Appendix A

3-D Interpolation Applied to Passive Data

A.1 Interpolation

Jon Claerbout (?) developed a 2-D interpolation scheme based on a mono-planewave assumption. To fill in a gap between two traces, a spatial prediction method is used to determine the dip, and the missing traces are constructed by combining the two known traces from either end of the gap, with time shifts and weights appropriate to the estimated dip and the various distances.

In three dimensions, the location of a trace to be filled in by interpolation will not always lie on a line drawn between some pair of input traces. Thus while it was sufficient in 2-D interpolation to find the *apparent* dip between two traces, in 3-D interpolation we need to find the *true* dip. A pair of traces is insufficient for finding the true 3-D dip. At least three traces must be used, since three points are required to define a plane.

Given this realization, a simple 3-D analog to Claerbout's 2-D mono-planewave scheme can be proposed. To construct a trace at a location, examine the three nearest traces. Use the spatial predictor described in Claerbout's paper to find the dip between one pair of traces, then another pair (with a different source-receiver azimuth). From these two apparent dip measurements, the true dip can be recovered and used for interpolation, following Claerbout.

I develop a slightly more general alternative to this simple approach, one that uses an arbitrary number of neighboring traces and is less susceptible to problems that might be caused by a single bad trace.

A.1.1 Implementation

The steps in Claerbout's 2-D method can be summarized as follows:

- Identify a gap of missing or dead traces.
- Estimate coherence between the two traces at the ends of the gap, for all possible dips, using small overlapping time windows.
- For each time window, pick the best dip.
- Fill in each dead trace with a sum of the two end traces, each time shifted according to the dip, and weighted by distance.

In 3-D there are some additional complications. The location at which we wish to construct a trace by interpolation may not lie on a line connecting two live traces, particularly if the spatial sampling is irregular. When there is such a line, we could follow the 2-D logic and apply the prediction scheme to the two traces. This would give an estimate of the dip along that profile — the apparent dip, not the true 3-D dip. The true dip and apparent dip are related by (Slotnick, 1959):

$$\sin\phi' = \cos(\theta - \theta')\sin\phi, \tag{A.1}$$

where ϕ is the true dip, ϕ' is the apparent dip, and θ and θ' are the azimuth angles of the down-dip direction and the direction along which the apparent dip is measured, respectively.

We could shift the two traces using this apparent dip and sum. A better idea, however, and a necessity for the case of irregular spatial sampling, is to estimate instead the true dip, using a number of nearby traces. We need at least three; from two we can only find the apparent dip in one direction.

We take the three or more nearest traces, two at a time, and apply the spatial prediction operator, as in the 2-D case. This gives us the coherence as a function of the apparent dip along the direction joining the two traces. Having done these computations for each trace pair, we then loop over all possible 3-D dips (a two-dimensional space, p_x vs. p_y). For each dip, given the orientation of each trace pair, we compute what apparent dip we would see given a particular true dip. The coherence for this apparent dip is extracted from the table constructed earlier, and the coherencies are summed for all trace pairs. The result is a 2-D image of what we will call "generalized coherence" as a function of p_x and p_y. From this, we pick the best dip.

Then, for each of the neighboring traces, given the true dip, we compute the apparent dip along the line joining the trace and the location of the trace to be interpolated. This gives for each trace a time shift. The three or more neighboring traces are time shifted, weighted based on their relative distances, and summed to produce the interpolated trace.

The steps in the 3-D scheme can be summarized as follows:

- For a given output point, find the N nearest live traces.

- Estimate coherence between each pair of traces in the group, for all possible apparent dips, using small overlapping time windows. Tabulate the results.
- Loop over all 3-D dips (true dips), parameterized by p_x and p_y. For each (p_x, p_y) pair:
    - Given the true dip, compute the apparent dip between each pair of traces.
    - Extract the coherence for that apparent dip from the tables computed earlier.
    - Add coherencies for all trace pairs to get a generalized coherence for this dip.
- Scan over all (p_x, p_y) a second time to see which dip has the best coherence. Pick the best dip.
- Construct the output trace as a sum of the N neighbors, each time shifted according to the true dip, and weighted by distance.

These steps are illustrated graphically in Figures A.1 and A.2.

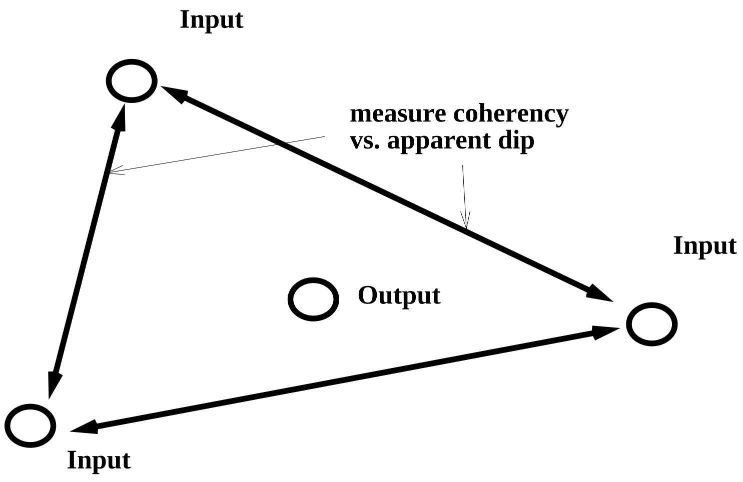

Figure A.1: Given the N nearest neighbors (here three), we first compute coherence as a function of the apparent dip between each trace pair. interp-idraw-step1 [NR]

A.1.2 Synthetic example

In this section, we test the algorithm on a simple synthetic dataset to illustrate its operation. Figures A.3 and A.4 show a number of slices taken through a synthetic 3-D cube, before and after interpolation. The cube contains 13x13 traces and 128 time samples. The geometry mimics the SEP passive experiment (Nichols et al., 1989) with a 38.1 meter spacing between traces in both directions.

The dataset contains four dipping events. One is near-vertically incident (apparent velocity 8 kilometers/second). A second has a low apparent velocity of 2 km/sec, and

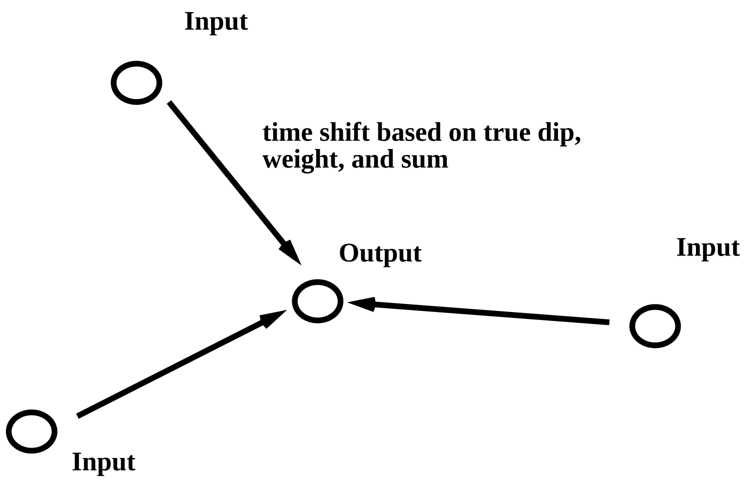

Figure A.2: After deciding on the best true dip, we compute the apparent dips between each neighbor and the output, then time shift, weight, and sum. interp-idraw-step2 [NR]

the third and fourth are close in both apparent velocity and azimuth angle - one has an apparent velocity of 3 km/sec and a down dip azimuth of 30 degrees (relative to the x axis) while the other has an apparent velocity of 4 km/sec at an azimuth of 60 degrees. Roughly half of the 169 traces were randomly removed prior to the interpolation, which was used to restore the missing data.

In this example, and in the real data example to follow, the five nearest traces were used to construct each interpolated trace. Since the contributing traces are weighted in inverse proportion to their distance from the output location, if the output location is the same as the location of one of the contributors, that trace gets a weight of one while all others get zero weight. In other words, the interpolated result honors the input data.

The interpolation has done a good job for the most part. Problems can be seen in a few places, such as in the crossline slice displayed at a larger scale in Figure A.5. There the top two events, the 8 km/sec and 2 km/sec events, cross. At the crossing point there is no problem, as whichever dip is picked works fairly well for both events. But there is a problem when the two events are near one another. Then they are close enough for both dips to occur in the same small computation window used by the program. But since only one dip is picked, the treatment of the second event can be poor.

The dips picked by the algorithm are tabulated and output by the program as an optional diagnostic. A contour plot of the dips picked for the synthetic is shown in Figure A.6. The 8 km/sec and 2 km/sec events can be seen at the center and bottom of the plot, respectively. The two events with quite similar dips give two maxima close together, in the upper right quadrant.

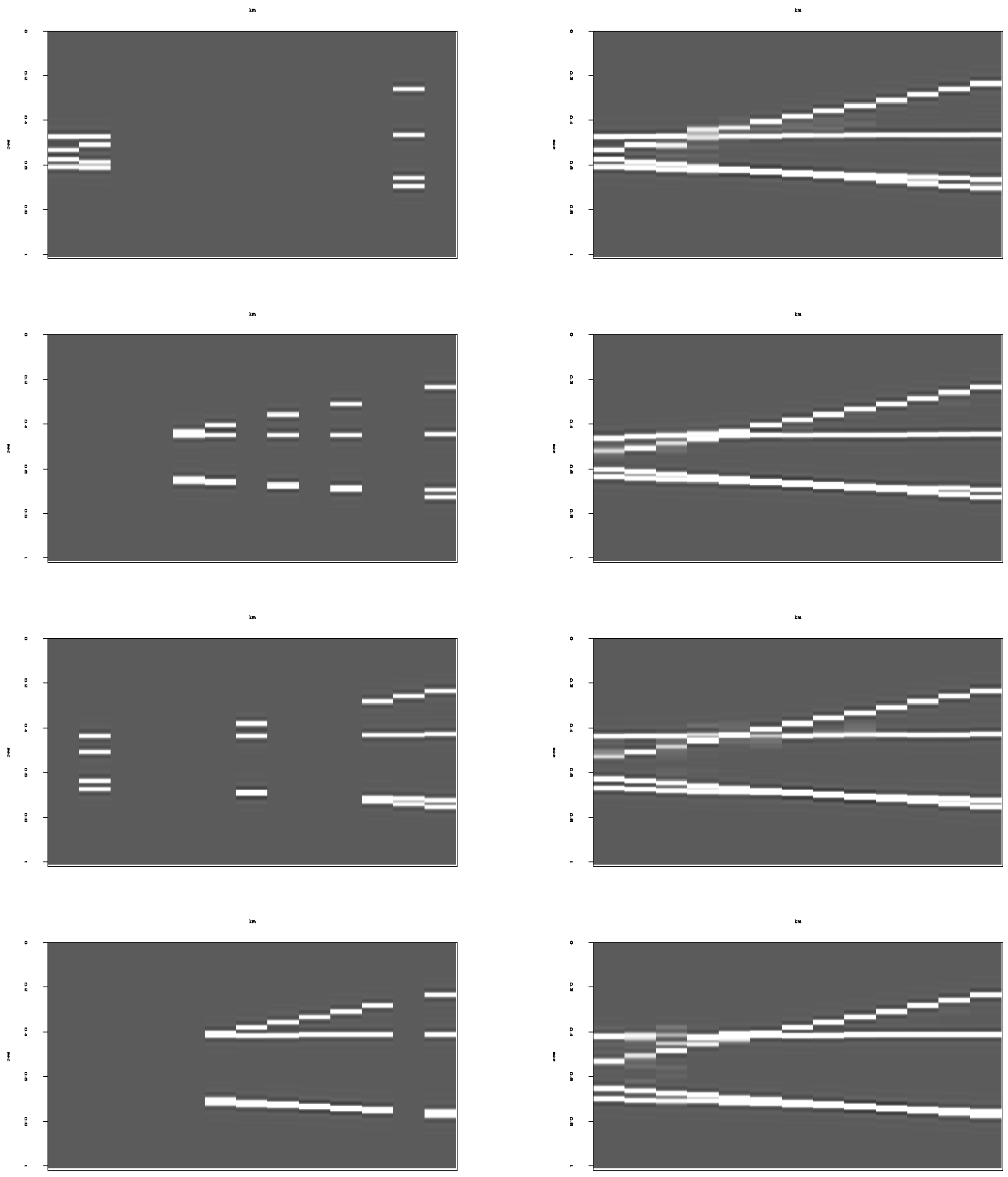

Figure A.3: Four crossline slices from synthetic dataset, before (left) and after interpolation. interp-synth-xlines [ER]

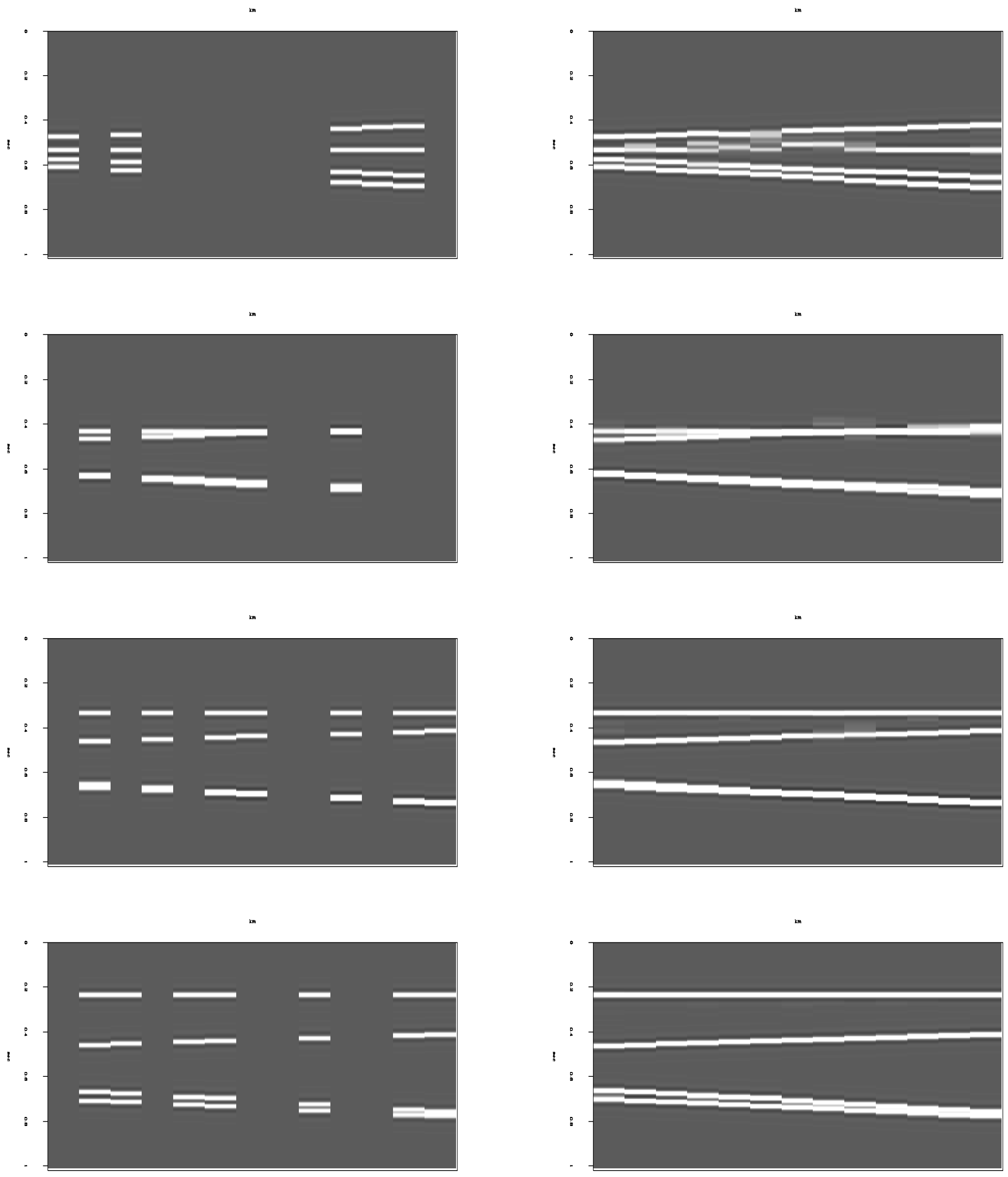

Figure A.4: Four inline slices from synthetic dataset, before (left) and after interpolation. interp-synth-inlines [ER]

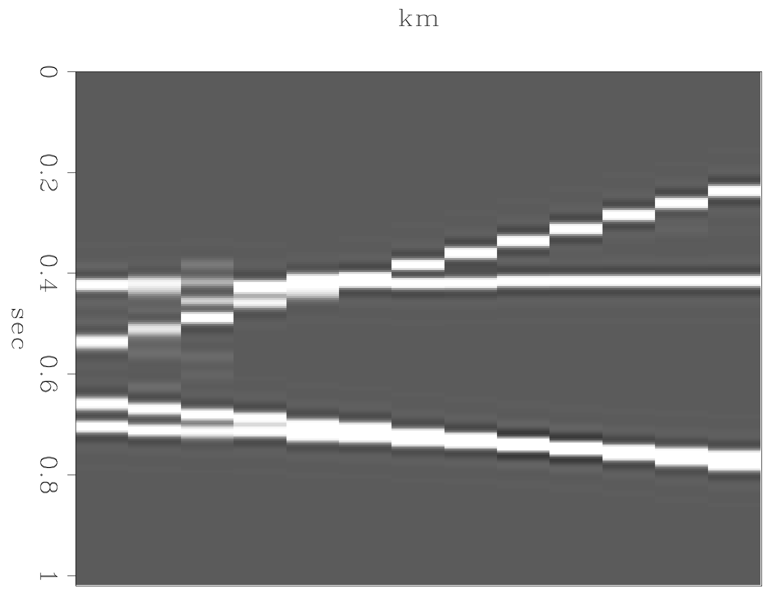

Figure A.5: A single crossline slice shown at a larger scale. The only significant error occurs just to the left of the point where the top two events cross. There the two dips occur in the same computation window, but only one is picked, and the treatment of the second one (the flatter event) is poor. interp-synth-1xline [ER]

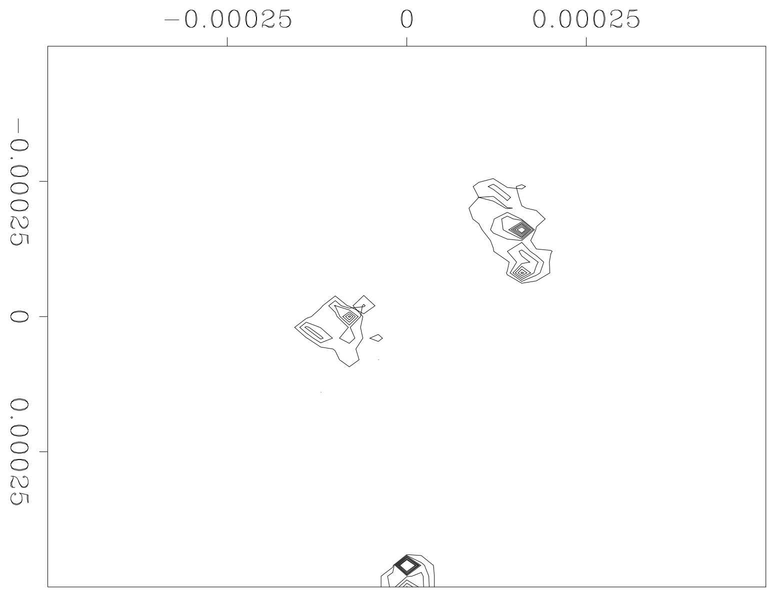

Figure A.6: Contour plot of dips picked by algorithm as a function of p_x and p_y. The 2 km/sec and 8 km/sec events are shown at the bottom and near the center. Although not well separated in apparent velocity or azimuth, the other two events are clearly distinguishable. interp-synth-picks [NR]

A.1.3 Quarry blast example

We have applied the method to a real 3-D dataset, the record from the SEP passive experiment containing the largest of the three quarry blasts.

The blasts were recorded late in the experiment, with many of the recording instruments beginning to fail because of dead batteries. At the time of the record used here, 87 of the 169 instruments had failed. The interpolation scheme gives us the opportunity to estimate the missing data. Actually the interpolations shown here, in Figures A.7 through A.10, are done onto a 25x25 grid, with half the original spacing, to make the data cube a little larger and easier to view.

The dips picked by the algorithm are shown in Figure **??**. Most of the picks are off to one side of the center of the plot — this indicates a dominant arrival direction, not surprisingly in the direction of the quarry. The range of dips for these arrivals is consistent with other analyses that have been done on the quarry blast data.

Note the significant number of picks around the edges, at very low apparent velocities. The prediction filtering used to estimate coherence as a function of dip is similar to a crosscorrelation of the two traces. Given a fixed data length, correlation uses more data for smaller lags than for larger lags (lower apparent velocities) where it must avoid going off the end of the trace. Coherence values for the lower apparent velocities, then, are based on less data; this introduces a bias that makes the extreme points more likely to be picked, we believe. This problem can easily be circumvented by padding the input data to allow for a uniform correlation length.

A.1.4 An alternate approach - plane fitting

Spatial prediction gives us the coherence as a function of apparent dip, measured along a number of azimuths. As an alternative to the dip scanning method used above, we could pick the best apparent dip for each trace pair, and then from these many apparent dips, select one (or more) true dips for interpolation. For example, we could fit a single plane to all the apparent dip measurements using least-squares. The overdetermined problem looks like this:

$$\begin{pmatrix} \cos\theta_1' & \sin\theta_1' \\ \cos\theta_2' & \sin\theta_2' \\ \vdots & \vdots \\ \cos\theta_N' & \sin\theta_N' \end{pmatrix} \begin{pmatrix} \sin\phi\cos\theta \\ \sin\phi\sin\theta \end{pmatrix} = \begin{pmatrix} \sin\phi_1' \\ \sin\phi_2' \\ \vdots \\ \sin\phi_N' \end{pmatrix} . \tag{A.2}$$

Here the unknows are ϕ and θ, the dip and azimuth angles of the true dip, the plane we are fitting to the data. The known ϕ_i' and θ_i' are the measured apparent dips and the azimuth angles along which they are measured.

In the presence of a single dip, this method works well, and is significantly less costly than a global search of the dip space. In the presence of multiple dips, however, the least-squares technique will choose an intermediate dip to minimize the error, a

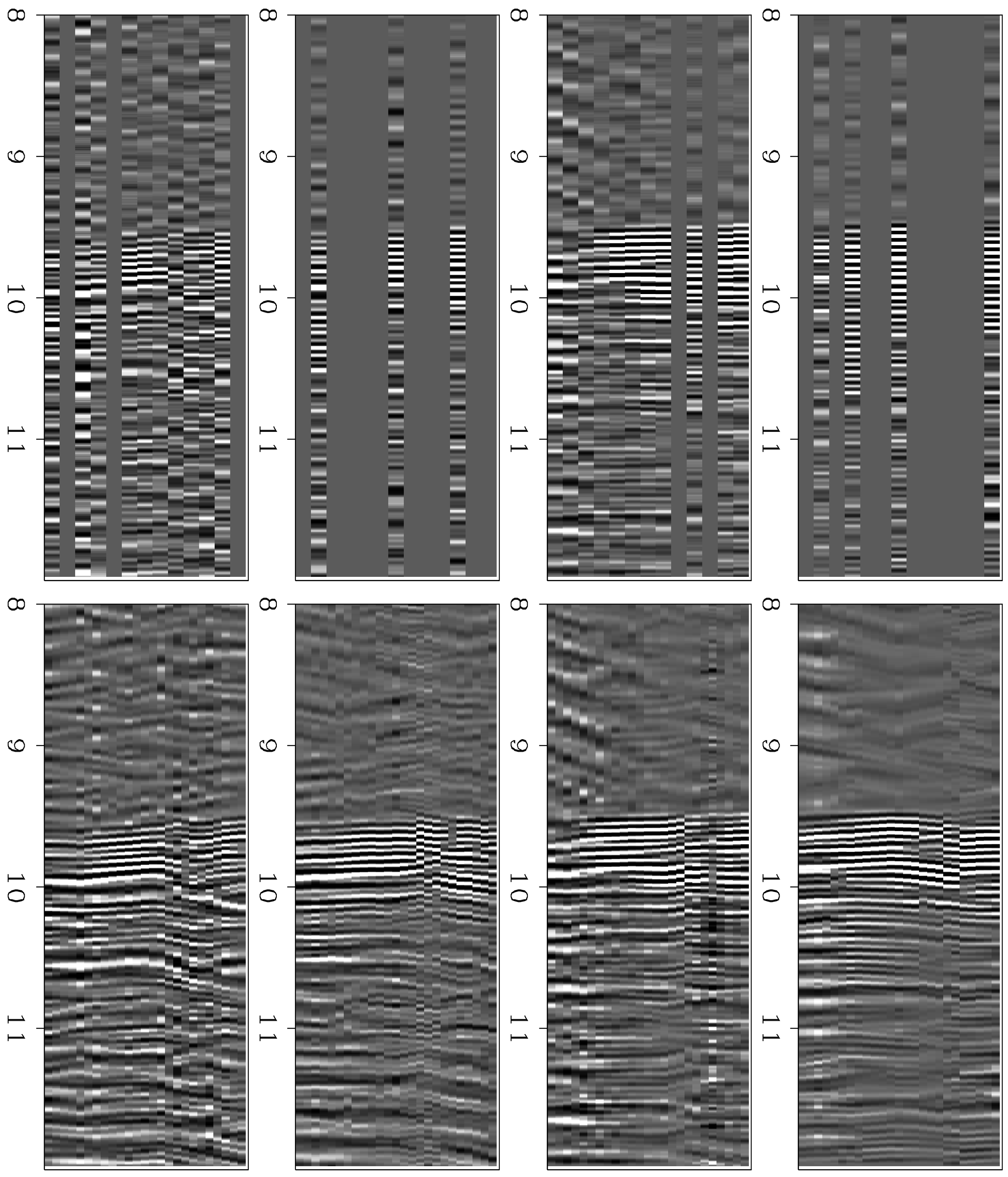

Figure A.7: Crosslines from quarry blast, before (top) and after interpolation. interp-blast-xlines [CR]

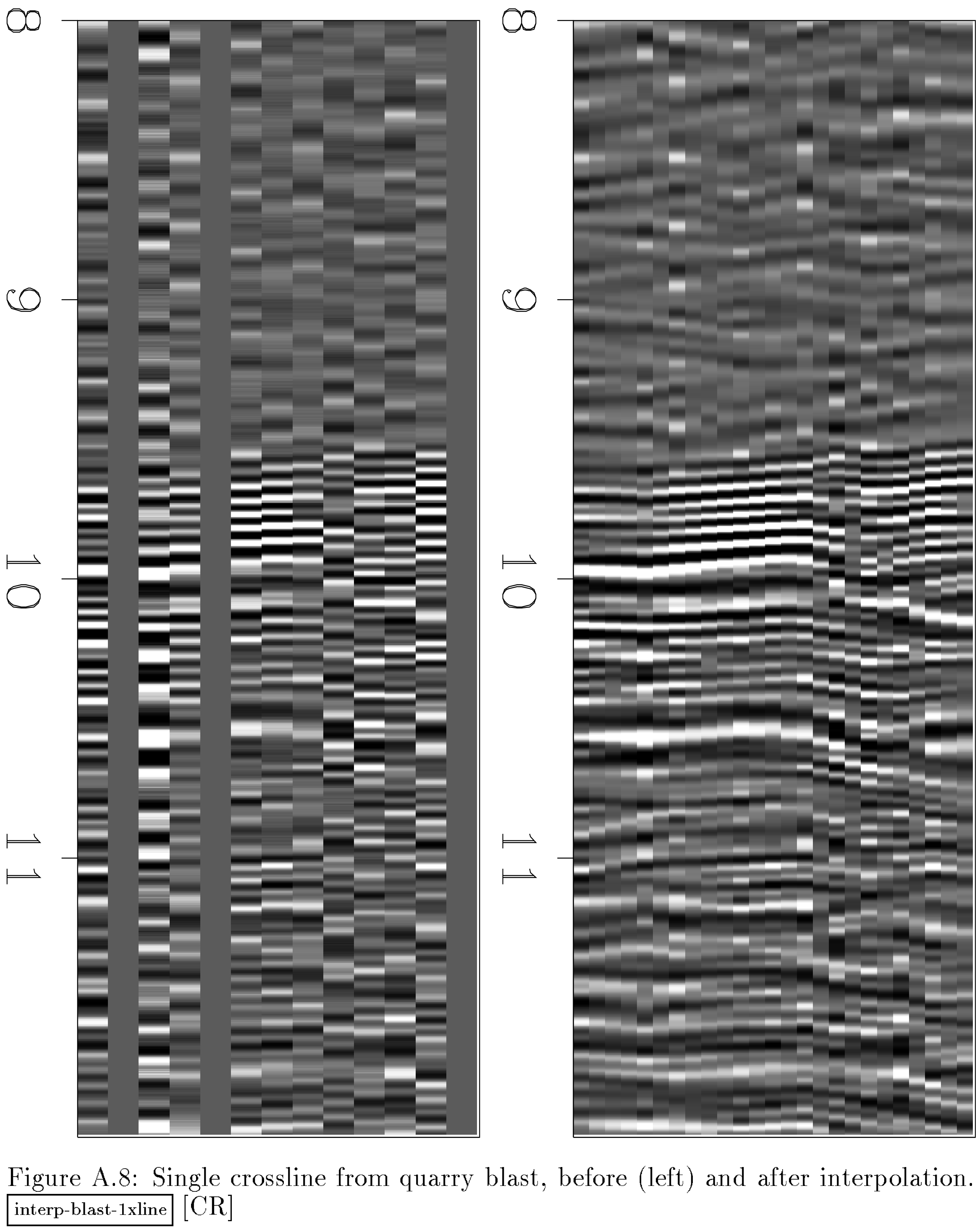

Figure A.8: Single crossline from quarry blast, before (left) and after interpolation. interp-blast-1xline [CR]

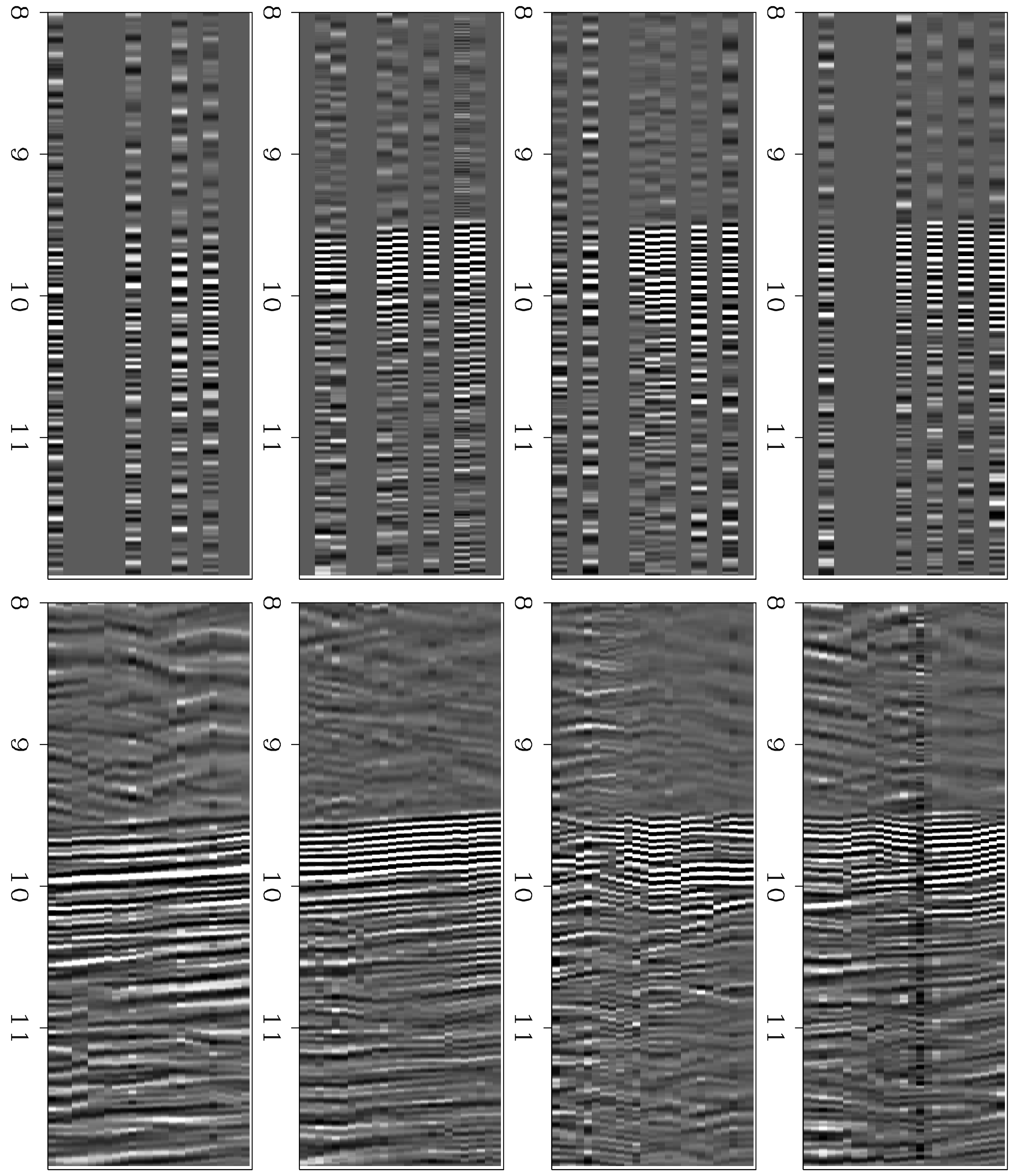

Figure A.9: Inline profiles from quarry blast, before (top) and after interpolation. interp-blast-inlines [CR]

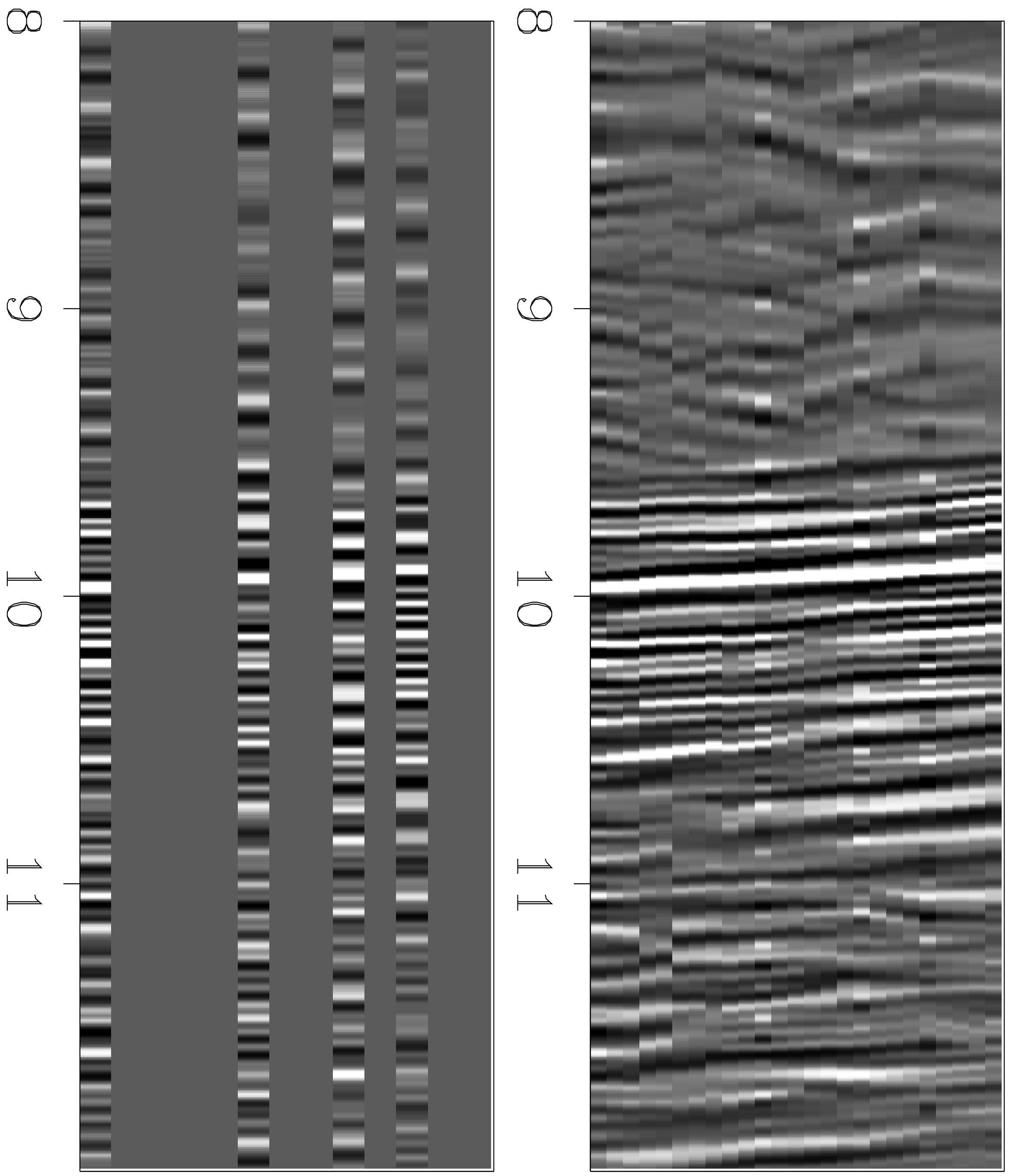

Figure A.10: Single inline profile from quarry blast, before (left) and after interpolation. interp-blast-1inline [CR]

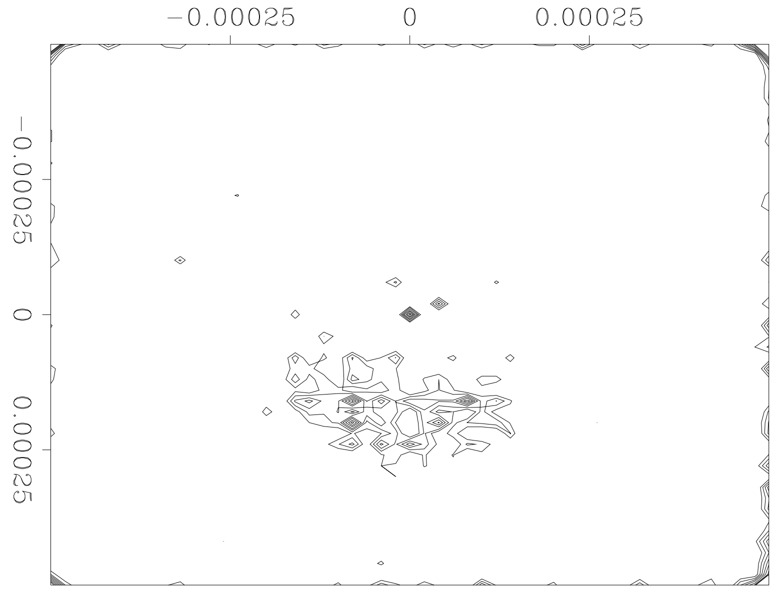

Figure A.11: Contour plot of dips picked by algorithm as a function of p_x and p_y. Most picks lie below the origin, indicating energy incident on the array from the south, the direction of the quarry. interp-blast-picks [NR]

dip which may not interpolate any of the dips particularly well. The dip-scanning method presented above will at least interpolate one dip, that which gives the best coherence, well.

The best solution to the case of conflicting dips may be a plane-fitting approach that fits multiple planes to the set of apparent dip measurements. Fitting multiple planes is significantly more difficult than fitting a single plane. Possibly an L1-norm scheme could help.

A.1.5 Application: removal of cross-line smear

The theory behind the interpolation scheme presented here could also be used to address the problem of cross-line smear in 3-D marine surveys, discussed by Yilmaz (?). Cable feathering causes midpoint locations to be distributed over the cells of a 3-D marine survey. If there is a significant amount of cross-dip in the subsurface, this midpoint scatter brings with it time shifts that cause a departure from hyperbolic moveout, and a smearing of the stacked amplitudes. Removal of the midpoint smear requires an estimate of the dip. The algorithm described here could be used to automatically determine the dip. Time shifts could be computed and applied to map all midpoints to the cell center.