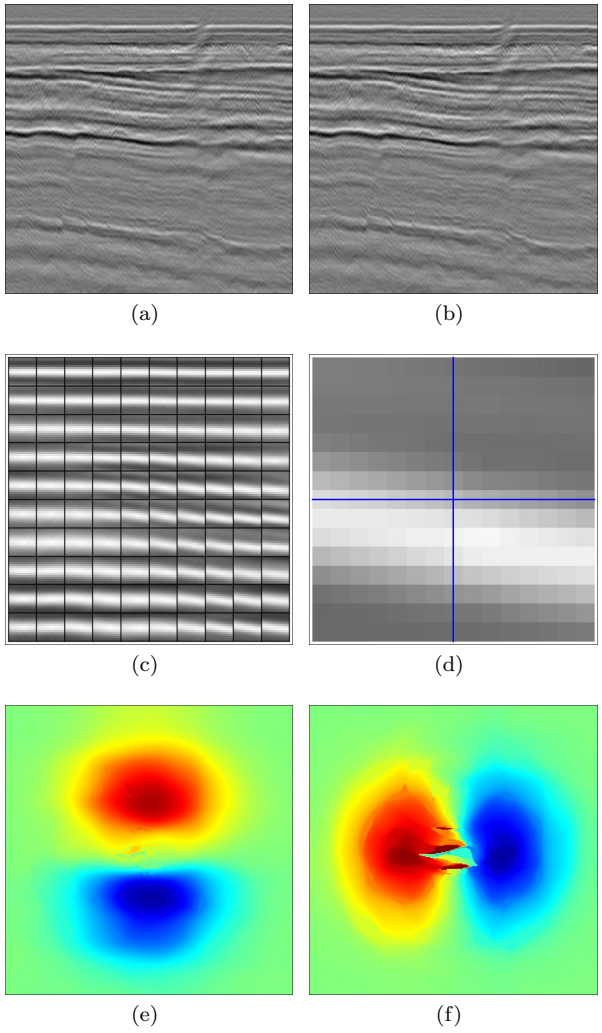


**Figure 6.** Estimates of displacement vectors from two 315 × 315-pixel images. The two images $f$ (a) and $g$ (b) are related by equation 1 for the displacements shown in Figure 5. The subset (c) of normalized local 2-D cross-correlations corresponds to the upper-left quadrant of these images, where displacements are downward and rightward. The downward shift is especially visible in one of these cross-correlations (d) shown in detail. A straightforward search for peaks of such cross-correlations yields estimates of both vertical (e) and horizontal (f) components of displacements.

need to store for each image sample only those correlation values required to locate correlation peaks.

The Gaussian window $w$ makes the cross-correlations *local*. For any lag $[l_1, l_2]$, equation 2 represents convolution of this Gaussian window with a lagged image product $f[j_1, j_2]\ g[j_1 + l_1, j_2 + l_2]$. We perform this Gaussian filtering efficiently using recursive implementations (Deriche, 1992; van Vliet et al., 1998; Hale, 2006) with computational cost that is independent of the window radius $\sigma$.

Local windows can make cross-correlations sensitive to local amplitude variations. As we vary the lag $[l_1, l_2]$ in equation 2, high-amplitude events in $g$ may slide in and out of the local Gaussian window, creating spurious correlation peaks that are inconsistent with true displacements.

### 2.3 Normalized local cross-correlations

To avoid this problem, the cross-correlation values displayed in Figures 6c and 6d have been normalized. Shown here are values of

$$\begin{aligned} c[k_1, k_2; l_1, l_2] \quad \equiv \quad & c_{fg}[k_1, k_2; l_1, l_2] \\ & \times \frac{1}{\sqrt{c_{ff}[k_1, k_2; 0, 0]}} \\ & \times \frac{1}{\sqrt{c_{gg}[k_1 + l_1, k_2 + l_2; 0, 0]}}. \end{aligned} \tag{4}$$

We compute the normalization factors $1/\sqrt{c_{ff}}$ and $1/\sqrt{c_{gg}}$ using special cases of equation 2:

$$c_{ff}[k_1, k_2; 0, 0] \equiv \sum_{j_1, j_2} f^2[j_1, j_2]\ w[k_1 - j_1, k_2 - j_2]$$

and

$$c_{gg}[k_1, k_2; 0, 0] \equiv \sum_{j_1, j_2} g^2[j_1, j_2]\ w[k_1 - j_1, k_2 - j_2].$$

These scale factors can be computed once and reused for all lags $[l_1, l_2]$. With these definitions (and by the Cauchy-Schwarz inequality), *normalized* local cross-correlations have the property $|c| \leq 1$.

Figure 6c shows a small subset of the 2-D normalized local cross-correlations computed for the upper-left quadrant of the two images in 6a and 6b; Figure 6d shows just one of these cross-correlations. We compute cross-correlations like these for every image sample. Each cross-correlation is local in the sense that it depends on samples within a 2-D Gaussian window of radius $\sigma = 12$ samples. As the window slides across the images, the local cross-correlations vary as in Figure 6c.

Figures 6c and 6d indicate a displacement of cross-correlation peaks vertically downward, which is consistent with the upper-left quadrant of Figure 5a. Horizontal (rightward) displacements of those peaks are more difficult to see. Displacements perpendicular to image features are more well resolved than those parallel to those features.

### 2.4 Quadratic interpolation

By simply searching over all sampled lags in Figure 6d, we can easily find the integer indices $[l_1, l_2]$ of the lag with highest correlation value. We might then fit some function to that value and others nearby to resolve the correlation peak location with sub-pixel precision.

A common choice for the fitting function is a

quadratic polynomial. For example, in one dimension, this quadratic function has the form

$$c(u) = a_0 + a_1 u + a_2 u^2.$$

We can choose the three coefficients $a_0$, $a_1$ and $a_2$ so that this polynomial interpolates exactly three sampled correlation values $c[l-1]$, $c[l]$ and $c[l+1]$. If a lag $l$ is found such that $c[l]$ is not less than the other two correlation values and is greater than at least one of them, then the quadratic polynomial has a peak at

$$u = l + \frac{c[l-1] - c[l+1]}{2c[l-1] + 2c[l+1] - 4c[l]}. \quad (5)$$

It is both easy to prove and sensible that $|u-l| \leq \frac{1}{2}$. The correlation peak found by quadratic fit lies within half a sample of the largest correlation value found by scanning integer lags $l$. A scan of sampled correlation values provides a rough integer estimate of displacement; quadratic interpolation simply refines that estimate.

Other fitting functions are possible. In particular, a sinc function is most accurate for interpolating bandlimited sampled signals. But sinc interpolation does not provide a simple closed-form expression like equation 5 for the peak location. Therefore, quadratic interpolation is often used to find peaks.

Though small for bandlimited signals, the error in quadratic interpolation is biased in both peak amplitude and location. In addition to this error, two more problems arise with quadratic interpolation in higher dimensions.

### 2.5 Quadratic 2-D and 3-D interpolation

One problem is that the number of coefficients in a quadratic polynomial in higher dimensions does not equal the number of samples in any symmetric neighborhood nearest a sampled maximum correlation value. For example, in two dimensions the bi-quadratic polynomial has six coefficients:

$$c(u_1, u_2) = a_0 + a_1 u_1 + a_2 u_2 + a_3 u_1^2 + a_4 u_1 u_2 + a_5 u_2^2.$$

This number exceeds the number of correlation values in a five-sample neighborhood consisting of the values $c[l_1, l_2]$, $c[l_1 \pm 1, l_2]$, and $c[l_1, l_2 \pm 1]$; it is less than the number in a nine-sample neighborhood obtained by also including $c[l_1 \pm 1, l_2 \pm 1]$. To resolve this inconsistency, a bi-quadratic may be least-squares fit to the nine correlation values in the nine-sample neighborhood, but this fitted function does not generally interpolate any of the sampled correlation values within that neighborhood.

Likewise, in three dimensions a tri-quadratic polynomial has 10 coefficients, but symmetric neighborhoods of correlation values nearest a sampled maximum have either 7 (too few), 19 or 27 (too many) values. Quadratic polynomials in dimensions greater than one are either under- or over-constrained by cross-correlation values sampled at integer lags.

A second problem is that a peak location found by least-squares quadratic fit in two (or higher) dimensions need not lie within half a pixel of the integer lag indices $[l_1, l_2]$ corresponding to the maximum sampled correlation value. Indeed, for sampled correlations like those shown in Figures 6c and 6d, a least-squares-fit bi-quadratic may have a saddle point instead of a peak.

In other words, in two (or higher) dimensions, a peak may not exist for the least-squares quadratic fit to nine (or more) correlation values nearest to a sampled maximum value. And even when a quadratic peak does exist, it may be far away from the integer lag indices $[l_1, l_2]$ corresponding to the sampled maximum.

Such cases are pathological but not exceptional. We have observed them often while fitting bi-quadratic polynomials to the nine sampled values nearest the maximum value in correlations like those shown in Figures 6c and 6d.

These problems account for the most significant errors in the estimated components $u_1$ and $u_2$ of displacement vectors shown in Figures 6e and 6f. In this example errors are most significant for the horizontal component $u_2$, because the locations of correlation peaks in Figures 6c and 6d are least well resolved in that direction.

Discontinuities in apparent displacement vectors like those shown in Figure 6f are unreasonable, for they imply infinite apparent strain.

One way to eliminate or at least reduce such discontinuities is to seek displacements that maximize a weighted combination of image correlation and displacement smoothness. As discussed and implemented by Hall (2006), this approach implies a tradeoff. We must choose relative weights for correlation and smoothness. Rickett et al. (2006) discuss a similar correlation and smoothing tradeoff in estimating only vertical (time) shifts. These authors highlight the importance of not smoothing too much.

We describe a different approach below. We improve the accuracy of estimated displacements with a process that includes improved image processing and a more robust method for finding correlation peaks. With this process we obtain more accurate and thereby more continuous estimates of displacements without explicitly smoothing them.

## 3 LOCAL PHASE-CORRELATION

The first step in our process is to improve the spatial resolution of cross-correlations in directions parallel to features in seismic images. We do this by applying spatially-varying multi-dimensional prediction error filters to both images before cross-correlating them. The prediction error filters whiten the spectra of our images in all spatial dimensions.

Figures 7 illustrate the effect that this spectral whitening step has on our estimates of apparent displacements. After whitening, cross-correlation peaks are

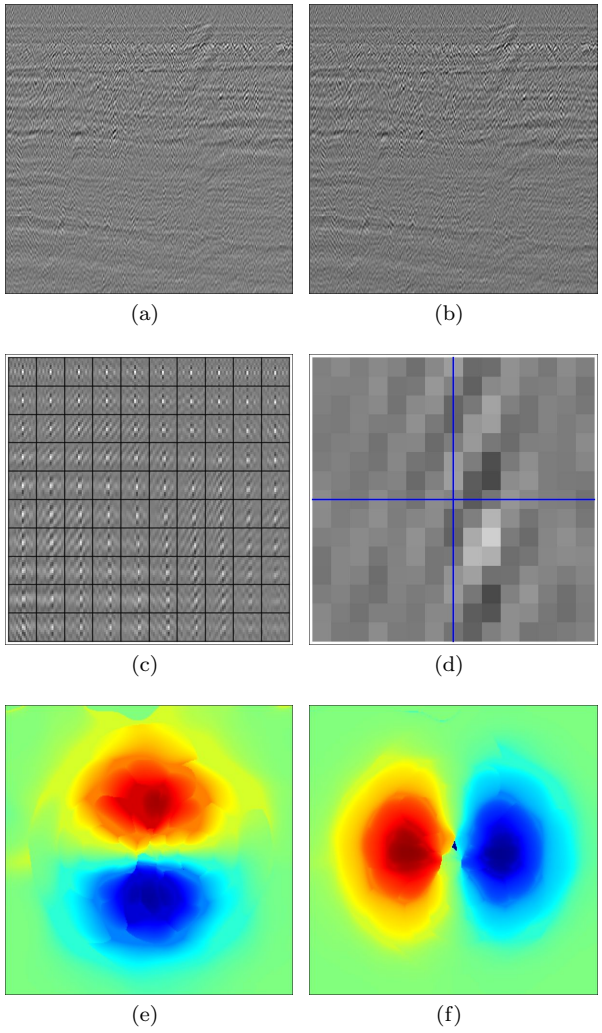


**Figure 7.** Images (a) and (b) after whitening with local prediction error filters. Peaks of local phase-correlations in (c) and (d) are well resolved in both vertical and horizontal directions. Estimates of vertical and horizontal components of displacement in (e) and (f) are more reliable than those without whitening. Visible patterns of errors and discontinuities in these estimates are caused by fitting 2-D quadratic functions to sampled correlation values nearest the peaks.

well resolved in both vertical and horizontal directions. In Figures 7c and 7d we see horizontal displacements of those peaks that are not apparent in Figures 6c and 6d.

### 3.1 Phase-correlation with Fourier transforms

Cross-correlation of whitened images is equivalent to phase-correlation, a process that is widely used in the context of image registration (Kuglin and Hines, 1975).

Phase-correlations are usually computed using Fourier transforms. Let $F$ and $G$ denote the Fourier transforms of images $f$ and $g$, respectively. Then the cross-correlation $c = f \star g$ has Fourier transform $C = F^*G$. Assume temporarily that $f$ and $g$ are related by a constant displacement vector shift $\mathbf{u}$ such that

$$f(x_1, x_2) = g(x_1 + u_1, x_2 + u_2).$$

Then

$$F(k_1, k_2) = G(k_1, k_2)e^{ik_1u_1 + ik_2u_2}$$

and

$$C(k_1, k_2) = |F(k_1, k_2)||G(k_1, k_2)|e^{-ik_1u_1 - ik_2u_2}.$$

In phase-correlation we divide by the amplitude factors to obtain

$$P(k_1, k_2) \equiv \frac{F^*(k_1, k_2)}{|F(k_1, k_2)|} \frac{G(k_1, k_2)}{|G(k_1, k_2)|} = e^{-ik_1u_1 - ik_2u_2}.$$

These divisions in the frequency domain whiten the amplitude spectra of our images $f$ and $g$. After this division the inverse-Fourier transform of $P(k_1, k_2)$ is a shifted delta function

$$p(x_1, x_2) = \delta(x_1 - u_1, x_2 - u_2).$$

Note that the peak at $(u_1, u_2)$ of the phase-correlation delta function $p$ is equally well-resolved in all directions.

In practice both $p$ and $P$ are sampled functions and we compute the latter with fast Fourier transforms. One method for then estimating the components of constant displacement $u_1$ and $u_2$ is to fit by least-squares a plane to the sampled phase of $P(k_1, k_2)$, perhaps restricting the fit to those frequencies $(k_1, k_2)$ for which signal-to-noise ratios are high. The fitting parameters are the unknown components $u_1$ and $u_2$. More sophisticated frequency-domain methods are described by Hoge (2003) and by Balsi and Foroosh (2006).

In our discussion of phase-correlations above we temporarily assumed that the components of displacement $u_1$ and $u_2$ are constants. When displacements vary spatially, Fourier-transform methods applied to local windows are costly, especially when estimating apparent displacement vectors for every image sample. To estimate a dense spatially varying vector field of apparent displacements, we need an alternative space-domain method.

### 3.2 Local prediction error filtering

Our alternative is to apply local prediction error filters to the images $f$ and $g$ before computing local cross-correlations. These filters approximately whiten the amplitude spectra of the images, much like frequency-domain division by $|F|$ and $|G|$ in phase-correlation. The difference is that the prediction error filters are local; we compute and apply a different filter for every image sample.

For a sampled image $f$, the simplest 2-D prediction error filter that could possibly work computes prediction

errors

$$e[j_1,j_2;k_1,k_2] \equiv f[j_1,j_2] \\ -a_1[k_1,k_2]f[j_1-1,j_2] \\ -a_2[k_1,k_2]f[j_1,j_2-1],$$

where the coefficients $a_1[k_1,k_2]$ and $a_2[k_1,k_2]$ are found by minimizing a sum of squared prediction errors:

$$E[k_1,k_2] \equiv \sum_{j_1,j_2} e^2[j_1,j_2;k_1,k_2]\ w[k_1-j_1,k_2-j_2].$$

Again, the Gaussian window $w$ localizes our computation of the prediction coefficents $a_1$ and $a_2$. For each image sample indexed by $[k_1,k_2]$, we minimize this sum by setting both $\partial E/\partial a_1$ and $\partial E/\partial a_2$ to zero, which leads to the following system of equations:

$$\left[\begin{array}{cc} R_{11} & R_{12} \\ R_{12} & R_{22} \end{array}\right] \left[\begin{array}{c} a_1 \\ a_2 \end{array}\right] = \left[\begin{array}{c} r_1 \\ r_2 \end{array}\right], \qquad (6)$$

where

$$\begin{aligned} R_{11} &= \sum_{j_1,j_2} f^2[j_1-1,j_2]\ w[k_1-j_1,k_2-j_2] \\ &= \sum_{j_1,j_2} f^2[j_1,j_2]\ w[k_1-1-j_1,k_2-j_2] \\ &= C_{ff}(k_1-1,k_2;0,0), \end{aligned}$$

and likewise

$$R_{12} = C_{ff}(k_1-1,k_2;1,-1),$$

$$R_{22} = C_{ff}(k_1,k_2-1;0,0),$$

$$r_1 = C_{ff}(k_1,k_2;-1,0), \text{ and}$$

$$r_2 = C_{ff}(k_1,k_2;0,-1).$$

We use equation 2 to compute the auto-correlation values $C_{ff}(k_1,k_2;l_1,l_2)$. The $2 \times 2$ matrix in equations 6 is symmetric and positive-definite (for non-constant $f$), as it is a Gaussian-weighted sum of outer products:

$$\left[\begin{array}{c} f[j_1-1,j_2] \\ f[j_1,j_2-1] \end{array}\right] \left[\begin{array}{cc} f[j_1-1,j_2] & f[j_1,j_2-1] \end{array}\right].$$

Therefore, this matrix is never singular and a solution $[a_1\ a_2]$ always exists.

In the 3-D example of Figures 1–4, we computed three prediction coefficients $a_1$, $a_2$, and $a_3$ in a straightforward extension of equations 6. We again used a 3-D Gaussian window with radius $\sigma = 12$ samples.

It is important that we evaluate the auto-correlation values in equation 6 at the correct sample indices. For example, the value $R_{11} = C_{ff}[k_1-1,k_2;0,0]$ that we need to compute $a_1[k_1,k_2]$ and $a_2[k_1,k_2]$ typically does not equal the value $C_{ff}[k_1,k_2;0,0]$. If the latter value is used, then the system of equations 6 may not be positive-definite and solutions may not exist.

By computing and applying prediction error filters for every sample of the images shown in Figures 6a and 6b, we obtain the images shown in Figures 7a and 7b. Although our local prediction error filters are simple, with only two coefficients, the normalized local cross-correlations shown in Figures 7c and 7d have peaks that are more well-resolved than those in Figures 6c and 6d.

Unfortunately, resolution of the correlation peaks in Figures 7c and 7d is now too high. A quadratic function is inadequate for interpolation of broadband signals such as the sampled correlation function shown in Figure 7d. Errors in quadratic fitting are responsible for the discontinuities and patterns visible in the estimated components of displacement displayed in Figures 7e and 7f.

### 3.3 Bandlimited local phase-correlations

To reduce errors in quadratic fitting, we apply a 2-D low-pass smoothing filter to our images after spectral whitening. This filter has an isotropic Gaussian impulse response, like the 2-D window $w$ that we use for local cross-correlations, but with a smaller radius $\sigma = 1$ sample.

Figures 8 show the result of smoothing after whitening for our test images. Like those in Figures 7c and 7d, the correlation peaks in Figures 8c and 8d remain well-resolved and more isotropic than those in Figures 6c and 6d, while smooth enough to reduce artifacts caused by errors in quadratic fitting.

In addition to improving the accuracy of quadratic fitting, low-pass filtering has another benefit. Prediction error filtering tends to enhance high-frequency noise. Where this noise is not repeatable in time-lapse experiments, it will degrade estimates of displacements. Gaussian smoothing after prediction error filtering attenuates the higher frequencies.

Smoothing after prediction error filtering is common practice in seismic data processing. For example, spiking deconvolution, a form of prediction error filtering, is often followed by low-pass filtering of seismograms. We use the same processing here, but with multi-dimensional prediction error filters computed and applied seamlessly for each image sample.

### 3.4 Remaining problems

In all of the examples of Figures 6, 7, and 8, we computed cross-correlation values for multiple integer lags $[l_1,l_2]$, and then fit 2-D quadratic functions to correlation peaks.

Recall that the fitted quadratic function need not exactly interpolate any of the 9 samples nearest to the sampled maximum correlation value, because the bi-quadratic function has only six coefficients. Also recall that this misfit may be greater in 3-D, as we fit 27 sampled correlation values with only 10 tri-quadratic coefficients. Errors in quadratic fitting have been reduced but not eliminated in the example of Figures 8.

Another problem is the amount of memory needed

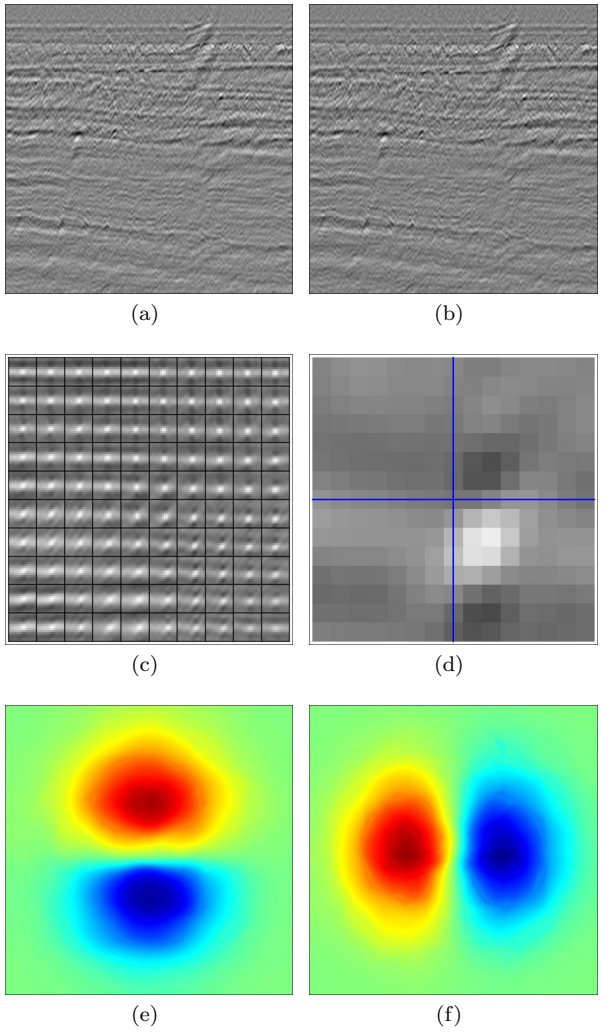


**Figure 8.** Images (a) and (b) after whitening with local prediction error filters and smoothing with a 2-D Gaussian filter. Smoothing attenuates high-frequency noise that is amplified by whitening, and facilitates location of the peaks of each of the phase-correlations shown in (c) and (d). Estimates of vertical and horizontal components of displacement in (e) and (f) are more accurate than those without smoothing in Figures 7e and 7f.

to hold all of the correlation values required for fitting. Recall that we compute for each lag $[l_1, l_2]$ an entire image of cross-correlation values $C_{fg}[k_1, k_2; l_1, l_2]$. As suggested by equation 2, this computation enables us to implement Gaussian windowing with efficient recursive filters.

As we iterate over lags, we must update for each image sample indexed by $[k_1, k_2]$ the lag $[l_1, l_2]$ for which a maximum sampled correlation value is found. For each sample, if the current correlation value exceeds the maximum value found so far, we update that maximum value and record the lag. After this first iteration over lags, the total number of correlation values computed is the product of the number of image samples times the number of lags.

This product can be a large number, especially for 3-D local correlations of 3-D images. Assuming that we do not store all of the correlation values computed in the first iteration, we must then recompute in a second iteration the sampled correlation values for lags nearest the lag with the maximum value. As we recompute those correlation values, we must update and store the coefficients of the quadratic polynomials required to locate correlation peaks. For 2-D images, 6 coefficients are required; for 3-D images, 10 coefficients.

Compared to memory required for other types of 3-D image processing, storage for 10 3-D volumes of coefficients is large but not prohibitive. And this factor of 10 is typically much less than the number of lags $[l_1, l_2, l_3]$ scanned in the search for correlation peaks. We need not store the correlation values for all lags scanned.

Nevertheless, as described in the following section, we can significantly reduce both the memory required and the number of correlation values computed, while eliminating any errors due to quadratic fitting.

## 4 CYCLIC SEARCH

Following the whitening-and-smoothing step describe above, the second step in our process is a cyclic sequence of correlations and shifts along each of the axes of our images.

### 4.1 Correlate-and-shift

We begin by cross-correlating two images in the vertical direction and finding the locations of peaks of those correlations. The peak locations that we find for each image sample correspond to one component of the displacement vectors that we wish to estimate.

We then shift one of the images using high-fidelity sinc interpolation to compensate for our estimated vertical components of displacements. This interpolation aligns the two images by applying spatially varying vertical shifts to one of them.

After compensating for vertical displacements, alignment is incomplete where horizontal components of displacement are non-zero. We therefore repeat this correlation and shifting to estimate and compensate for those horizontal components. After correlating and shifting for each image axis, we repeat the entire sequence of vertical and horizontal correlations and shifts until all shifts are negligible.

Figures 9 show estimated components of displacement for a four-cycle sequence of correlations and shifts. In each cycle we correlate and shift in both vertical and horizontal directions.

As we cross-correlate for one component of lag, say,

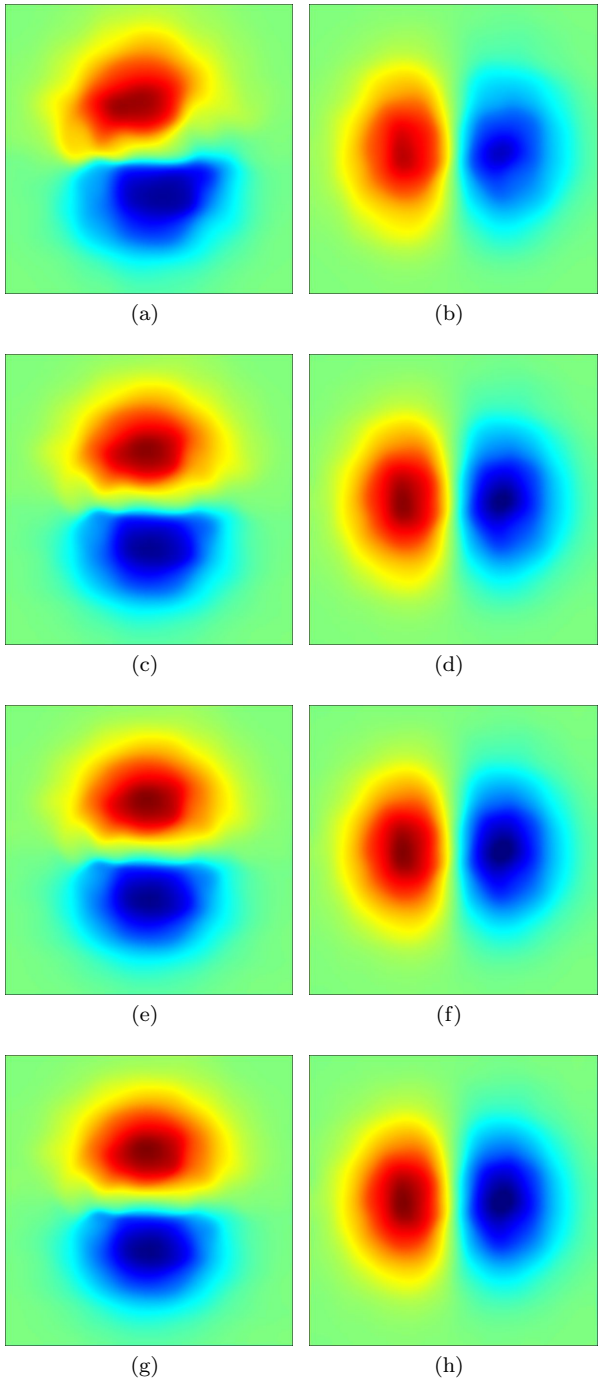


**Figure 9.** Four cycles of sequential estimation of two components of displacement for the images in Figures 8a and 8b. We first estimate vertical components of displacement (a). After shifting the image in Figure 8b vertically to compensate for these displacements, we estimate and compensate for horizontal components (b). Repeating this process, we obtain second (c and d), third (e and f) and fourth (g and h) estimates of displacements. Compare these final estimates with the known displacements in Figures 5.

$l_1$, the other components of lag are zero. In other words, we compute only the central column of normalized local cross-correlation functions like that displayed in Figure 8d. The images and the Gaussian correlation windows remain multi-dimensional; we use equation 2 just as before. But we restrict our computation of cross-correlation values to lags for which $l_2 = 0$.

In the example of Figure 8d, the maximum correlation value for $[l_1, l_2 = 0]$ occurs for lag $l_1 = 3$. So we would use quadratic interpolation of the sampled correlation values $c[2, 0]$, $c[3, 0]$, and $c[4, 0]$ with equation 5 to estimate the location of the correlation peak (somewhere between lags $l_1 = 2$ and $l_1 = 3$) with sub-pixel precision.

As we iterate over lags $l_1$, we need only keep the most recent three cross-correlation values for each image sample. These values correspond to lags $l_1 - 1$, $l_1$, and $l_1 + 1$. When the correlation value for the middle lag $l_1$ exceeds the values for the other two lags, we use equation 5 and quadratic interpolation to interpolate a correlation peak value. If that peak value exceeds the maximum peak value found so far, we update the maximum and the displacement $u_1$ at which the maximum occurs.

The resulting estimates of displacement shown in Figures 9g and 9h are the most accurate of all such estimates shown in this paper. Compare these estimates with the known displacements in Figures 5.

### 4.2 Why cyclic search is better

Several features make this cyclic sequence of correlations and shifts attractive.

First, because only three (not six or ten) correlation images are required to interpolate peaks, a cyclic sequence of one-dimensional searchs for correlation peaks requires less memory than the direct multi-dimensional search used for Figures 6, 7, and 8.

Second, each 1-D quadratic interpolation we perform to locate peaks is guaranteed to find a peak value within one-half sample of the integer lag at which the maximum sampled correlation value occurs. Recall that no such guarantee exists for bi-quadratic and tri-quadratic fitting of 2-D and 3-D correlation values; the best-fitting quadratic polynomial may be a saddle with no peak at all. In this aspect 1-D quadratic interpolation is more robust.

Third, the cyclic sequence eliminates errors due to quadratic interpolation, because those errors go to zero as the shifts converge to zero. In each correlate-and-shift cycle, we compute shifts with quadratic interpolation, but we apply these shifts using sinc interpolation. (Sinc interpolation is commonly used in seismic data processing to apply one-dimensional shifts that vary with time and space. An example is normal-moveout correction.) Therefore, errors in quadratic interpolation do not accumulate and are gradually eliminated.

Finally, in a cyclic search for correlation peaks, we may compute fewer correlation values than in an exhaustive search over all possible lags $[l_1, l_2]$ (or, in 3-D, $[l_1, l_2, l_3]$) displayed in Figure 8d. In each correlate-and-shift step of cyclic search, we compute only one column or one row of cross-correlation values marked by blue axes in Figure 8d.

Computational cost is proportional to the number of correlate-and-shift cycles required to align the two images, that is, for shifts to become negligible. When displacements are small, convergence requires few iterations. And as shifts decrease in later iterations, cost can be reduced by limiting the range of lags for which we compute correlation values.

Our cyclic search resembles iterative Gauss-Seidel solution of large sparse systems of linear equations, in which one iteratively solves one of many equations for one variable while holding constant the other variables. Cyclic search is also a well-known algorithm for optimization of functions of several variables. In that sense here we use cyclic search to maximize cross-correlation functions computed for every image sample.

In the 3-D example of Figures 1–4, we used two cycles of vertical-crossline-inline shifts. The shifts in the second cycle were large enough to be worth applying, but not so much as to warrant a third cycle.

### 4.3 Displacements from a sequence of shifts

As our cyclic search converges, the two images become well aligned, and the shifts tend toward zero. How do we estimate displacements from the shifts that we compute and apply in each iteration of cyclic search?

*Estimated components of displacements should not be simple sums of shifts computed and applied in each cycle.*

To understand how to compute displacements from a sequence of shifts, consider equation 1 for some unknown components of displacement $u_1$ and $u_2$. Then suppose that we have initial estimates $u_1^{(0)}$ and $u_2^{(0)}$ (which may be zero) for these components and a corresponding shifted image

$$h_0[j_1, j_2] = g\big(j_1 + u_1^{(0)}[j_1, j_2],\ j_2 + u_2^{(0)}[j_1, j_2]\big).$$

Cross-correlating images $f$ and $h_0$ for lags $[l_1, l_2 = 0]$, we estimate shifts $s_1$ that best align these two images vertically. We then use sinc interpolation to compute

$$h_1[j_1, j_2] = h_0\big(j_1 + s_1[j_1, j_2],\ j_2\big)$$

Cross-correlating images $f$ and $h_1$ for lags $[l_1 = 0, l_2]$, we estimate horizontal shifts $s_2$, which we again apply with sinc interpolation to obtain

$$h_2[j_1, j_2] = h_1\big(j_1,\ j_2 + s_2[j_1, j_2]\big).$$

Now suppose that this one cycle of two shifts has aligned the two images, that our cyclic search has converged such that $f[j_1, j_2] = h_2[j_1, j_2]$. How do we compute the components of displacements $u_1$ and $u_2$?

Combining equations in the sequence above,

$$\begin{aligned} h_2[j_1, j_2] &= h_1\big(j_1,\ j_2 + s_2[j_1, j_2]\big) \\ &= h_0\big(j_1 + s_1(j_1,\ j_2 + s_2[j_1, j_2]),\ j_2 + s_2[j_1, j_2]\big) \\ &= g\big(j_1 + u_1[j_1, j_2],\ j_2 + u_2[j_1, j_2]\big), \end{aligned}$$

where the components of displacements $u_1$ and $u_2$ are

$$u_1[j_1, j_2] = s_1\big(j_1,\ j_2 + s_2[j_1, j_2]\big) + u_1^{(0)}\big(j_1 + s_1(j_1,\ j_2 + s_2[j_1, j_2]),\ j_2 + s_2[j_1, j_2]\big)$$

and

$$u_2[j_1, j_2] = s_2[j_1, j_2] + u_2^{(0)}\big(j_1 + s_1(j_1,\ j_2 + s_2[j_1, j_2]),\ j_2 + s_2[j_1, j_2]\big).$$

It would be awkward and inefficient to compute displacements in this way, only after our cyclic correlate-and-shift sequence has converged. Instead we compute

$$u_1^{(1)}[j_1, j_2] = u_1^{(0)}\big(j_1 + s_1[j_1, j_2],\ j_2\big) \ + \ s_1[j_1, j_2]$$

$$u_2^{(1)}[j_1, j_2] = u_2^{(0)}\big(j_1 + s_1[j_1, j_2],\ j_2\big),$$

and then

$$u_1^{(2)}[j_1, j_2] = u_1^{(1)}\big(j_1,\ j_2 + s_2[j_1, j_2]\big)$$

$$u_2^{(2)}[j_1, j_2] = u_2^{(1)}\big(j_1,\ j_2 + s_2[j_1, j_2]\big) \ + \ s_2[j_1, j_2].$$

And because this single cycle of sequential shifts has converged,

$$u_1[j_1, j_2] = u_1^{(2)}[j_1, j_2]$$

$$u_2[j_1, j_2] = u_2^{(2)}[j_1, j_2].$$

More generally, in the $m$th iteration of cyclic search, we estimate shifts $s_m$ from local cross-correlations of images $f$ and $h_{m-1}$. If $m$ is odd, we then compute

$$h_m[j_1, j_2] = h_{m-1}\big(j_1 + s_m[j_1, j_2], j_2\big)$$

$$u_1^{(m)}[j_1, j_2] = u_1^{(m-1)}\big(j_1 + s_m[j_1, j_2], j_2\big) \ + \ s_m[j_1, j_2]$$

$$u_2^{(m)}[j_1, j_2] = u_2^{(m-1)}\big(j_1 + s_m[j_1, j_2], j_2\big);$$

otherwise, if $m$ is even, we compute

$$h_m[j_1, j_2] = h_{m-1}\big(j_1, j_2 + s_m[j_1, j_2]\big)$$

$$u_1^{(m)}[j_1, j_2] = u_1^{(m-1)}\big(j_1, j_2 + s_m[j_1, j_2]\big)$$

$$u_2^{(m)}[j_1, j_2] = u_2^{(m-1)}\big(j_1, j_2 + s_m[j_1, j_2]\big) \ + \ s_m[j_1, j_2].$$

We repeat this cyclic sequence of correlations and shifts until $s_m$ is negligible.

The least obvious result of this analysis is this: in the $m$th iteration of cyclic search we should use the computed shifts $s_m$ to interpolate not only the image $h_{m-1}$, but both estimated components of displacement

$u_1^{(m-1)}$ and $u_2^{(m-1)}$ as well, *before* adding those shifts to either $u_1^{(m-1)}$ or $u_2^{(m-1)}$.

In this way we iteratively computed the two components of displacement shown in Figures 9. We used a straightforward generalization of this cyclic sequence to estimate the three components of displacement shown in Figures 2–4.

The significance of interpolating displacements before accumulating shifts depends on the spatial variability of the displacements. Where displacements are constant, this interpolation is unnecessary. Where displacements vary, as in the examples shown in this paper, omitting their interpolation would yield biased errors. Whether small or large, these errors can be eliminated by the sequence of computations described above.

## 5 CONCLUSION

Our process for estimating apparent displacements from time-lapse seismic images consists of two steps: (1) spectral whitening and Gaussian low-pass filtering followed by (2) a cyclic sequence of local correlations and shifts.

This process exploits readily available tools for image processing. We use efficient recursive filters to achieve seamlessly overlapping Gaussian windows in our computation of local cross-correlations and for isotropic low-pass filtering. We use local cross-correlations to estimate displacements and to compute multi-dimensional local prediction error filters

The combination of cross-correlation after spectral whitening with prediction error filters approximates phase-correlation, a well-known tool used for image registration. Our adaptation of this tool enables a local phase-correlation function to be computed efficiently for every image sample.

The cyclic sequence of correlations and shifts is a natural extension of today's common estimation of vertical apparent displacements from time-lapse seismic images. Indeed, we typically begin by correlating and shifting in the vertical direction, because that direction is likely to yield the largest shifts. We then apply repeatedly the same simple, accurate and robust process commonly used today for the vertical dimension to all three spatial dimensions of 3-D images.

The one parameter that must be chosen with care in our process is the radius $\sigma$ of the Gaussian windows. Computational cost is independent of this radius, but the accuracy and resolution with which we can measure apparent displacements depends on it. Local correlations become less reliable as windows become smaller. Our ability to resolve changes in these correlations decreases as windows become larger.

In all of the examples shown in this paper, we have used $\sigma = 12$ samples. If we assume that a Gaussian is effectively zero for radii greater than $3\sigma$, then each of the 3-D correlation windows used for the example shown in Figures 1–4 contains almost $200,000$ samples. This number implies extensive averaging, and accounts in part for the smoothness in our estimated displacements.

Large windows do not however guarantee smooth displacements. For example, in Figure 6 estimated horizontal components of apparent displacement are discontinuous, implying infinite apparent strain. Where others (e.g., Rickett et al., 2005; Hall, 2006) have imposed smoothness constraints on estimated apparent displacements (in addition to using local windows for correlations) we have instead refined our processing to address the sources of these discontinuities.

While there is no guarantee that this improved processing will ensure sufficient accuracy or resolution, there is also no reason why this same processing could not be used in conjunction with the additional smoothness constraints developed by others.

## ACKNOWLEDGMENT

Thanks to Shell U.K. Limited and the Shearwater partnership (Shell, BP, and ExxonMobil) for providing access to their time-lapse seismic images and permission to publish results derived from them.

## REFERENCES

Balci, M., and H. Foroosh, 2006, Subpixel estimation of shifts directly in the Fourier domain: IEEE Transactions on Image Processing, **15**, 1965–1972.

Deriche, R., 1992, Recursively implementing the Gaussian and its derivatives: Proceedings of the 2nd International Conference on Image Processing, Singapore, 263–267.

Hale, D., 2006, Recursive Gaussian filters: CWP Report 546.

Hall, S., 2006, A methodology for 7D warping and deformation monitoring using time-lapse seismic data: Geophysics, **71**, O21–O31.

Hatchell, P. and S. Bourne, 2005, Measuring reservoir compaction using time-lapse timeshifts 75th Annual International Meeting, SEG, Expanded Abstracts, 2500–2504.

Hoge, W. S., 2003, A subspace identification extension to the phase correlation method: IEEE Transactions on Medical Imaging, **22**, 277–280.

Kuglin, C. D., and D. C. Hines, 1975, The phase correlation image alignment method: IEEE Proceedings of the International Conference on Cybernetics and Society, 163–165.

Rickett, J.E., L. Duranti, T. Hudson, and N. Hodgson, 2006, Compaction and 4-D time strain at the Genesis Field: 76th Annual International Meeting, SEG, Expanded Abstracts, 3215–3219.

van Vliet, L., Young, I., and Verbeek, P. 1998, Recursive Gaussian derivative filters: Proceedings of the International Conference on Pattern Recognition, Brisbane, 509–514.