Stanford Exploration Project, Report 115, May 22, 2004, pages 357–366

Improved image segmentation for tracking salt boundaries

Jesse Lomask, Biondo Biondi and Jeff Shragge[1]

ABSTRACT

Normalized cut image segmentation can be used to track salt boundaries within an iterative velocity analysis scheme. To overcome the formidable computational expense and storage requirements of normalized cut image segmentation, three cost saving approaches are proposed. First, pixels are sampled from windows centered at powers of 2. This greatly increases the sparseness of the weight matrix. Second, initial solutions are provided to subsequent segmentations for multiple segmentation passes in iterative velocity analysis. Third, an iterative multi-scale approach would be necessary for tracking of the bright salt events in large 3D cubes.

INTRODUCTION

An accurate velocity model is needed to image beneath a salt body. This velocity model is typically created by manually picking the top of the salt. Improvements are then made to the velocity model and the data is remigrated causing the salt boundary to move and refocus. This process may be repeated several times and the manual picking of the salt boundary can be time consuming. Using image segmentation to pick the boundary could make this process easier. The interpreter is then only required to check the results and make some minor modifications.

Hale and Emanuel (2003, 2002) apply the normalized cut image segmentation method developed by Shi and Malik (2000) to paint a 3D coherency-based reservoir model. Our approach for tracking salt boundaries is similar (Lomask, 2003). This image segmentation technique creates a matrix containing weights relating each pixel to every other pixel in a local neighborhood. The matrix is then used to cut the image where the normalized sum of weights cut is minimized. We have modified the weight calculation to be dependent on the negative absolute value of the complex trace (instantaneous amplitude) of the seismic. This makes the weights very weak at salt boundaries, causing the segmentation algorithm to cut along the boundary.

In this paper, we give a very general review of the normalized cut segmentation technique. We then describe how we modified it for application to salt dome seismic data. We test this technique on synthetic seismic sections to illustrate its efficacy with discontinuous salt boundaries. Approaches to make this method more robust and cost effective are also presented.

[1]**email:** lomask@sep.stanford.edu, biondo@sep.stanford.edu, jeff@sep.stanford.edu

SEGMENTATION METHODOLOGY

The normalized segmentation method described by Shi and Malik (2000) is designed to look for clusters of pixels with similar intensity. To do this, a weight matrix is created that relates each pixel to every other pixel within a local neighborhood. The strongest weights are given to pixels of similar intensity and close proximity. The method then seeks to partition the image into two groups, A and B, by minimizing the normalized cut:

$$\mathrm{N_{cut}} = \frac{\mathrm{cut}}{\mathrm{total_A}} + \frac{\mathrm{cut}}{\mathrm{total_B}}, \tag{1}$$

where cut is the sum of the weights cut by the partition. $total_A$ is the sum of all weights in Group A, and $total_B$ is the sum of all weights in Group B. Normalizing the cut by the sum of all the weights in each group prevents the partition from selecting overly small groups of nodes.

The minimum of N_{cut} can be found by solving the generalized eigensystem:

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y}, \tag{2}$$

created from weight matrix (**W**) and a diagonal matrix (**D**), with each value on the diagonal being the sum of each column of **W**. The eigenvector (**y**) with the second smallest eigenvalue λ) is used to partition the image by taking all values greater than zero to be in one group, and its complement to be in the other.

APPLICATION TO SEISMIC DATA

To apply this segmentation method to seismic data, the weight calculation needs to be modified. Rather than looking for clusters of pixels with similar intensity, we are now looking for groups of pixels on each side of the bright amplitude salt boundary. Therefore, we want the weights connecting pixels on either side of the salt boundary to be low and the weights connecting pixels on the same side of the salt boundary to be relatively high. We get the best results when the weight for every pair of pixels is made weak(set to zero) when the following two criteria are met, otherwise the weight is made strong(set to one). The first criteria is that the minimum negative instantaneous amplitude along the shortest path between pixels is below a threshold. The threshold in our examples thus far is 85 percent of the maximum negative amplitude. The second criteria is the minimum is less than the two pixels themselves. The second criteria insures that the minimum is actually located between the two pixels. This is illustrated in Figure 1.

The top panel of Figure 2 shows the negative of the instantaneous amplitude of a synthetic salt boundary. This is the input for the weight calculation. The resulting weight matrix is extremely large: $(\mathtt{m} \times \mathtt{n})^2$ where $\mathtt{m}$ and $\mathtt{n}$ are the dimensions of the image. Because this matrix is so large, checking the quality of weights is problematic. We can however view all of the weights for particular pairs of pixels. For instance, the bottom of Figure 2 is the weight

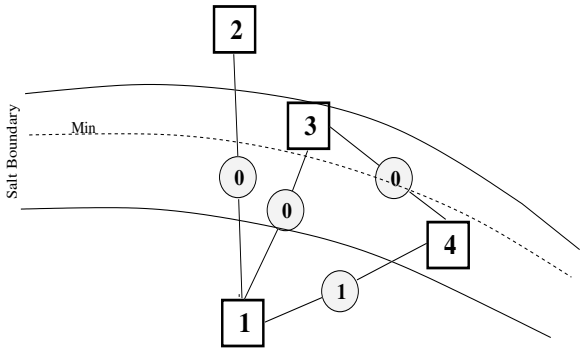

Figure 1: A cartoon illustrating some sample weights on the negative instantaneous amplitude. Several pixels are shown in squares and their corresponding weights in circles. The salt boundary is defined as areas where the negative instantaneous amplitude is below a threshold. jesse1-pixel [NR]

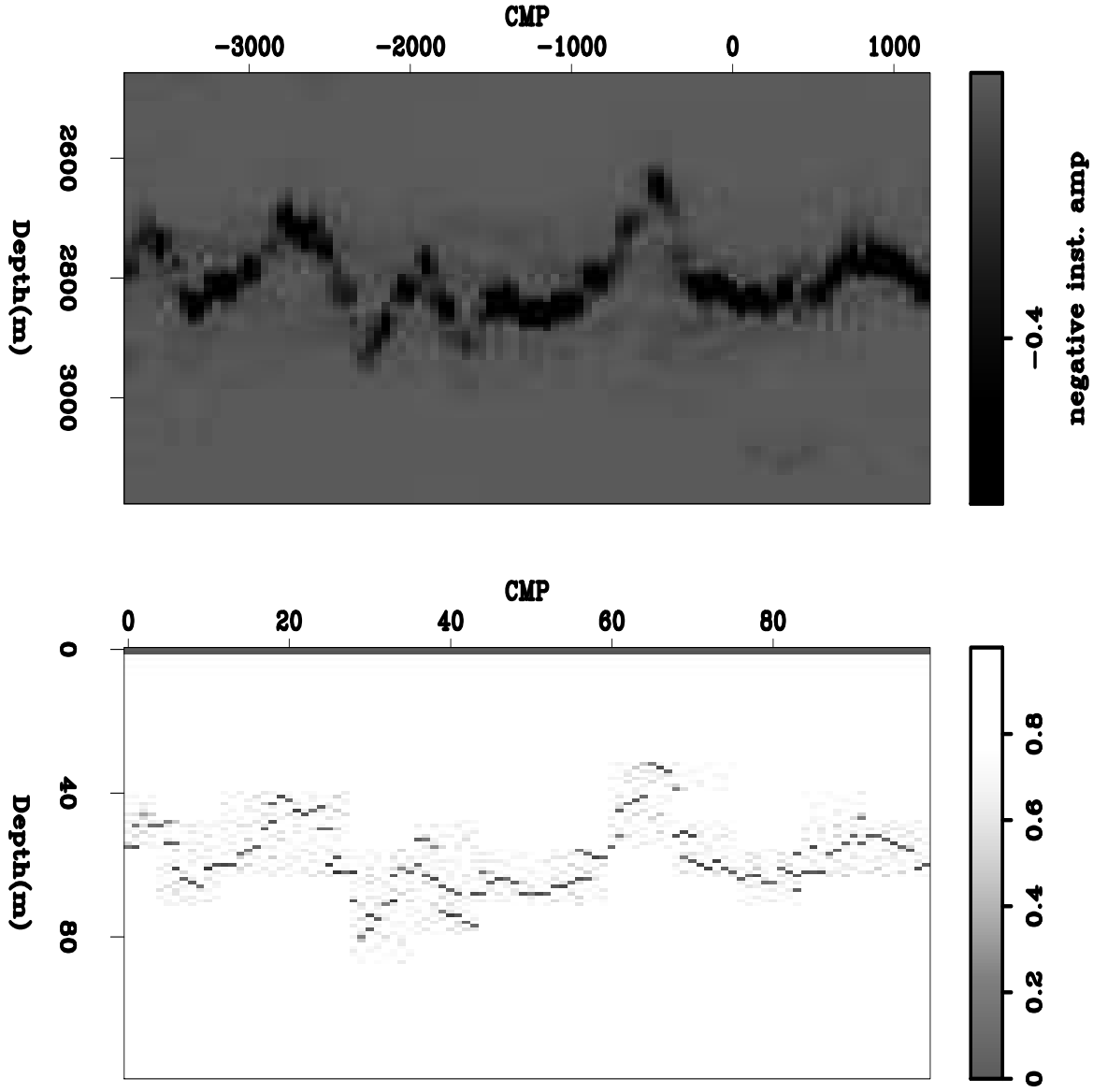

Figure 2: Top is negative absolute value of the complex trace of a synthetic salt boundary. Bottom is the weight connecting each pixel with the pixel directly above it. It is zero where two pixels are separated by the salt boundary. jesse1-amp.wt [ER]

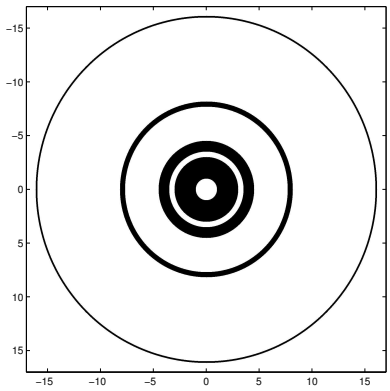

Figure 3: A cartoon of the $\log_2$ sampling of one pixel up to 16 samples. The black areas are sampled. jesse1-sampling [NR]

between each pixel and the adjoining pixel directly above it. Notice that it follows the peak of the amplitude.

We found that the best results are obtained when the maximum search distance is greater than 30 pixels. The maximum search distance is the maximum distance between pixels, beyond which the weights are set to zero. Using large search distances causes two problems. First, the matrix becomes more dense and requires more storage space. Second, the weights of the pixels at greater distances far outnumber those at close distances causing a bias in the weight matrix. Shi and Malik (2000) used a decaying distance weight to reduce this bias. We found that if we only use windows that are centered at powers of 2, then the matrix is still sufficiently sparse while still benefiting from greater search distances. However, this obviously causes many distances to be not sampled at all. Figure 3 shows the $\log_2$ sampling of one pixel up to a distance of 16 samples. To correct this, we plan to randomly sample from within the maximum search distance so that the number of non-zero points is approximately the same for all distances.

Iterative Velocity Analysis

This method can be used to pick salt boundaries that are discontinuous. An updated velocity model can then be created and used to remigrate the data.

Additionally, if the salt boundary changes only slightly, then the eigenvector solution to equation (2) from the previous segmentation can be used as an initial solution to the current segmentation.

The top panel of Figure 4 is a synthetic 2D section that has been migrated with a preliminary velocity model. The results of applying the segmentation method can be seen in the bottom. In general it does a good job picking the salt boundary, however at cmp locations `-750` and `-1950` it has some difficulty.

The top panel of Figure 5 contains the eigenvector with the second smallest eigenvalue from equation (2) that was partitioned to get the result in Figure 4. Below is a contour plot of this eigenvector. Notice that the spreading contours correspond to areas where the picking had some difficulty.

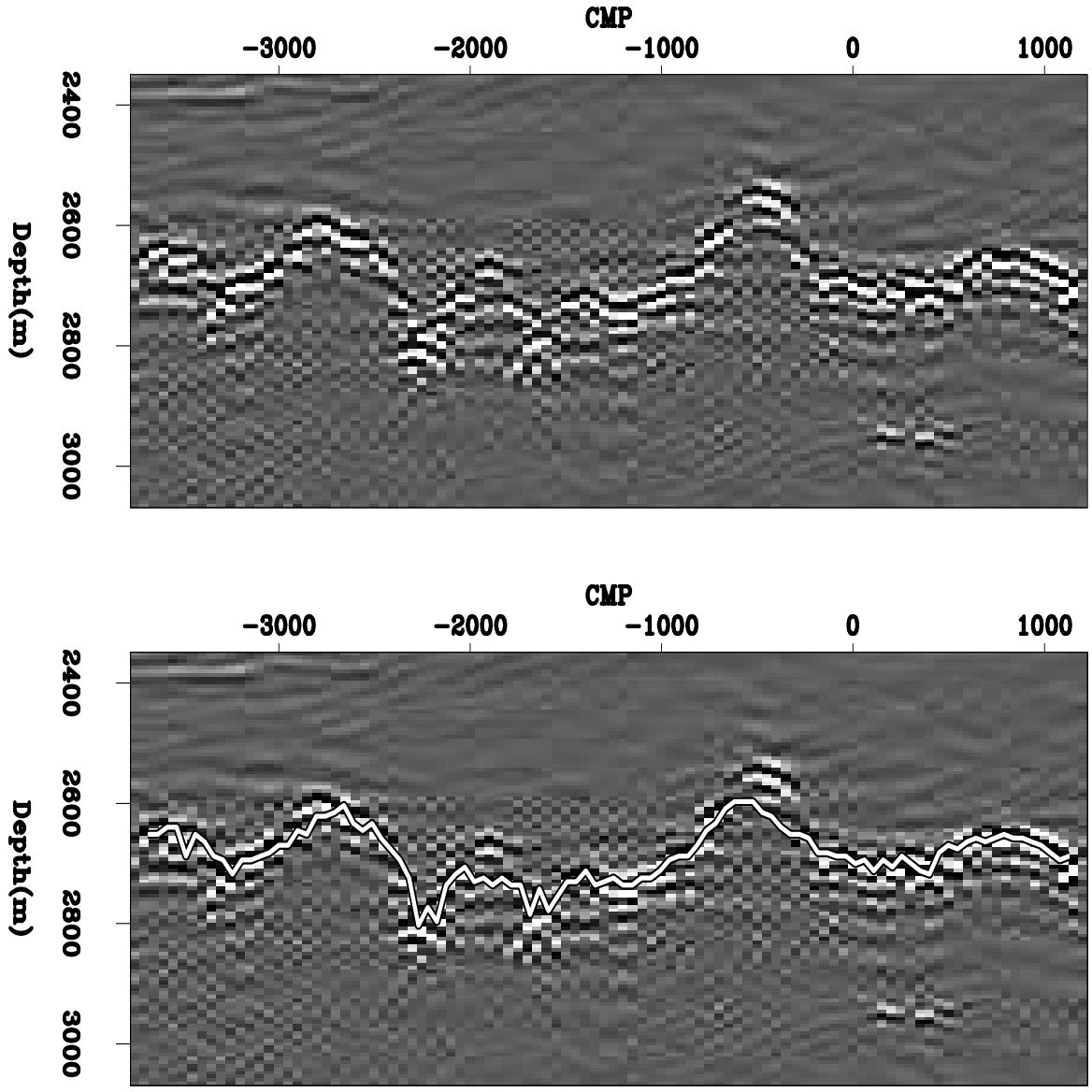

Figure 4: Segmentation can be an effective picker even in the presence of noise and discontinuities. Top is a synthetic salt boundary. Bottom is the resulting partition from the segmentation method. jesse1-p1.horizon [ER]

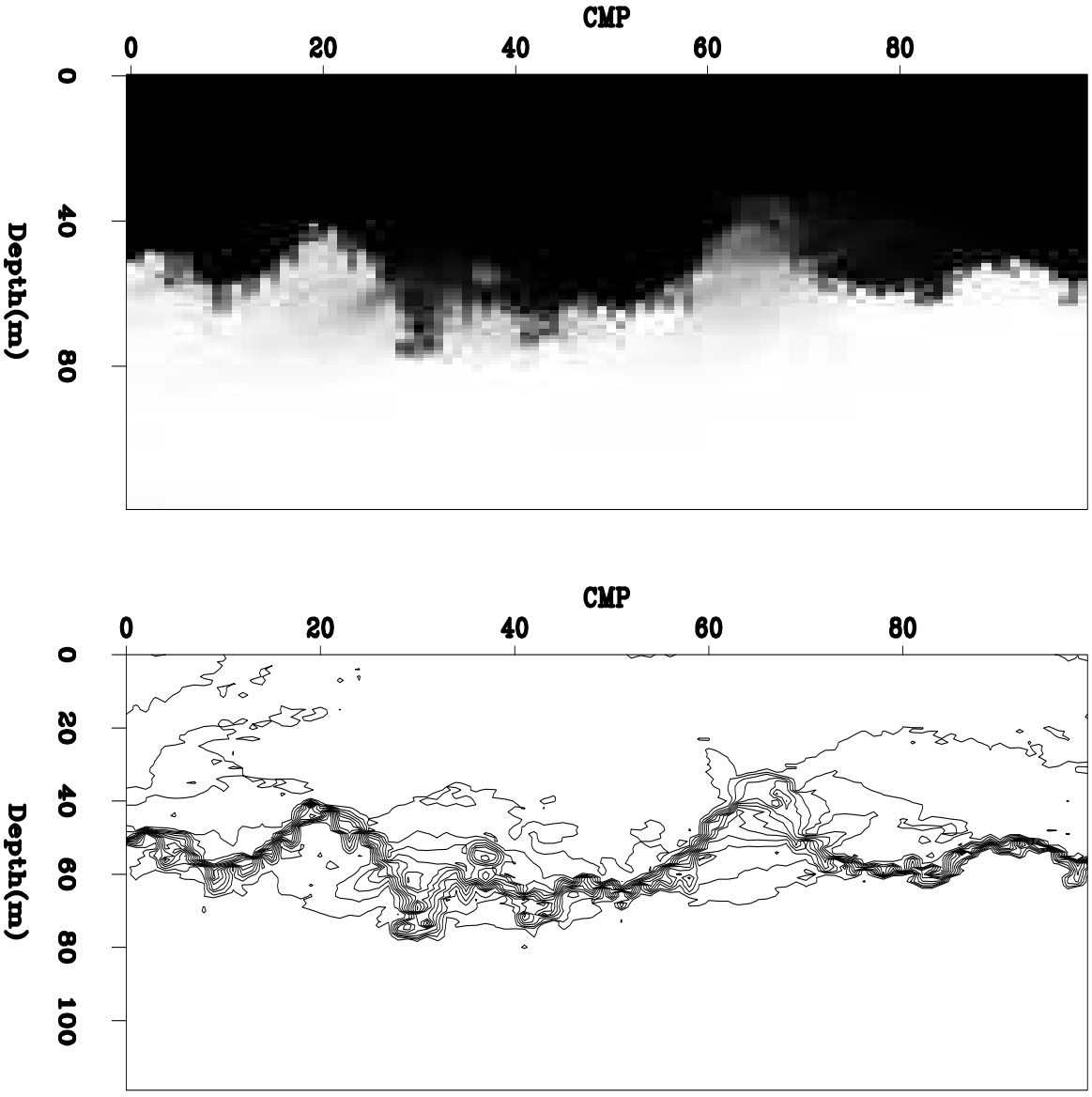

Figure 5: Areas of uncertain picking can be found by inspection of the eigenvector to be partitioned. Top is the eigenvector used to find the boundary in the lower part of Figure 4. Bottom is a contour plot of the eigenvector. Notice the areas of uncertainty where the contours are spreading. jesse1-p1.eig [ER]

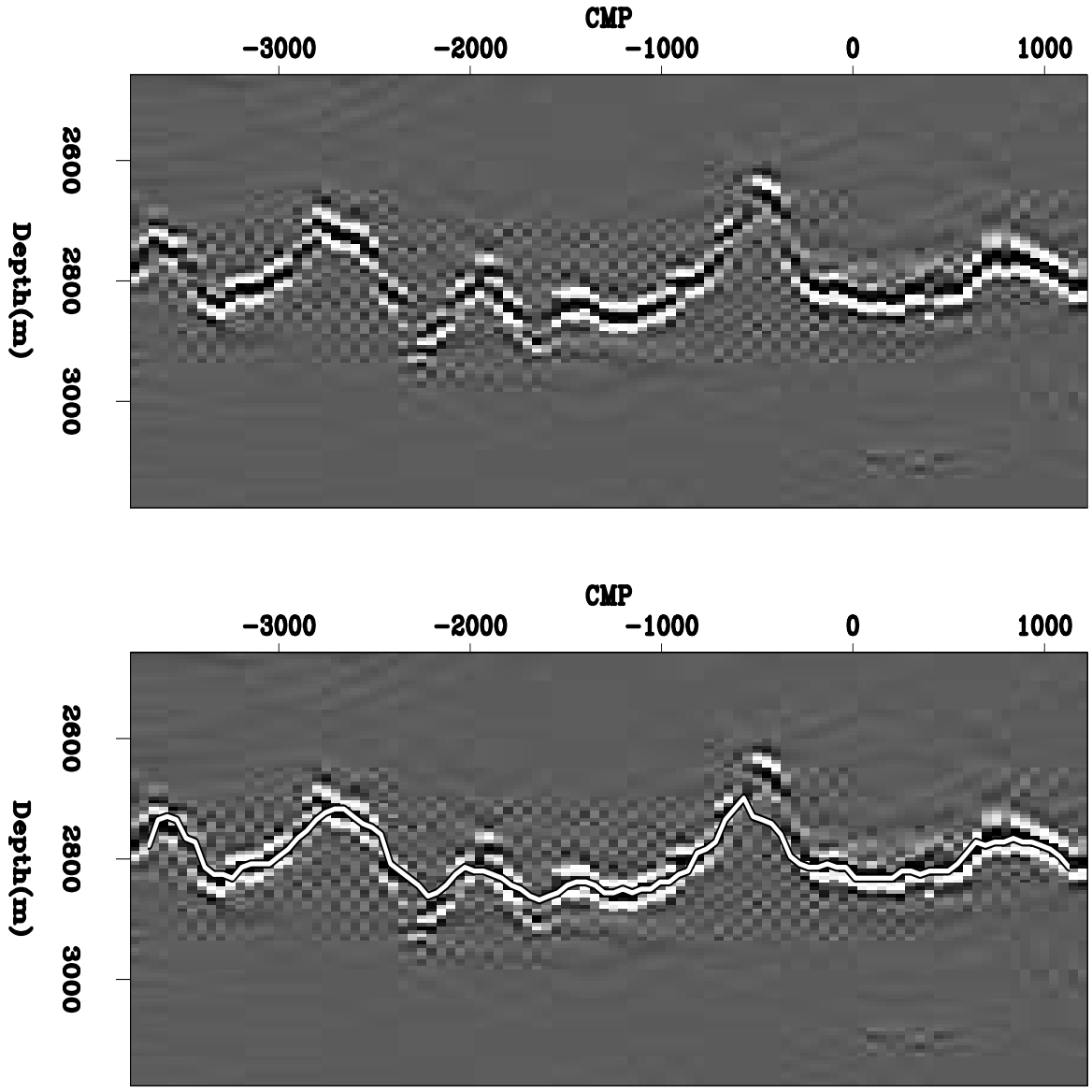

Figure 6: Top is a synthetic salt boundary after the velocity has been adjusted and remigrated. Bottom is the resulting partition from the segmentation method. jesse1-p2.horizon [ER]

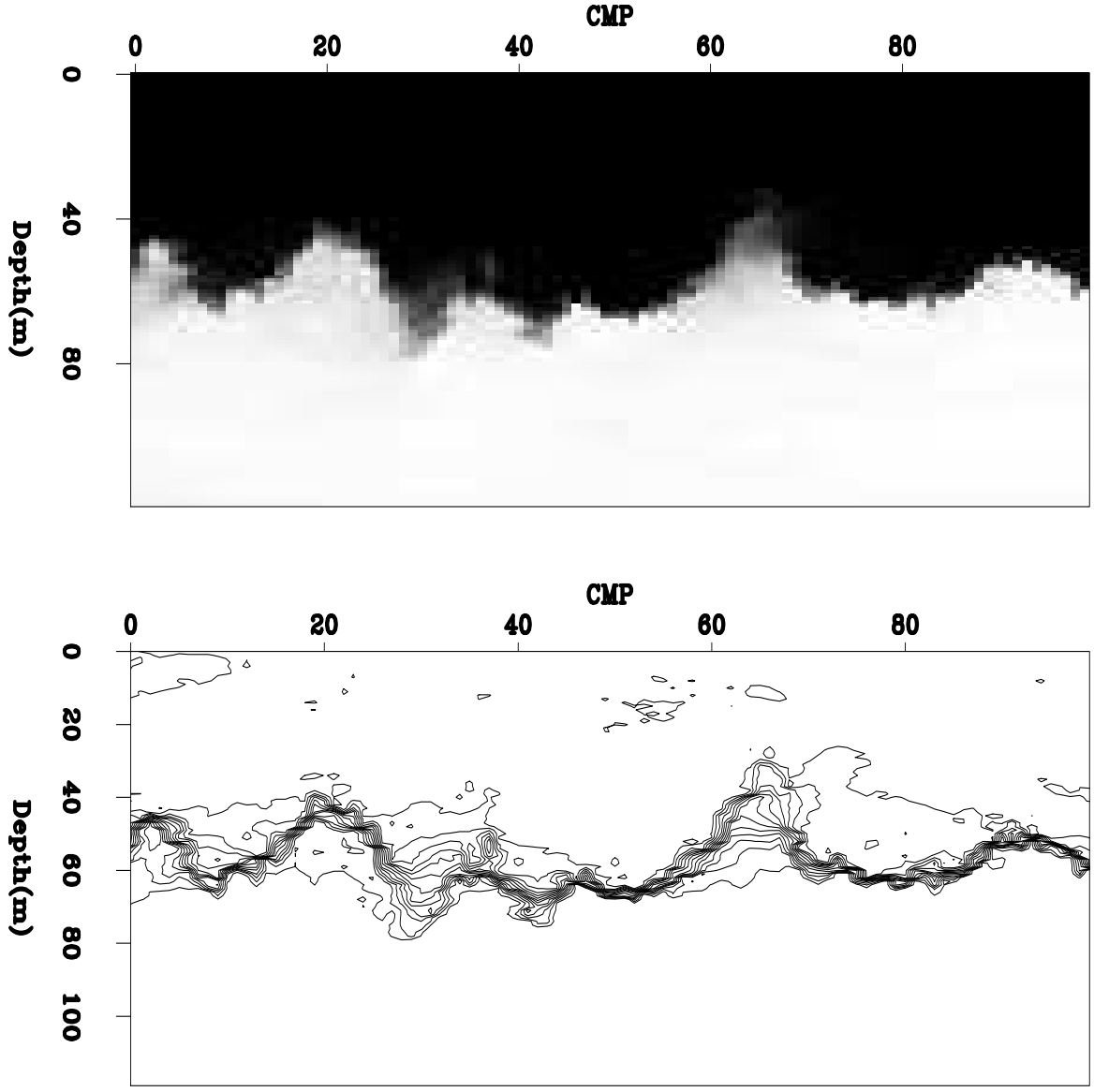

Figure 7: Top is the eigenvector used to find the boundary in the lower part of Figure 6. Bottom is a contour plot of the eigenvector. Notice the areas of uncertainty where the contours are spreading. jesse1-p2.eig [ER]

The top of Figure 6 is the same synthetic data used in Figure 4 except migrated with an updated velocity. Again, notice that the partitioning result in the bottom successfully picks the salt boundary except in a couple of places. The corresponding eigenvector is presented in Figure 7. Notice its similarity to the eigenvector from Figure 5. Initializing subsequent segmentation processes with previous eigenvectors should speed convergence.

CONCLUSIONS AND FUTURE WORK

Our modified segmentation method successfully tracked the salt boundaries in our test cases. These test cases present challenges including non-continuous salt boundaries with noise. An interpreter would still have to slightly adjust the tracking result in a few places.

Because of this method's excessive storage requirements, a multi-scale approach will be necessary. The size of the weight matrix in equation (2) is $(\mathtt{m} \times \mathtt{n})^2$ where $\mathtt{m}$ and $\mathtt{n}$ are the dimensions of the image. Even using sparse matrices in 2 dimensions, this matrix can be prohibitively large. In 3D, things get much worse as the matrix is now $(\mathtt{m} \times \mathtt{n} \times o)^2$, where $\mathtt{o}$ is the 3rd dimension. The segmentation method applied to a coarsely sub-sampled input cube will still select the salt boundary as long as it is the brightest amplitude in the cube. Smaller, more finely sampled cubes can then be segmented along the boundary.

Application of this method to 3D datasets is going to be a computational challenge that will require us to take advantage of maximizing the sparseness of the weight matrix, using accurate starting solutions, and taking a multi-scale approach.

ACKNOWLEDGMENTS

We would like to thank Frederick Billette and BP for making available to SEP the synthetic data used for our tests. We would also like to thank Antoine Guitton and Paul Sava for suggestions and data.

REFERENCES

Hale, D., and Emanuel, J. U., 2002, Atomic meshing of seismic images: Soc. of Expl. Geophys., Expanded Abstracts, 2126–2129.

Hale, D., and Emanuel, J. U., 2003, Seismic interpretation using global image segmentation: Soc. of Expl. Geophys., Expanded Abstracts, 2410–2413.

Lomask, J., 2003, Flattening 3-D data cubes in complex geology: SEP–**113**, 247–260.

Shi, J., and Malik, J., 2000, Normalized cuts and image segmentation: IEEE Trans on Pattern Analysis and Machine Intelligence, **22**, no. 8, 838–905.