Stanford Exploration Project, Report 115, May 22, 2004, pages 337–346

Short Note

Dynamic programming and trace alignment: Part 2

Chris Liner and Robert G. Clapp [1]

INTRODUCTION

Dynamic programming is an effective tool for finding a solution for certain types of relatively small, non-linear problems. In biology, dynamic programming is used for pairwise alignment of amino acid sequences (Needleman and Wunsch, 1970). In electrical engineering, it is used for error correction in wireless communication and speech recognition (Hosom et al., 1999) among many other things. We can also find examples of its use in geophysics. Kruse (1988) used dynamic programming for signal correlation and trace interpolation. He calculates an error function based on the difference in instantaneous frequency between all points along two signals. Dynamic programming is then used to find the error path with the least energy. Zhang (1991) used it for a starting solution when doing event picking.

In this paper we will continue the work started in Liner and Clapp (2002). We use dynamic programming for trace alignment. The Needleman-Wunsch algorithm (Needleman and Wunsch, 1970) proves effective when we expect significant stretch between compared traces, while alternate dynamic programming approaches can be more effective when we expect smoother variations and desire continuity between alignments.

In this paper we begin by discussing the basic tenets of dynamic programming and the specific features of both algorithms. We show how to extend the algorithms to seismic trace alignment and apply them on several different types of alignment problems.

DYNAMIC PROGRAMMING

Dynamic programming offers a way to solve certain classes of non-linear problems. It is useful for problems that can be thought of as making one decision after another. To understand how it works, it is easiest to start with our final decision. Our goal is to maximize our 'score' S of a series of decision $1...n$. Each decision has several potential outcomes $(1...m)$. Our final score is going to be based on some score we have calculated for all of our possible options 'states' at $j = n-1$, and the best score we can get from moving from all of the possible states at $n-1$

[1] **email:** bob@sep.stanford.edu, cll@utulsa.edu

to all possible states at n. We can write the score as

$$S(i,j) = max_{k=1...m}[S(k,j-1) + v(i,j,k)], \tag{1}$$

where i is the given state and $v(i,j,k)$ is the value obtained from moving from state k to state i at decision j. The best series of decisions is then found by going backwards, taking the state at each decision i that corresponded to the highest score.

In this most general form, the algorithm is quite expensive. The cost is on order $n*m*m$. For a large number of states the problem quickly becomes impractical. The easiest way to reduce the cost is to limit the number of states that we must search when moving from one decision to another. The next two subsections describe ways to limit the search that are practical for certain classes of problems.

Needleman-Wunsch algorithm

The Needleman-Wunsch (NW) algorithm (Needleman and Wunsch, 1970) is a method that was developed for amino acid sequence alignment in proteins. This was the first of many important alignment techniques which now find application in the Human Genome Project.

Human DNA consists of some 30,000 genes which are in turn composed of 20 amino acids represented by letters of a reduced alphabet (ADCEFGHILKMNPQRSTVWY). The total genome is composed of about 3 billion letters, or 100,000 per gene. Finding where a particular string of amino acids fits on a protein is an optimization problem that aims to find the optimal alignment of two character strings with respect to a defined set of rules and parameter values for comparing different alignments.

In terms of dynamic programming, we have two strings of length n and m that we would like to align. The key observation of the algorithm, and what makes it a computationally attractive process, is that the strings can be compressed and stretched but must remain in order. As a result, when comparing the two strings we have only to evaluate three possible states: a match, a compression, or a dilation.

To implement the NW algorithm we start by constructing a 'similarity' matrix σ of size n by m. In the genome example, the matrix is going to have binary values: either the amino acid matches (1) or it doesn't (0). We then construct our score matrix. The optimality of the path is defined by our similarity measurement σ. At each point in the scoring matrix, we evaluate whether the optimal path is from a point below (E), a point to the right (F), or a point along the diagonal (G). Our preferred solution is to move along the diagonal (we want to give preference to finding as much similarity between the two strings as possible) so we penalize the other two options. The scoring matrix (V) will then be recursively built by selecting the maximum of E, F, and G. We can write this in terms of the following equations,

$$\begin{aligned} V(i,j) &= max\left[G(i,j), F(i,j), E(i,j)\right] & (2)\\ G(i,j) &= \sigma(x_i, y_j) + V(i+1, j+1) & (3)\\ F(i,j) &= -(p+q) + max\left[V(i+1,j), F(i+1,j) + p\right] & (4)\\ E(i,j) &= -(p+q) + max\left[V(i,j+1), E(i,j+1) + p\right]. & (5) \end{aligned}$$

The parameters (p,q), often referred to as 'gap' penalties, are an important aspect of the algorithm. They allow the user to control how discontinuous a path to allow through the similarity matrix.

The NW algorithm has some nice properties. The cost is relatively low, $n*m*3$. In addition, it can handle very discontinuous traces. This can also be a disadvantage, sometimes we would like to force a greater level of continuity than the gap penalties allow.

Extension to seismic trace alignment

There are two obvious problems with a straightforward implementation of the NW algorithm. Both are related to the fact that our data is continuous, rather then discrete. We do not have the limited alphabet of the DNA example, so our similarity matrix can't simply be matching of values. Therefore, a more appropriate measure of similarity in the seismic case is a short-window correlation. We capture this idea in a similarity function as

$$\sigma(i,j) = \frac{\sum_{a=-n}^{n} x_{i+a} y_{j+a}}{\sum_{a=-n}^{n} x_{i+a} \sum_{a=-n}^{n} y_{j+a}} \tag{6}$$

or a semblance measure

$$\sigma(i,j) = \frac{\sum_{a=-n}^{n} \frac{(x_{i+a}+y_{j+a})^2}{(x_{i+a}^2+y_{j+a}^2)}}{2*n} \tag{7}$$

where x and y are again the two traces, and n is the correlation length.

The second problem is how to use our alignment data. We are not dealing with discrete points but smooth functions. We can estimate a prediction error filter (Claerbout, 1998) or a time variant, non-stationary prediction error filter upon the original data to describe the wavelet. We have adopted this approach, allowing us to estimate our aligned model **m** from our unaligned data **d** by minimizing

$$\min_{\mathbf{m}} = ||\mathbf{m}-\mathbf{d}||^2 + \epsilon^2 \; ||\mathbf{A}\mathbf{m}||^2 \tag{8}$$

where **A** is filtering with the estimated prediction error filter and ϵ is a scalar controlling how much weight to give the aligning. This controls the relative importance of the alignment versus fitting the wavelet.

NW EXAMPLES

To illustrate the application and results of NW seismic alignment, we present three examples. The first is alignment of events in synthetic P-wave and S-wave zero-offset sections. The second is flattening of events in a common image gather (CIG) from a marine 2D seismic

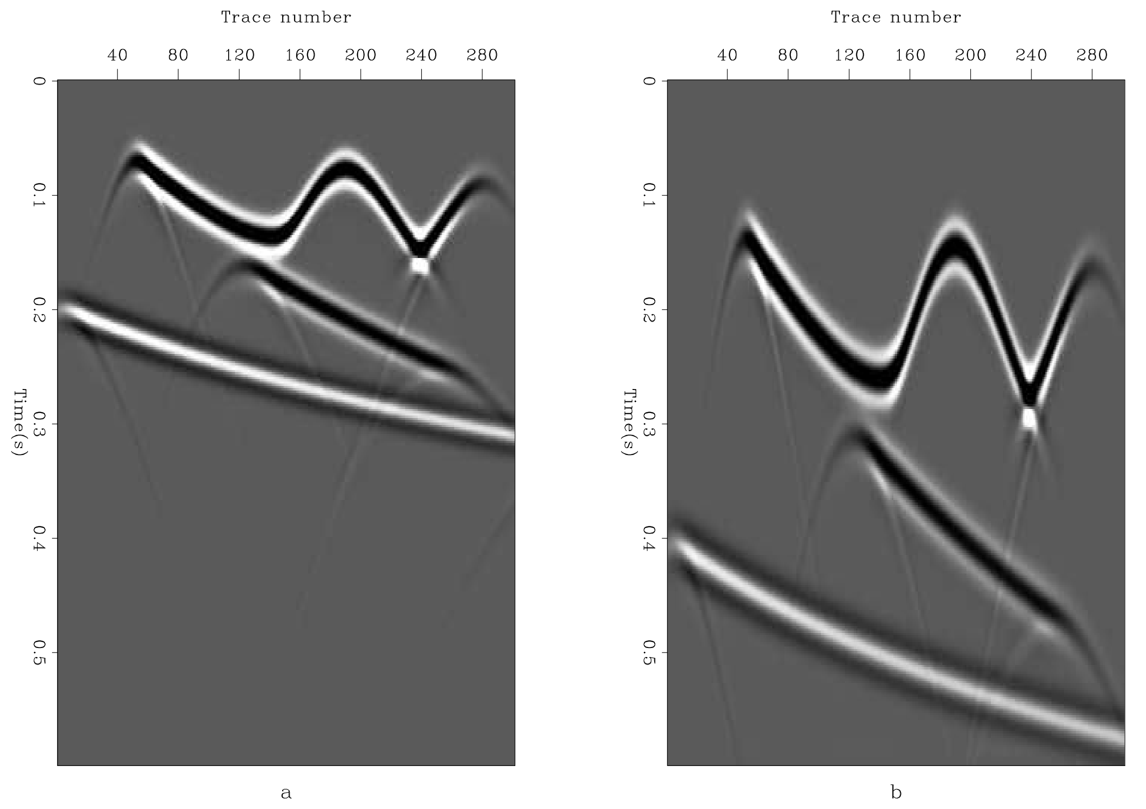

Figure 1: The left panel (a) shows a synthetic 3-layer P-wave section. The right panel (b) shows the S-wave section for the same earth model. Both simulations represent zero-offset data. bob1-sv.input [ER]

line. In the final case we attempt to align a real P- and S-wave section. For the first example, two synthetic zero-offset seismic sections (Figure 1) were generated by 2D acoustic Kirchhoff modeling. The data is moderately complex, containing low and steep dips, diffractions, and a buried focus. The subsurface reflector geometry is identical in each case. The S-wave section (Figure 1b) was modeled using the P-wave velocities and a VS-VP ratio that varied both laterally and with depth. This means a spatially-variant, nonlinear stretch is needed to align the sections. The alignment algorithm works here in a pairwise fashion on each trace pair that has the same trace number in each section. Every alignment is calculated independently, so there is no error propagation in this case.

The alignment result (Figure 2) represents the P-wave section after application of nonlinear stretch coefficients. We could just as easily have aligned the S-wave section to the P-wave data. The gap penalties for this example are $(p,q) = (0.0, 0.1)$ and $\epsilon = 10$. Comparison of the S-wave section (Figure 1) with the alignment section demonstrates the accuracy and sensitivity of this method. Considering the complexity of the input data, we feel the alignment is of excellent quality. For a second example, we chose a common reflection point gather from a 2D marine data set (left panel of Figure 3). This collection of traces forms an common image gather in the angle domain (Prucha et al., 1999; Sava and Fomel, 2000) after phase-shift plus-

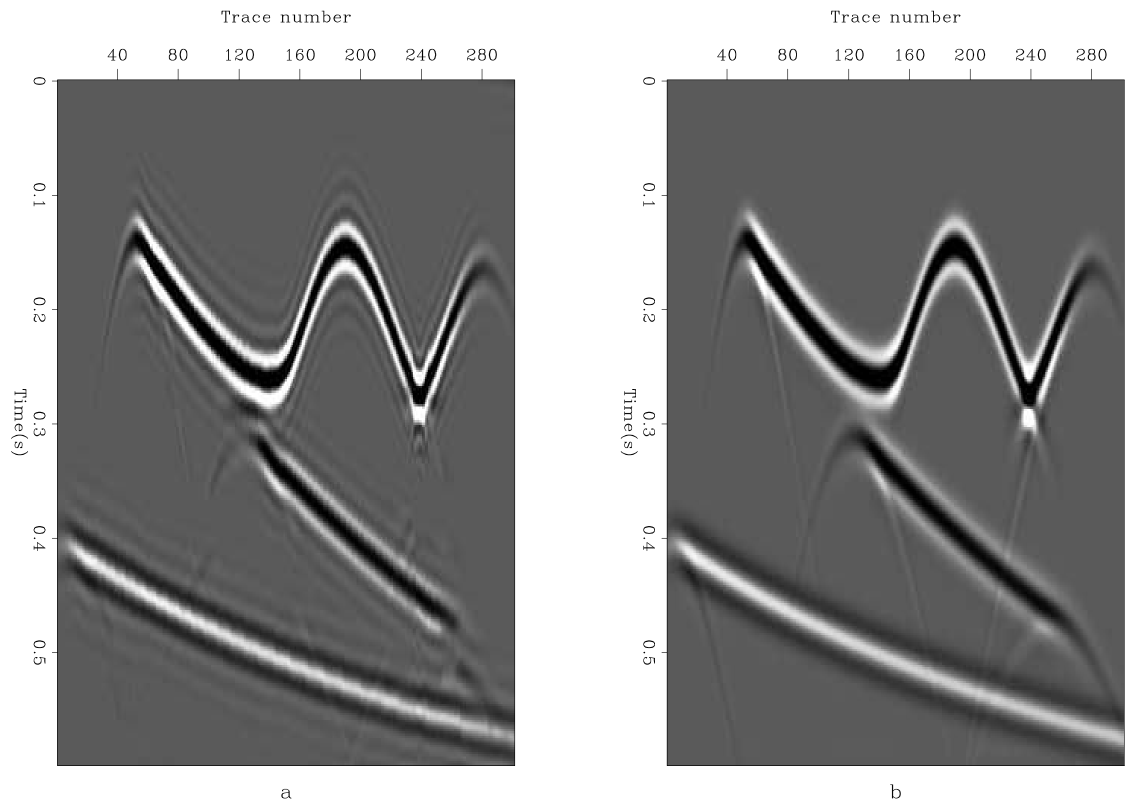

Figure 2: The left panel (a) shows the P-wave section after trace alignment based on the S-wave section from Figure 1a. The right panel (b) displays the S-wave section again for comparison. $\boxed{\text{bob1-sv.output}}$ [ER]

interpolation (PSPI) migration (Gazdag and Sguazzero, 1985). Note that we see some residual moveout in the CIG. The number of events, low-slope residual moveout, and lateral variations make this a significant alignment problem. The goal here is to align events in the gather with the first (zero-angle) trace. Constructing the aligned model in this case is far more complex than the procedure necessary for independent pairwise trace alignments. The effect of aligning trace two to trace one affects the alignment of traces three and two, etc. In our view, there are three potential ways around this connectivity. The first is to do the alignments sequentially. Align trace n-1 to trace n, then align both n and n-1 to n-2, and repeat the procedure until all traces are aligned. This procedure can be effective but is prone to error propagation. The second is to recognize that all possible pairs of traces can be aligned to each other and setup an inverse problem that takes all possible alignments into account. This would probably lead to the best solution but how best to relate the different combinations isn't obvious and the computational cost is significantly increased (we now have to compute and account for $\frac{n(n-1)}{2}$ comparisons). We chose a third approach to multi-trace alignment involving a linear operator that performs sequential shifting. The idea is to begin at an initial trace (for example, the near offset) and to work outward toward the final trace (far offset). As each trace pair is aligned, these alignment coefficients (ACs) are applied to all remaining traces. In this way, an intermediate trace receives many partial alignments, and one last alignment from its nearest neighbor which itself has been modified several times. The algorithm can be summarized as follows:

```
for trace i from 1 to (n-1) {
    for trace j from (i+1) to n {
         modify trace j using ACs between traces i and (i+1)
    }
}
```

This method evolved from extensive testing of alternative approaches, each of which introduced serious error propagation problems.

In this case, single application of the algorithm was not sufficient. In order to obtain a reasonable solution, a larger gap penalty $(p,q) = (0.1, 0.5)$ was found to be necessary, along with $\epsilon = 100$ as in the CMP test. The resulting section still had residual moveout. As a result we reversed the output of the first alignment along the time axis and used this as input to a second run of the alignment procedure. Total compute time on a workstation-class machine was 25 seconds. The right panel of Figure 3 shows the result after the two-pass alignment. The flattening is not perfect, but the events are much better aligned than before while maintaining the same amplitude versus angle (AVA) characteristics.

For the final example, we attempted to align a PP and PS stacked section from the Alba field in the North Sea. Figure 4 shows the PP and PS section before alignment. This problem poses significant additional problems compared to the synthetic used in the first example. We are dealing with much higher frequency data than the first example and have many more events. The left panel of Figure 5 shows the output of the alignment procedure. Note the jumps in the sections, and the gaps at certain CMP locations. The first problem is an inherent property of the algorithm. The penalty we can impose for jumps is limited. The second problem is because we are performing a series of independent alignments. The result of the previous alignment has no affect on the current alignment. As a result we get the inconsistencies seen in the image.

BACK TO DYNAMIC PROGRAMMING

There are several ways we can try to improve the result seen in the left panel of Figure 5. We can think of smoothing the paths through the score matrix. We can also make some reasonable *a priori* assumptions. We can add to our similarity matrix. We might put large weights in the matrix by saying that time 0 should be the same in the PP and PS section or by visually matching a single reflector. These approaches improve our result, but still lead to something less than ideal.

A better solution is to return to a more generic form of dynamic programming. We can improve our 1-D estimates by remembering that what we are really doing is estimating a $\frac{v_p}{v_s}$ ratio. This ratio has to be a continuous function of time, as a result it is not reasonable to have large gaps in our trace alignment. First, if we consider the traces to be samples in a PP gather,

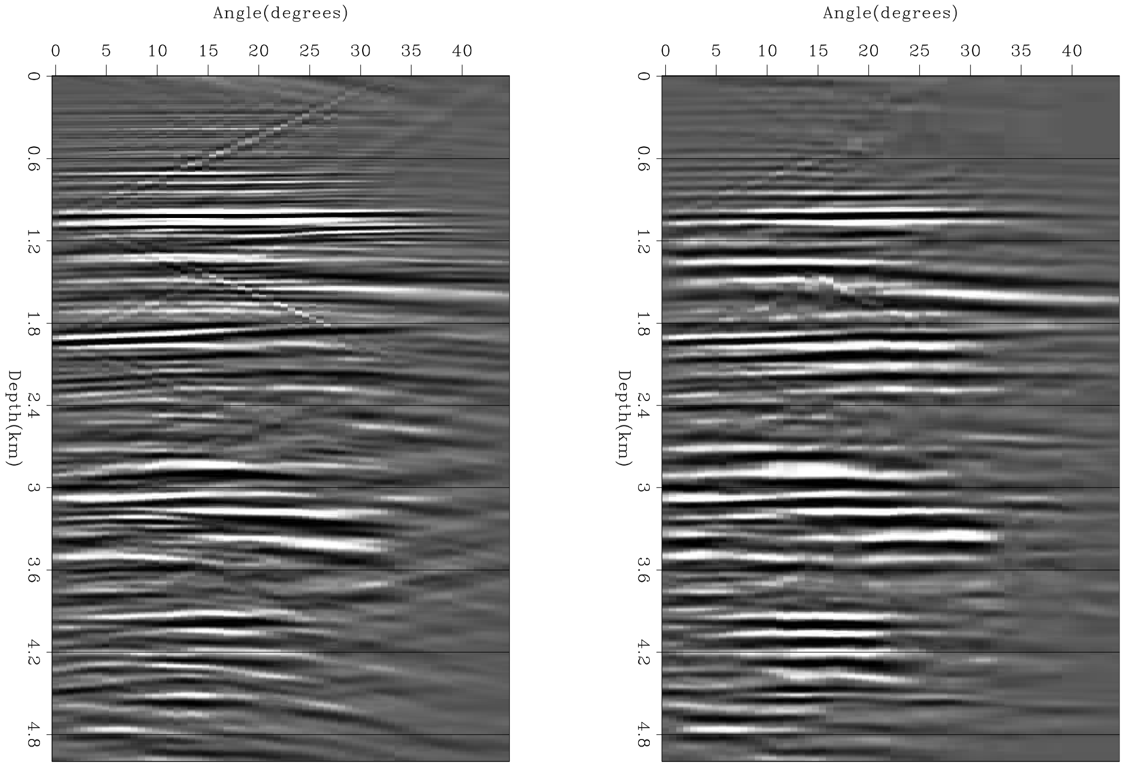

Figure 3: A CIG gather after PSPI migration and conversion to angle. Note that the gather still has significant residual moveout. The CIG gather shown ins then right panel after multi-trace alignment. Note how almost all residual moveout has been removed while preserving relative amplitude variations. bob1-crp [ER]

we can assume that the our alignment function will be single valued. As a result, we can make our series of decisions one PP time element at a time. Changing our decision definition takes away the nice feature of the NW algorithm that we only have to consider three states at each decision. On the other hand, we can limit ourselves to considering the subset of states that correspond to a reasonable $\frac{v_p}{v_s}$ path. We know that each PP sample must correspond to the previous PP sample $+x$, where x is a very small number. As a result, we again need to only consider two to four states at each decision. The right panel of Figure 5 shows the result of applying this dynamic programming concept, along with the constraints discussed above, to the Alba alignment problem. Note how the solution is better than the result shown in the left panel of Figure 5, but still could be improved.

CONCLUSION

We have shown how dynamic programming can be applied to a series of seismic trace alignment problems. For many cases, a straight implementation of an algorithm developed for protein sequences is desirable. For problems with relatively high coherent noise, and when extending from 1-D to multiple dimensions, more complex dynamic programming algorithms

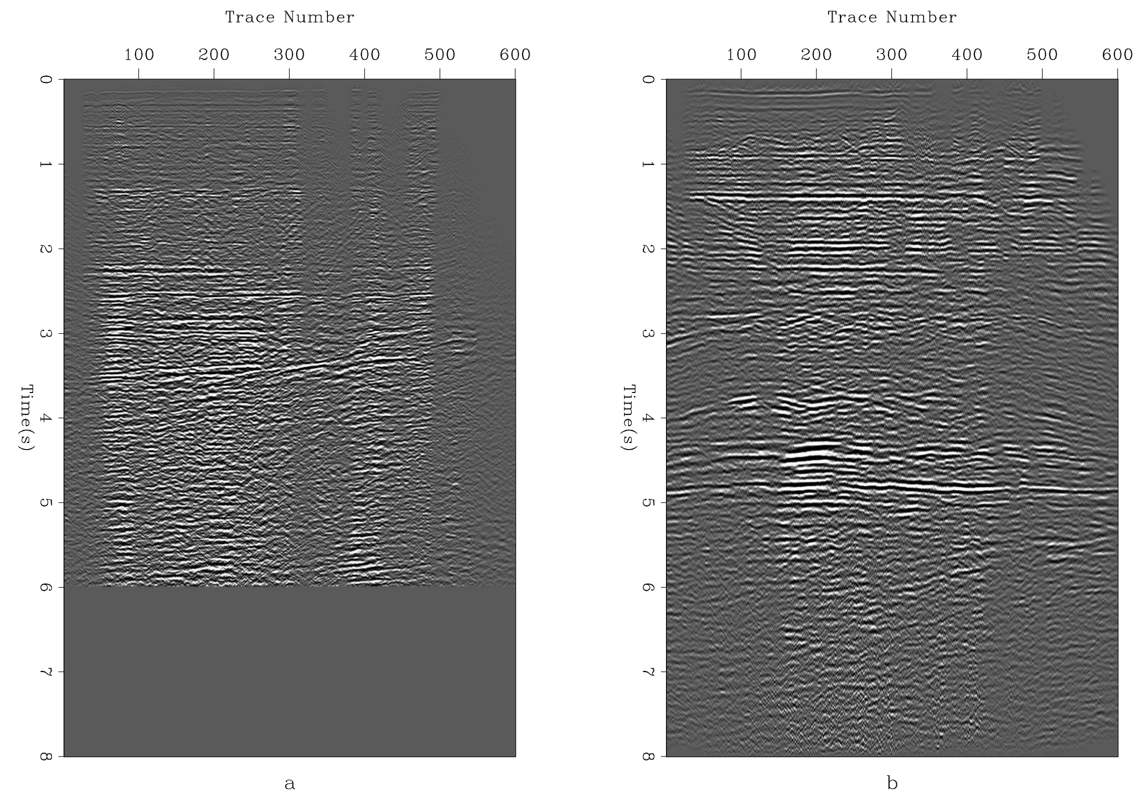

Figure 4: The left panel shows a PP section from the Alba field in the North Sea. The right panel shows a PS section from the same line. bob1-alba.in [CR]

can be useful.

With further work, this approach may supply a general tool for nonlinear alignment of seismic traces for use in processing and interpretation.

ACKNOWLEDGMENTS

We would like to thank ChevronTexaco for providing the data and Daniel Rosales for the preliminary processing of the data used in this paper.

REFERENCES

Claerbout, J., 1998, Geophysical Estimation by Example: Environmental soundings image enhancement:.

Gazdag, J., and Sguazzero, P., 1985, Migration of seismic data by phase shift plus interpolation, *in* Gardner, G. H. F., Ed., Migration of seismic data: Society Of Exploration Geophysicists, 323–330.

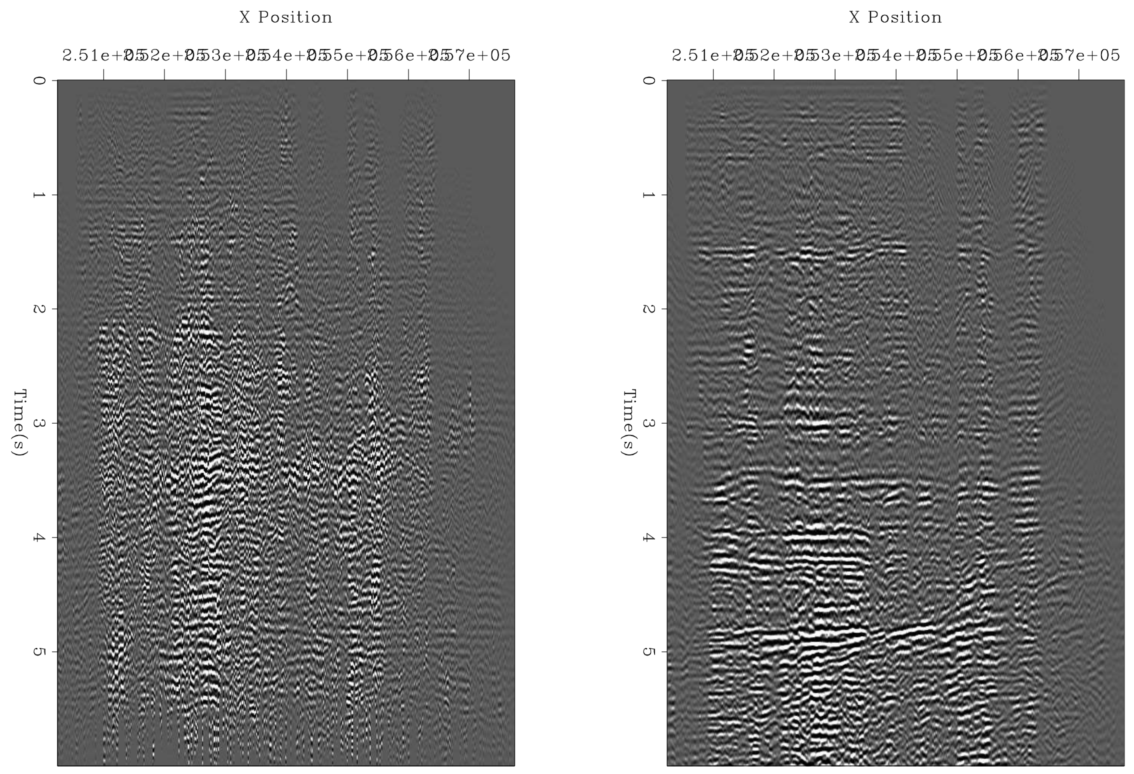

Figure 5: The PP section from the Alba field in the North Sea after alignment. Note the jumps in both time and CMP. bob1-alba.out [CR]

Hosom, J.-P., Cole, R., and Fanty, M. Speech recognition using neural networks at the center for spoken language understanding:. http://cslu.cse.ogi.edu/tutordemos/nnet_recog/recog.html, 1999.

Kruse, C., 1988, A new method of nonlinear signal correlation using the instantaneous spectrum: 68th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, pages 1321–1323.

Liner, C. L., and Clapp, R. G., 2002, Nonlinear pairwise alignment of seismic traces: SEP–**112**, 171–180.

Needleman, S., and Wunsch, C., 1970, A general method applicable to the search for similarities in the amino acid sequence of two proteins: J. Mol. Biol., **48**, 443–453.

Prucha, M., Biondi, B., and Symes, W., 1999, Angle-domain common image gathers by wave-equation migration: 69th Ann. Internat. Meeting, Soc. Expl. Geophysics, Expanded Abstracts, 824–827.

Sava, P., and Fomel, S., 2000, Angle-gathers by Fourier Transform: SEP–**103**, 119–130.

Zhang, L., 1991, Automatic picking and its applications: SEP–**70**, 275–292.