Stanford Exploration Project, Report 120, May 3, 2005, pages 137–159

Update on flattening without picking

Jesse Lomask, Antoine Guitton, Sergey Fomel, and Jon Claerbout[1]

ABSTRACT

We present a method for efficiently flattening 3D seismic data volumes. First local dips are calculated over the entire seismic volume. The dips are then resolved into time shifts using a Gauss-Newton iterative approach that exploits the Fourier domain to maximize efficiency. To handle faults (discontinuous reflections), we apply a weight inversion scheme. This approach successfully flattens a synthetic faulted model, a field salt peircement dataset, a field dataset with an angular unconformity, and a faulted field dataset.

INTRODUCTION

In spite of numerous advances in computational power in recent years, interpretation still requires a lot of manual picking. One of the main goals of interpretation is to extract from the seismic data geological and reservoir features. One commonly used interpretation technique that helps with this effort is to flatten data on horizons [e.g. Lee (2001)]. This procedure removes structure and allows the interpreter to see geological features as they were emplaced. For instance, after flattening seismic data, an interpreter can see in one image an entire flood plain complete with meandering channels. However, in order to flatten seismic data, a horizon needs to be identified and tracked throughout the data volume. If the structure changes often with depth, then many horizons need to be identified and tracked. This picking process can be time consuming and expensive.

Certain interpretation visualization products and auto-pickers seek to make picking and flattening processes as efficient as possible. However, they often suffer from weaknesses that prevent them from being truly practical. For example, 3-D volume interpretation packages allow interpreters to view their data with depth perception using stereo glasses. These products have an opacity ability (James et al., 2002) that allows interpreters to make unwanted data transparent. Unfortunately, unless the zone of interest has a known unique range of attribute values, interpreters resort to picking on 2-D slices. Additionally, traditional amplitude-based auto-pickers can fail if the horizon being tracked has significant amplitude variation, or worse, polarity reversal. Other tracking techniques such as artificial neural networks are less sensitive to amplitude variations but are still prone to error if the seismic wavelet character varies significantly from the training data (Leggett et al., 1996).

[1]**email:** lomask@sep.stanford.edu, antoine@sep.stanford.edu, sergey.fomel@beg.utexas.edu, jon@sep.stanford.edu

In this document, we propose a method for automatically flattening entire 3-D seismic cubes without picking that we first presented in Lomask (2003a). This is essentially an algorithm that is efficient enough to perform dense picking on entire 3D cubes at once. Our method involves first calculating dips everywhere in the data using a dip estimation technique (Claerbout, 1992; Fomel, 2002). These dips are resolved into time shifts via a non-linear least-squares problem. The data are subsequently shifted according to the time shifts to output a flattened volume. Bienati et al. (1999a,b); Bienati and Spagnolini (2001, 1998) use a similar approach to resolve numerically the dips into time shifts for the purpose of auto-picking horizons and flattening gathers, yet solving a linear version of the problem and without flattening the full volume at once. Stark (2004) takes a full volume approach to achieve the same end yet is unwrapping instantaneous phase. Blinov and Petrou (2003) use dynamic programming to track horizons by summing local dips. Here, a version of the non-linear problem of summing local dips (Lomask, 2003b; Guitton et al., 2005) is solved iteratively using a Gauss-Newton approach. Each iteration utilizes the Fourier domain to invert efficiently a linearized operator much like the approach of Ghiglia and Romero (1994) for unwrapping two-dimensional phase. For faulted data, weights identifying the faults are applied within the iterative scheme, allowing reconstruction of horizons for certain fault geometries. As with amplitude based auto-pickers, amplitude variation also affects the quality of the dip estimation and will, in turn, impact the quality of this flattening method. However, the effect will be less significant because this method can flatten the entire data cube at once, globally, in a least-squares sense, minimizing the effect of questionable dip information. Additionally, flattening the entire cube at once should make the method more robust in noisy data or complicated structures. Once a seismic volume is flattened, automatic horizon tracking becomes a trivial matter of reversing the flattening process to unflatten flat surfaces. The prestack applications for this method are numerous and can be easily incorporated into an automatic velocity picking scheme (Guitton et al., 2004).

In the following sections, we present an overview of the flattening methodology and a series of real world geological challenges for this method in order of increasing difficulty. The first is a 3D synthetic that is flattened to demonstrate how we can flatten faulted, folded data. Then we present a structurally simple salt peircement 3D field data from the Gulf of Mexico. We consider it structurally simple because the dip does not change much with depth. Increasing complexity, we flatten 3D field data from the North Sea that contains an unconformity and has significant folding. Lastly, we illustrate how this method is used to flatten datasets with faults on a field 3D Gulf of Mexico dataset.

FLATTENING THEORY

The basic idea is similar to phase unwrapping (Claerbout, 1999), but instead of summing phase differences to get total phase, dips are summed to get total time shifts that are then used to flatten the data. To apply the shifts, a reference is held constant and all other traces are shifted vertically to match it.

The first step is to calculate dips everywhere in the 3-D seismic cube. Dip can be efficiently

calculated using a plane-wave destructor as described in Claerbout (1992) or with an improved dip estimator that is described in Fomel (2002). We primarily use the latter. For each point in the data cube, two components of dip, p_x and p_y, are estimated in the x direction and y direction, respectively. These can be represented everywhere on the mesh as vectors $\mathbf{p}_x$ and $\mathbf{p}_y$.

Our goal is to find a time-shift (or depth-shift) field $\boldsymbol{\tau}(x,y,z)$ such that its gradient approximates the dip $\mathbf{p}(\mathrm{x},\mathrm{y},\boldsymbol{\tau})$. Our objective function is:

$$J(\boldsymbol{\tau}) = \int\int \left[\left(\mathbf{p}_x(\boldsymbol{\tau},x,y) - \frac{\partial \boldsymbol{\tau}}{\partial x} \right)^2 + \left(\mathbf{p}_y(\boldsymbol{\tau},x,y) - \frac{\partial \boldsymbol{\tau}}{\partial y} \right)^2 \right] dx\, dy \tag{1}$$

The Euler-Lagrange equation is used in calculus of variations (Farlow, 1993) to find a function $(\boldsymbol{\tau})$ that will minimize a functional (J).

We apply the Euler-Lagrange equation to equation (1) to find:

$$\frac{\partial^2 \boldsymbol{\tau}}{\partial x^2} + \frac{\partial^2 \boldsymbol{\tau}}{\partial y^2} = \frac{\partial \mathbf{p}_x}{\partial x} + \frac{\partial \mathbf{p}_y}{\partial y} + \frac{1}{2} \frac{\partial (\mathbf{p}_x{}^2 + \mathbf{p}_y{}^2)}{\partial \boldsymbol{\tau}}. \tag{2}$$

In our method, we ignore the last term of equation (2) and iteratively solve:

$$\frac{\partial^2 \boldsymbol{\tau}}{\partial x^2} + \frac{\partial^2 \boldsymbol{\tau}}{\partial y^2} = \frac{\partial \mathbf{p}_x}{\partial x} + \frac{\partial \mathbf{p}_y}{\partial y}. \tag{3}$$

This equation can be rewritten using the gradient ($\boldsymbol{\nabla} = [\frac{\partial}{\partial x} \;\; \frac{\partial}{\partial y}]'$) and the estimated dip ($\mathbf{p} = [\mathbf{p}_x \;\; \mathbf{p}_y]'$)

$$\boldsymbol{\nabla}' \boldsymbol{\nabla} \boldsymbol{\tau} \quad = \quad \boldsymbol{\nabla}' \mathbf{p}. \tag{4}$$

Ultimately, the regression to be solved is:

$$\boldsymbol{\nabla} \boldsymbol{\tau} \quad = \quad \mathbf{p}. \tag{5}$$

This equation means that we need to find a time-shift field $\boldsymbol{\tau}(x,y,z)$ such that its gradient approximates the dip $\mathbf{p}(\mathrm{x},\mathrm{y},\boldsymbol{\tau})$. Note that the estimated dip $\mathbf{p}(\mathrm{x},\mathrm{y},\boldsymbol{\tau})$ field is a function of the unknown $\boldsymbol{\tau}(x,y,t)$ field, making this problem non-linear and, therefore, difficult to solve directly.

We solve this using a Gauss-Newton approach by iterating over equations (6)-(8)

iterate {

$$\mathbf{r} \quad = \quad [\boldsymbol{\nabla} \boldsymbol{\tau}_k - \mathbf{p}] \tag{6}$$

$$\Delta \boldsymbol{\tau} \quad = \quad (\boldsymbol{\nabla}' \boldsymbol{\nabla})^{-1} \boldsymbol{\nabla}' \mathbf{r} \tag{7}$$

$$\boldsymbol{\tau}_{k+1} \quad = \quad \boldsymbol{\tau}_k + \Delta \boldsymbol{\tau} \tag{8}$$

}

To dramatically improve efficiency, we solve equation (7) in the Fourier domain. We apply the divergence to the estimated dips and divide by the z-transform of the Laplacian in the Fourier domain with:

$$\Delta\boldsymbol{\tau} \quad \approx \quad \mathrm{FFT_{2D}}^{-1}\left[\frac{\mathrm{FFT_{2D}}\left[\nabla'\mathbf{r}\right]}{-Z_x^{-1}-Z_y^{-1}+4-Z_x-Z_y}\right] \tag{9}$$

where $Z_x = e^{iw\Delta x}$ and $Z_y = e^{iw\Delta y}$. This amounts to calling both a forward and inverse FFT in each iteration. The ability to invert the 2D Laplacian in one step is the key to this method's exceptional performance.

Once we have converged the resulting time-shift field, $\boldsymbol{\tau}(x,y,t)$ contains all of the time-shifts required to map the original unflattened data to flattened data. This is implemented by applying the time-shifts relative to a reference trace. In other words, each trace is shifted to match the reference trace.

In general, we operate on a one-time slice at a time. After iterating until convergence, we then select the next time slice and proceed down through the cube. In this way, each time slice is solved independently.

The process of flattening tends to alter the spectrum of the data by stretching and compressing in time. Even worse, it can disrupt its continuity. To insure a monotonic and continuous result, it should be sufficient to first smooth the input dips in time (or depth). In some instances, it may be necessary to enforce smoothness during the integration of the dips. This can easily be accomplished by defining a 3D gradient operator with an adjustable smoothing parameter as:

$$\nabla_\epsilon\boldsymbol{\tau} \quad = \quad \begin{bmatrix} \frac{\partial\boldsymbol{\tau}}{\partial\mathbf{x}} \\ \frac{\partial\boldsymbol{\tau}}{\partial\mathbf{y}} \\ \epsilon\frac{\partial\boldsymbol{\tau}}{\partial\mathbf{z}} \end{bmatrix} \quad \approx \quad \begin{bmatrix} \mathbf{p}_x \\ \mathbf{p}_y \\ 0 \end{bmatrix} \quad = \quad \mathbf{p}. \tag{10}$$

Our new operator ∇_ϵ has a scalar parameter ϵ used to control the amount of vertical smoothing. This requires integrating the entire 3D volume of dips at once rather than slice by slice. The 2D FFT's in equation (9) are replaced with 3D FFT's. Consequently, each iteration is slowed.

Weighted solution for faults

Local dips estimated at fault discontinuities can be inaccurate. We can, however, sum dips around the faults and ignore the spurious dips across the faults to get a flattened result.

We can add a weight to the residual to ignore fitting equations that are affected by the bad dips estimated at the faults. The regression to be solved is now:

$$\mathbf{W}\nabla\boldsymbol{\tau} \quad = \mathbf{W}\mathbf{p}. \tag{11}$$

We iterate over the same equations as before, except equation (6) is now replaced with:

$$\mathbf{r} \quad = \quad \mathbf{W}[\nabla \boldsymbol{\tau}_{\mathbf{k}} - \mathbf{p}] \tag{12}$$

Because we cannot apply a non-stationary weight in the Fourier domain, we use the same Fourier method as shown in equation (9). This means that the Fourier method is not approximating the inverse as well as before. The cost of this is more iterations, but method is still relatively efficient compared to other iterative methods. This approach is similar to Ghiglia and Romero (1994) for phase unwrapping.

Computational cost

For a data cube with dimensions $n = n1 \times n2 \times n3$, each iteration requires n1 forward and reverse 2D FFT's. Therefore, number of operations per iteration is about $8n(1 + log(4n2 \times n3))$. The number of iterations is a function of the variability of the structure and the degree of weighting. For instance, if the structure is constant with depth, then it will be flattened in one iteration. On the other hand, if a weight is applied and the structure changes much with depth, it may take as many as 100 iterations.

EXAMPLES OF 3D FLATTENING

Here we demonstrate this flattening method's efficacy on synthetic and field 3D datasets. We start with a synthetic to illustrate how this method can handle faults with folds. Then we use several 3D field datasets to demonstrate how this method handles folds, angular unconformities, and faults.

Synthetic Data

Figure 1 is a 3D synthetic dataset that presents two geological challenges. Firstly, the structure is changing with depth, requiring multiple non-linear iterations. Secondly, a significant fault is present in the middle of the cube. Dips estimated at fault discontinuities are, in general, inaccurate and will compromise the quality of the flattening result. However, knowing the position of the fault, we pass it as a weight to equation (12). In this case, we used a binary weight of 0's at the fault (and a few samples on each side) and 1's everywhere else.

The flattening result is shown in Figure 2. Notice the cube is very flat except in the area of the fault itself. This method is able to flatten this cube because we passed it a weight identifying the fault and the fault is limited laterally. That is, the tip-line of the fault is contained within the data cube, enabling dips to be summed around the fault.

The estimated $\boldsymbol{\tau}$ field used to flatten data can also be used to reverse the process. That is, we can use the time-shifts to unflatten data that is already flat. By unflattening flat surfaces and overlaying them on the data, we are essentially picking any or all of the horizons in the data-cube. Figure 3 displays every 20th horizon overlain on the synthetic data shown in Figure 1.

We could just as easily have displayed every horizon but the image would appear too cluttered. Notice that the horizons are well tracked even across a fault.

Chevron Gulf of Mexico Salt Peircement Data

Figure 4 is a field 3D data cube from the Gulf of Mexico provided by Chevron. It consists of structurally simple horizons that have been warped up around a salt peircement. Numerous channels can be seen in time slices. In the time slice at the top of Figure 4, a channel can be seen snaking across the south side. Figure 5 shows the flattened output of the data in Figure 4. The seismic cube has been converted from a stack of time slices to a stack of horizon slices. Notice that the low frequency banding present in the unflattened data is no longer present in the flattened data. This is because horizons are no longer cutting across the image. Several channels are now easily viewed on the horizon slice. Also, notice the salt dome appears to be more localized in Figure 5.

Figure 5 displays three horizons overlain on the original data in Figure 4. The horizons track the reflectors on the flanks of the salt well as expected. Within the salt, the horizons gradually get off their respective reflector events as the estimated dip becomes very inaccurate. The time slice at the top displays the swath of a tracked horizon.

Elf North Sea Unconformity Data

Figure 7 shows a 3D Elf North Sea data set. Marked by considerable folding and a sharp angular unconformity, this data presents a formidable flattening challenge.

The flattening result is shown in Figure 8. In order to preserve the continuity of the data, we used a smoothing parameter ϵ of 1.0 in equation (10). Thus, it is no surprise that the result is not completely flat. Had we used an ϵ of 0.0, the data would be flatter but would lose its continuity. Consequently, the trade-off between continuity and flatness emerges in cases of pinch-outs and unconformities. Notice that at the bottom of the section, there are some discontinuities introduced by the flattening process.

When it comes to overlaying the picks on the data, an ϵ of 0.0 is preferable. The results are shown in Figure 9. The time slice at the top shows the swaths of two different horizons. Overall, the picked horizons track up to and along the unconformity. The only significant errors occur where the data quality is questionable and, as a result, the estimated dip is inaccurate.

Chevron Gulf of Mexico Faulted Data

Figure 10 is an image of faulted Gulf of Mexico data. When a significant discontinuity occurs within a data-set, local dip estimators return incorrect estimates. Faults with significant displacement contaminate the dip estimation at the faults. This, in turn, results in unsatisfactory flattening results.

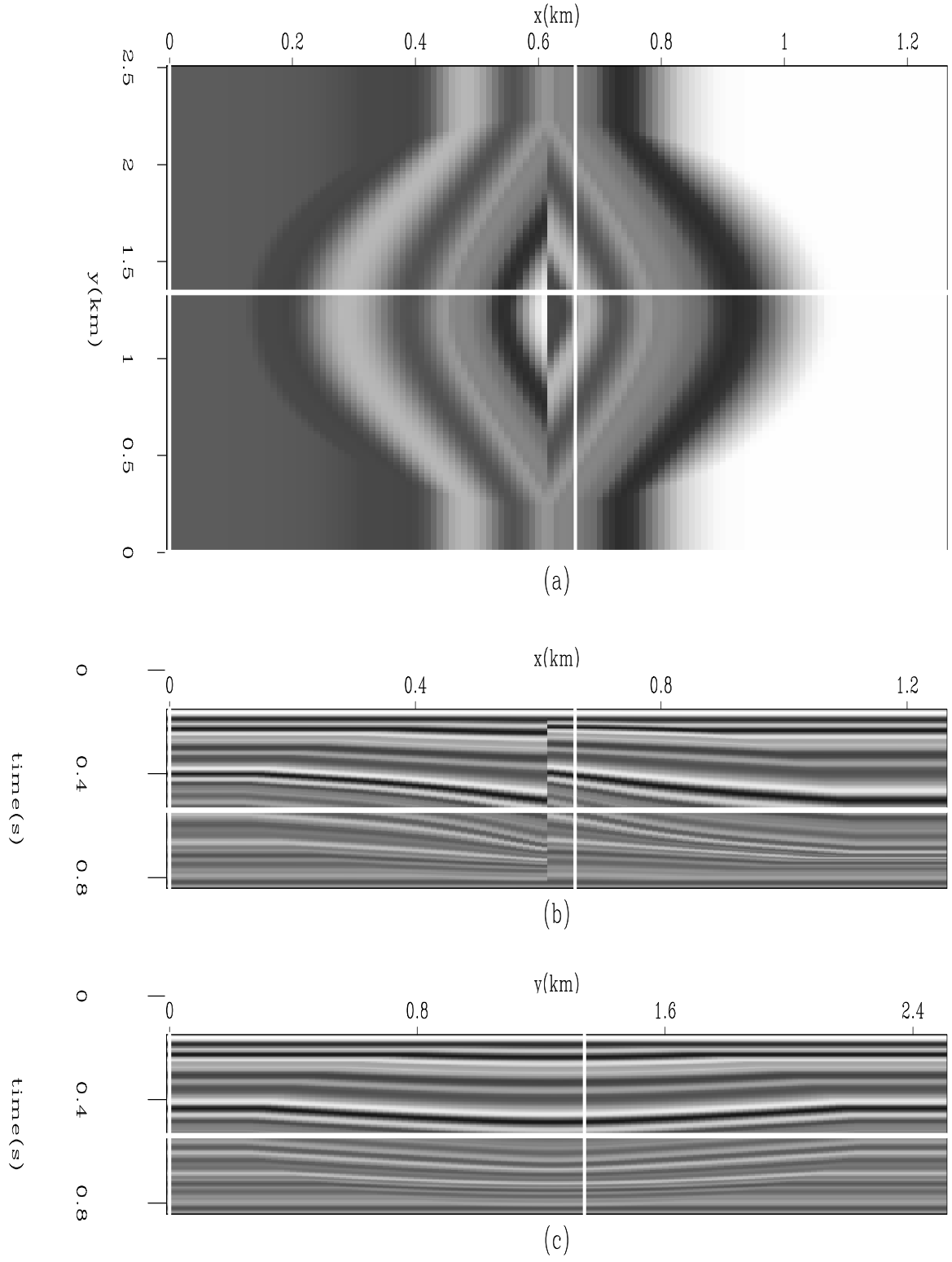

Figure 1: A synthetic model with structure varying with depth that necessitates multiple iterations as well as a vertical discontinuity representing a fault. (a) The time slice at time=.54 *s*. (b) An in-line section at y=1.34 *km*. (c) A cross-line section at x=.66 *km*. jesse1-down_lap [ER]

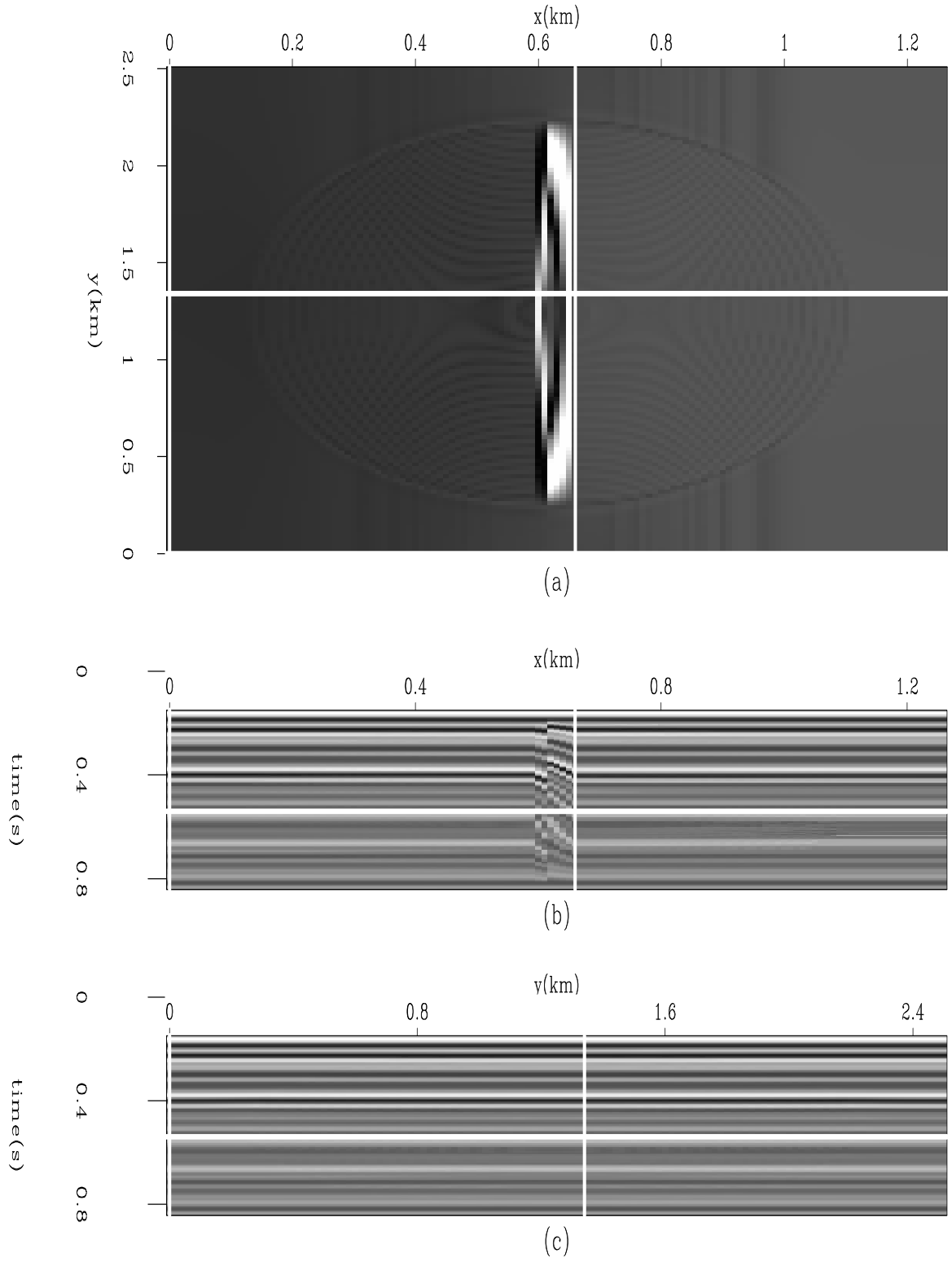

Figure 2: Result of flattening of Figure 1. Notice it is flat on both sides of the fault zone. (a) The horizon slice at time=.54 *s*. (b) An in-line section at y=1.34 *km*. (c) A cross-line section at x=.66 *km*. jesse1-down_lap_flat [ER]

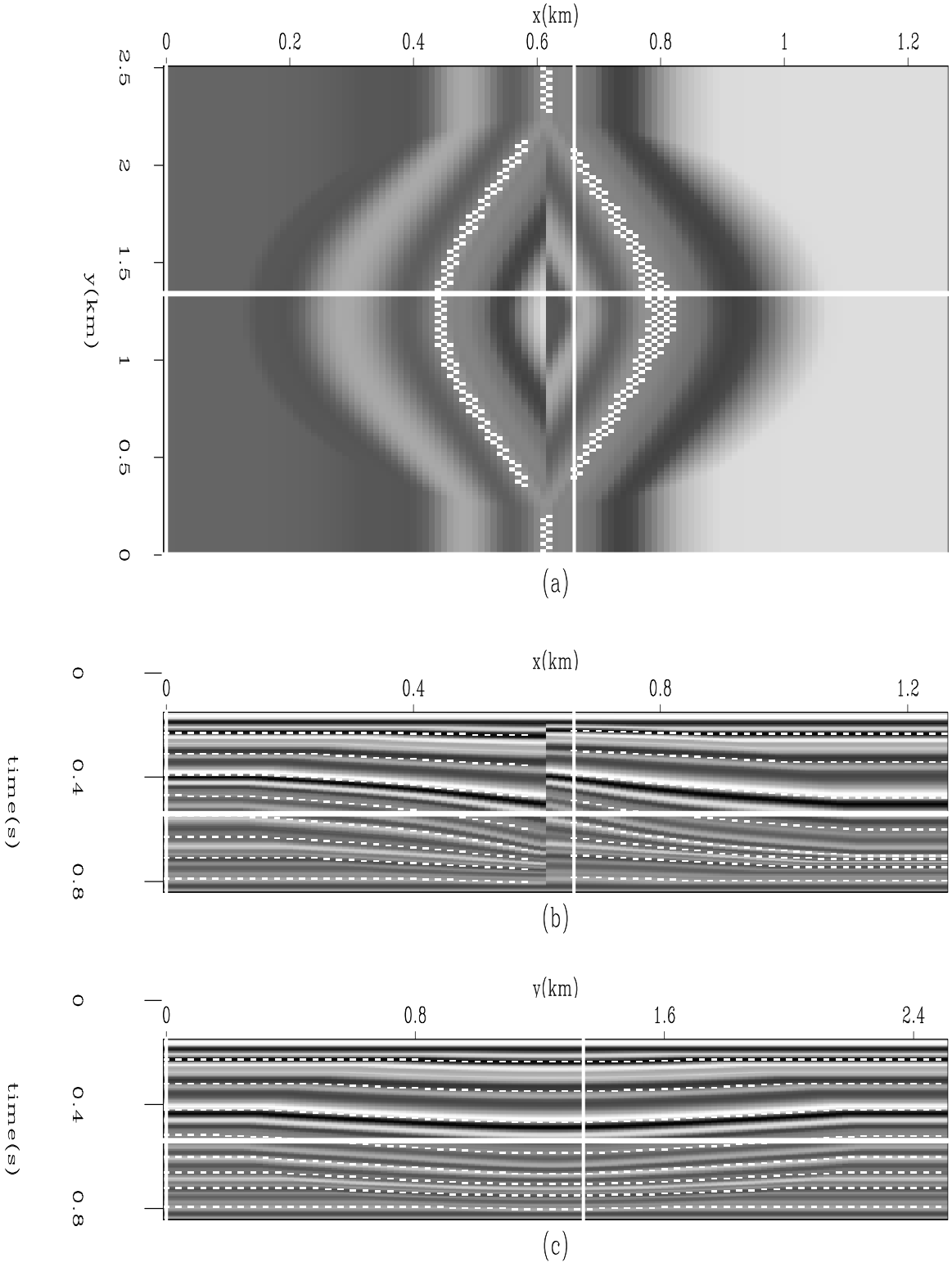

Figure 3: Result of overlaying tracked horizons on the image in Figure 1. The gap between reflections is where the weight was applied. It successfully tracks the horizons. (a) The time slice at time=.54 *s*. (b) An in-line section at y=1.34 *km*. (c) A cross-line section at x=.66 *km*. jesse1-down_lap.horizon_overlay [ER]

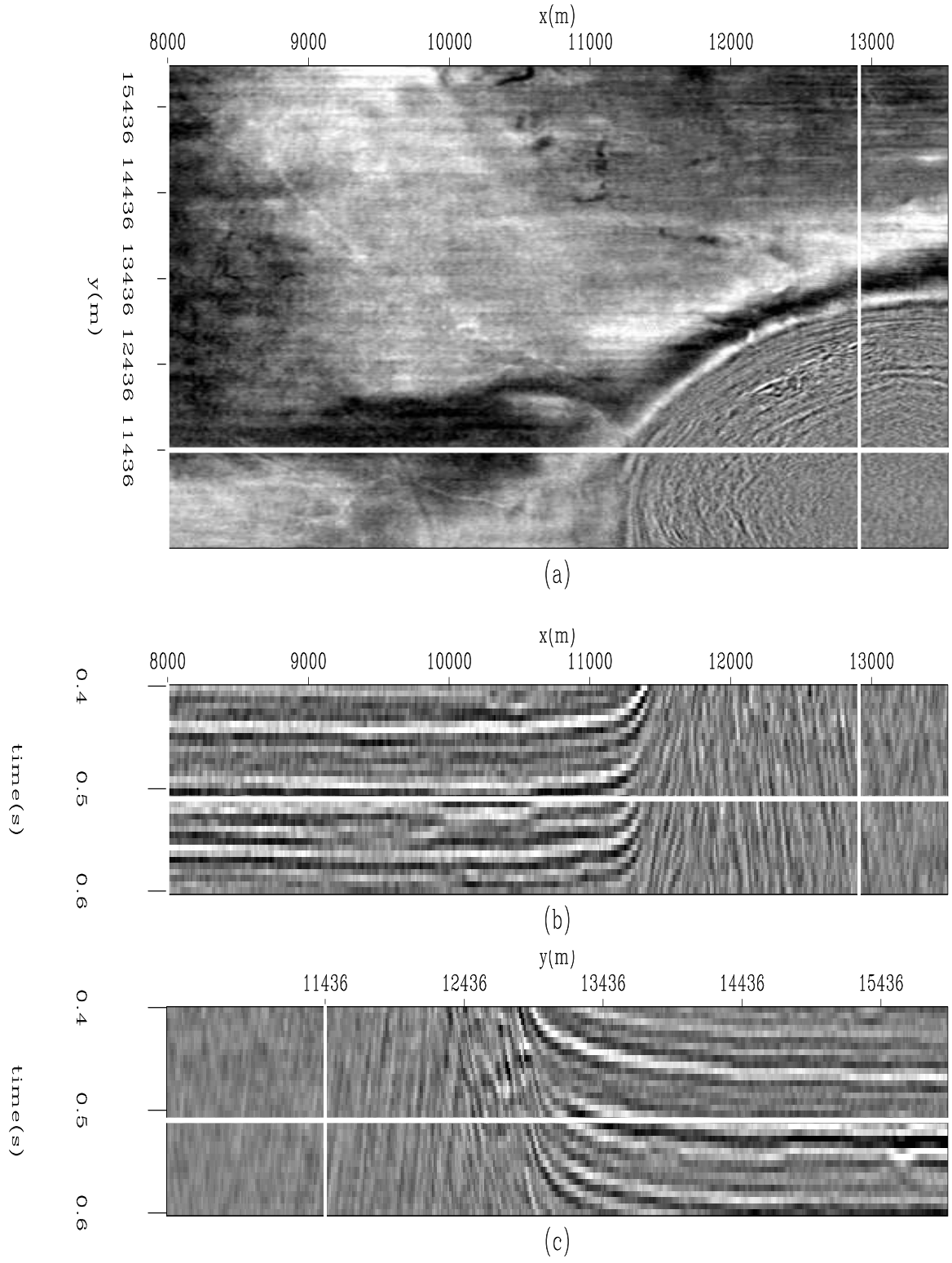

Figure 4: Chevron Gulf of Mexico data. Notice the beds have been forced up to steep angles by the emplacement of a salt peircement. There is some evidence of channels in the time slice. (a) The time slice at time=.51 s. (b) An in-line section at y=11436.1 m. (c) A cross-line section at x=12911 m. jesse1-chev [ER]

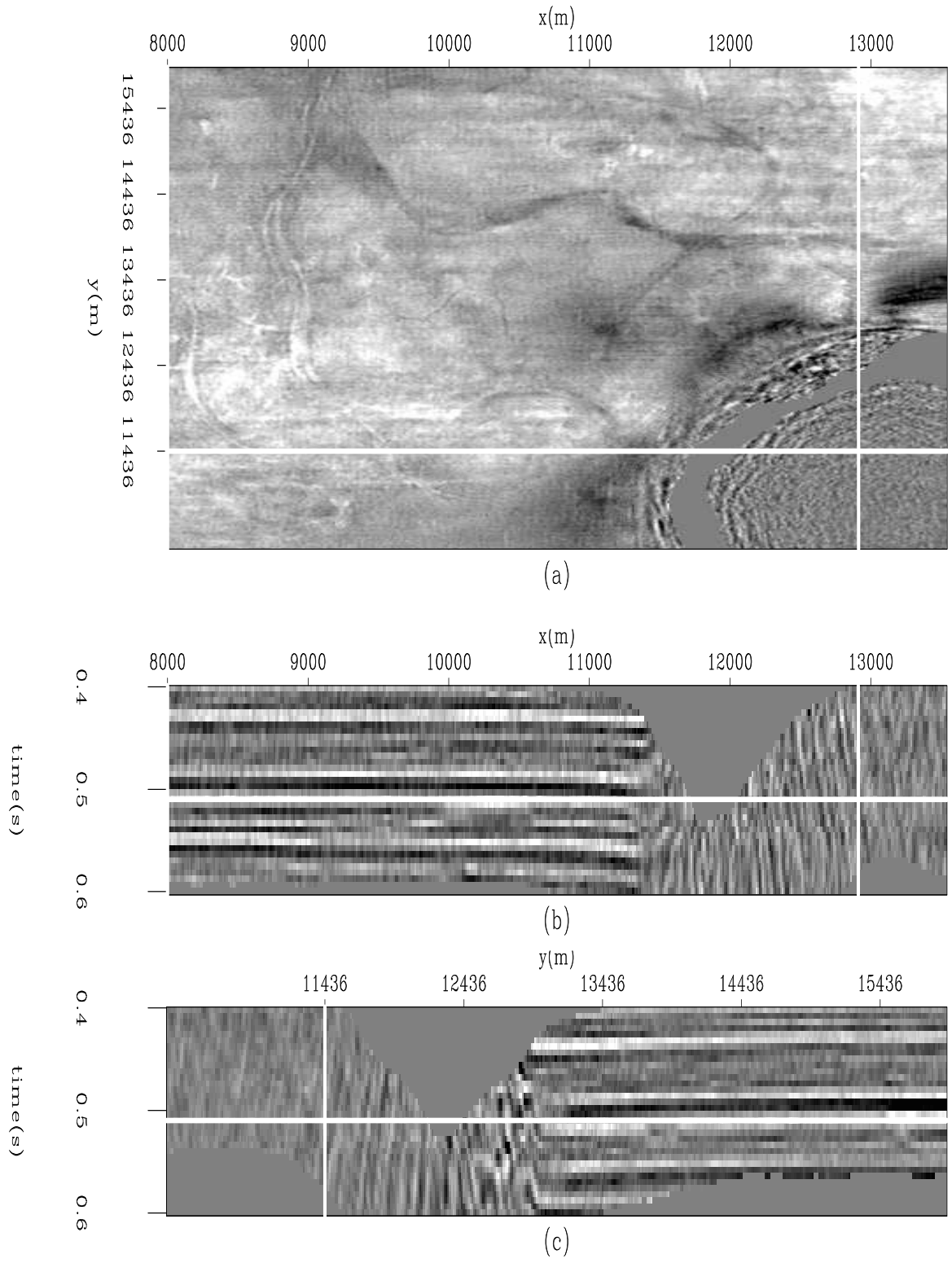

Figure 5: Result of flattening of Figure 4. The top panel is now a horizon slice displaying several channels. The vertical sections illustrate that the cube is flat in both the x and y directions. (a) The horizon slice at time=.51 s. (b) An in-line section at y=11436.1 m. (c) A cross-line section at x=12911 m. jesse1-chev.3Dflat [ER]

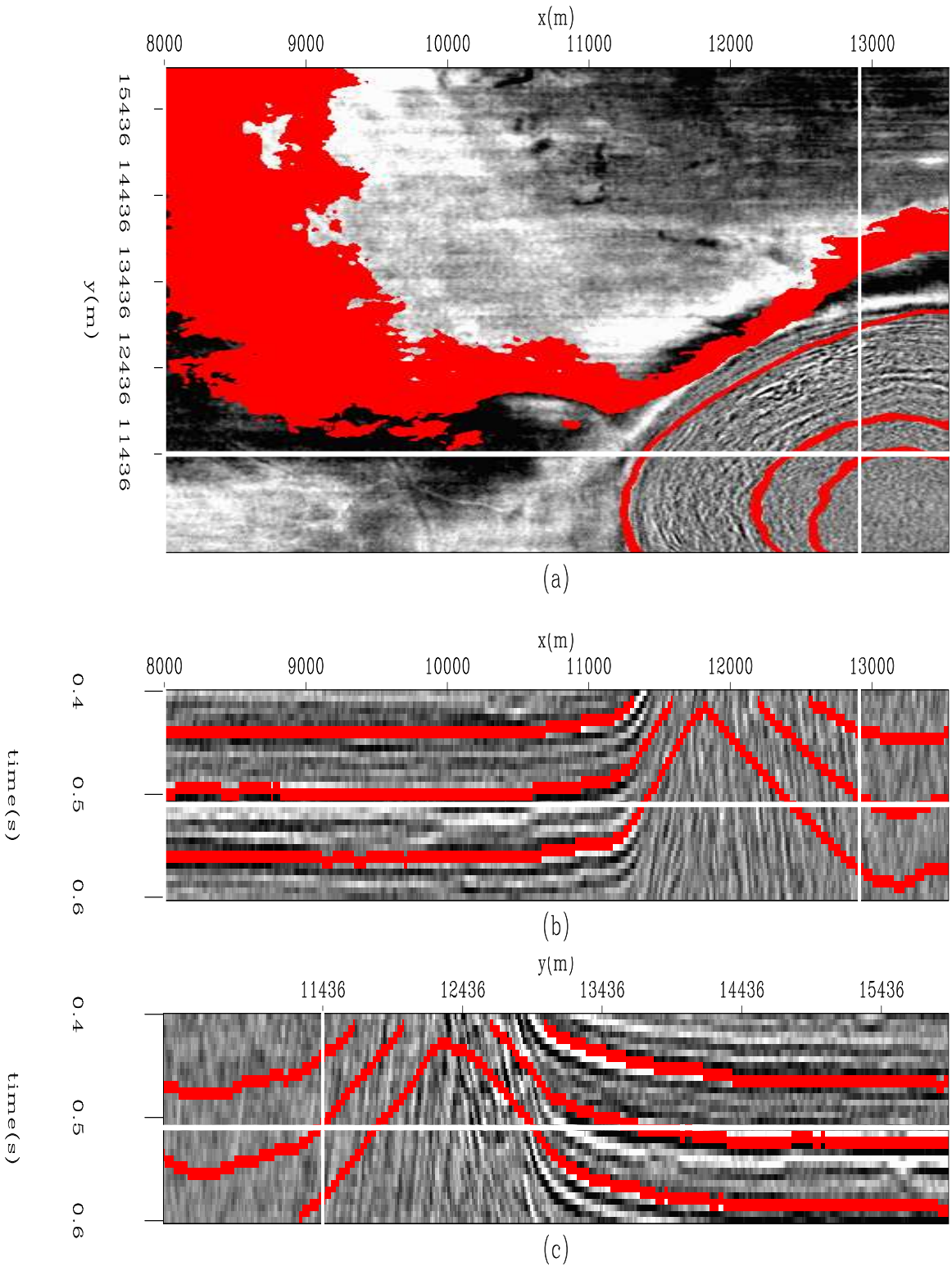

Figure 6: Result of overlaying tracked horizons on the image in Figure 4. It successfully tracks the horizons even to the considerably steep dips leading into the salt peircement. (a) The time slice at time=.51 *s*. (b) An in-line section at y=11436.1 *m*. (c) A cross-line section at x=12911 *m*. jesse1-chev.horizon_overlay [ER]

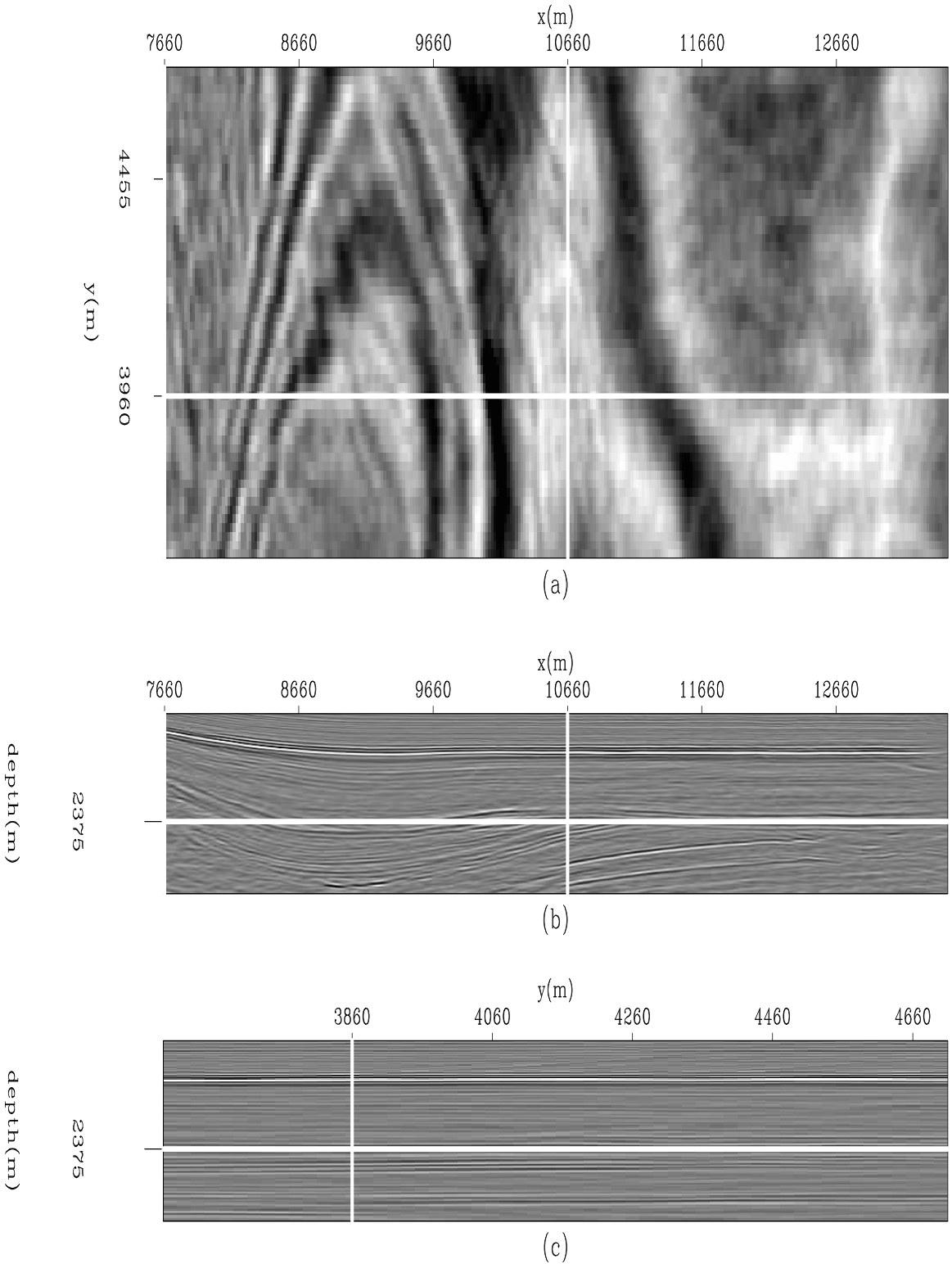

Figure 7: Elf North Sea data. Observe the angular unconformity at 2425 meters. (a) The depth slice at depth=2375 *m*. (b) An in-line section at y=3960 *m*. (c) A cross-line section at x=10660 *m*. jesse1-elf [ER]

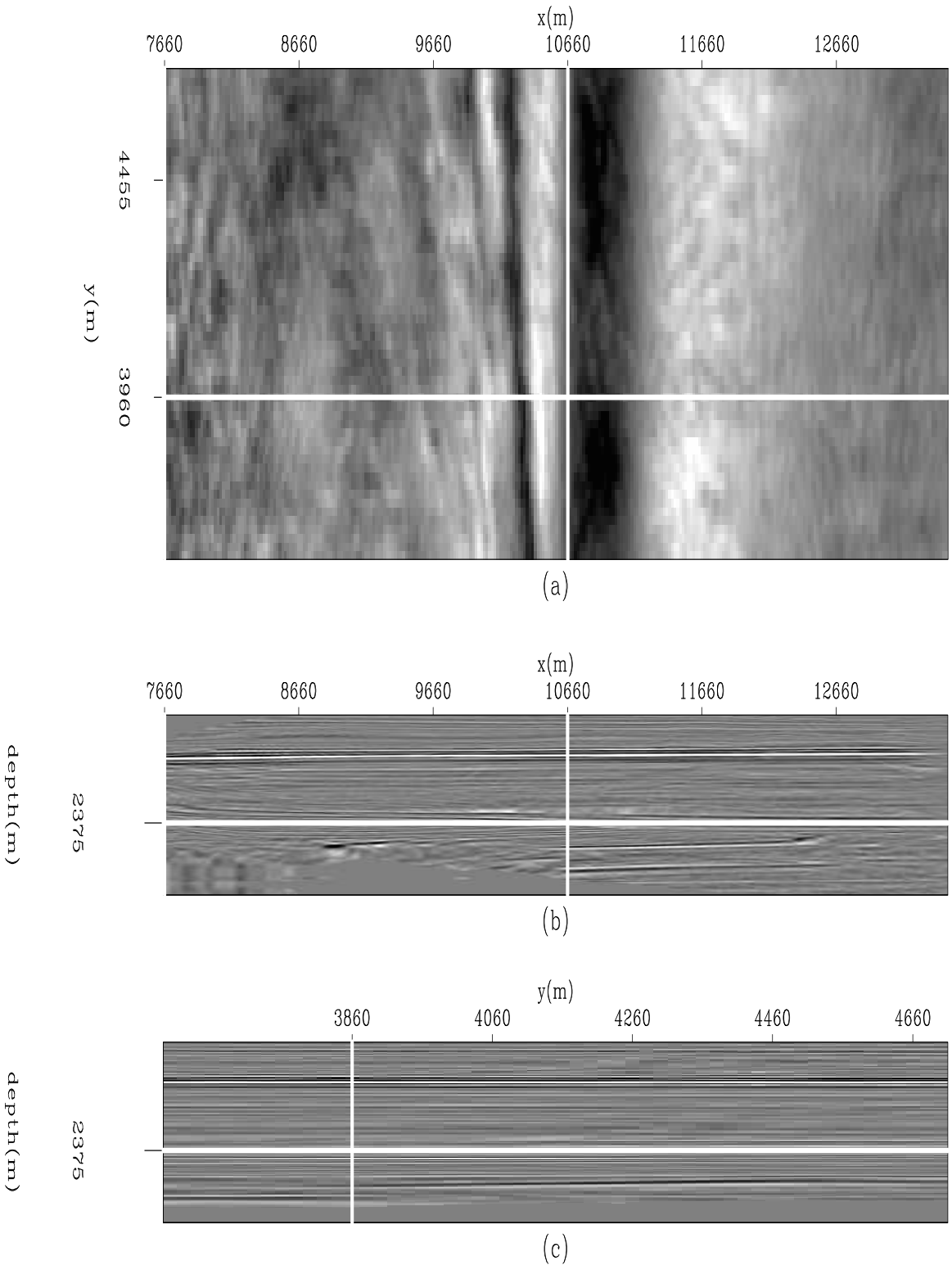

Figure 8: The result of flattening of Figure 7. This image illustrates the trade-off between data continuity preservation and flatness. We used a smoothing parameter ϵ of 1.0 to make this figure. Had we used a smaller ϵ, the output would be flatter but artifacts would be introduced into the data. (a) The horizon slice at depth=2375 m. (b) An in-line section at y=3960 m. (c) A cross-line section at x=10660 m. jesse1-elf_flat [ER]

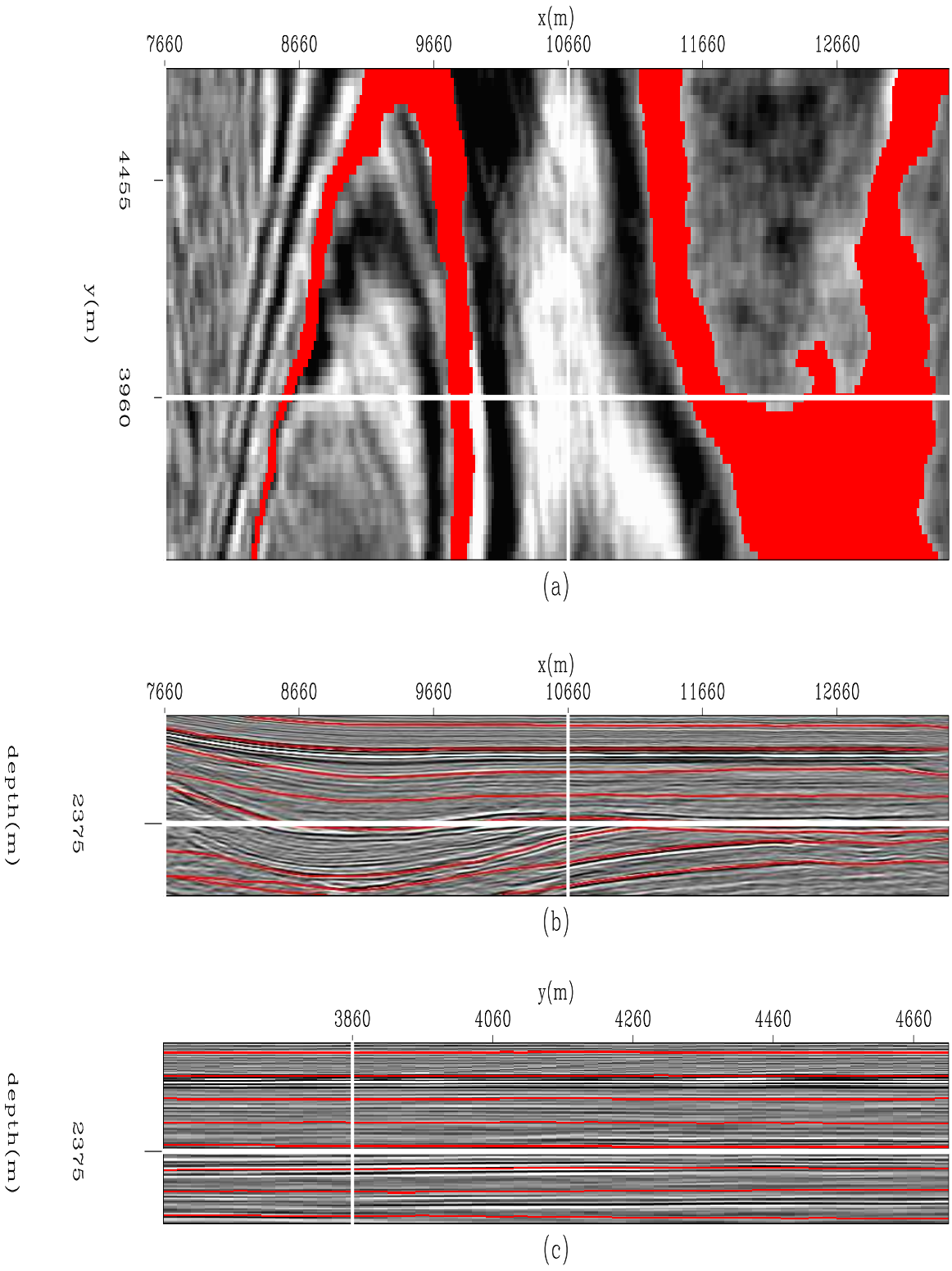

Figure 9: The result of overlaying tracked horizons on the image in Figure 7. Because the smoothing parameter ϵ is 0.0, horizons that lead to the angular unconformity are tracked. (a) The depth slice at depth=2375 *m*. (b) An in-line section at y=3960 *m*. (c) A cross-line section at x=10660 *m*. jesse1-elf.horizon_overlay [ER]

We use the crude binary weight cube shown in Figure 11 to identify areas where the inversion should ignore incorrect dips. This weight was created by manual picking. It should be noted that a suitable weight can also, in principle, be made from an automatic fault detector.

The flattening results are shown in Figure 12. Notice that the horizons are flat, even directly across from the fault. Also, notice the presence of a channel in the horizon slice. Figure 13 shows the original data with one unflattened horizon overlying it. It successfully tracks the horizon even across the fault. The time slice at the top of Figure 13 shows a horizon that is being partially cut by the fault at about x=2600 m.

DISCUSSION

To be fair, it should be noted that the flattening method only acts in the vertical dimension. This implicitly imposes several limitations. For instance, only vertical faults and beds that are not overturned can be properly restored. Also, the true stratigraphic thickness will be lost and replaced with the true vertical thickness at the reference trace.

The potential prestack applications of this method are numerous, and many of the challenges that must be overcome for poststack data are also present in prestack data.

The integrated time shift data has many potential uses. This method can easily be adapted to flatten data cubes on one or any particular combination of horizons. This would assist geologists in analyzing thicknesses for rates of deposition and timing of structural events.

CONCLUSIONS

We have developed a method to efficiently and robustly flatten 3D seismic cubes. This method uses an efficient implementation of the Fast Fourier transform within a non-linear iterative scheme. We demonstrated its effectiveness on a series of synthetic and field test cases.

Data cubes with vertical, laterally limited faults can be flattened by applying a residual weight. This allows horizons to be tracked across these faults.

We envision a tool that can be quickly and easily implemented by interpreters exploiting its superior computational performance. This method can provide an initial picking that interpreters can then adjust.

ACKNOWLEDGMENT

We would like to thank Chevron and Elf for the data used in this paper.

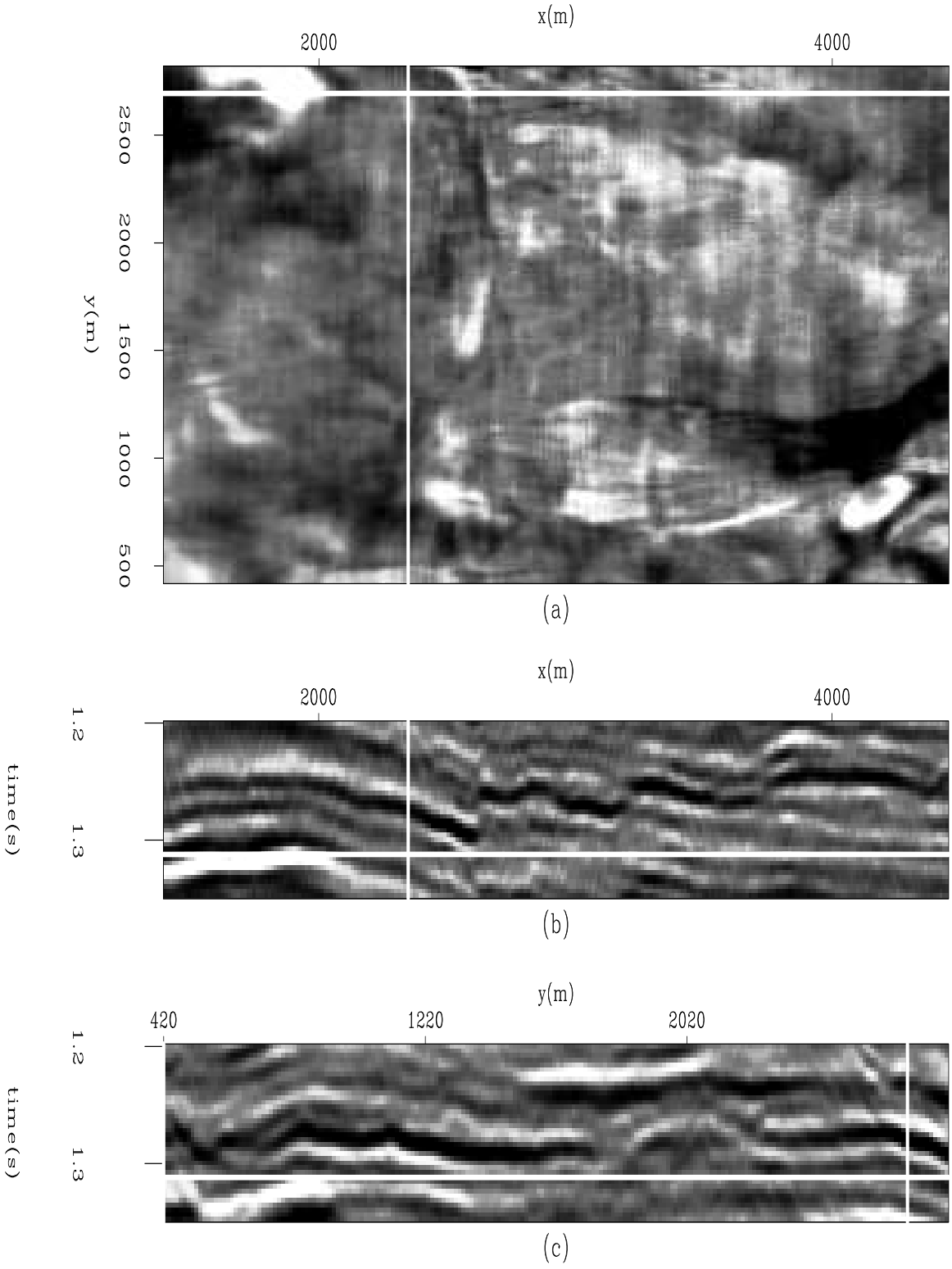

Figure 10: Faulted Chevron Gulf of Mexico data. The 2D vertical section shows fault with significant displacement (enough to cause our dip estimator to return erroneous values). (a) The time slice at time=1.32 *s*. (b) An in-line section at y=2693 *m*. (c) A cross-line section at x=2348 *m*. jesse1-shoal [ER]

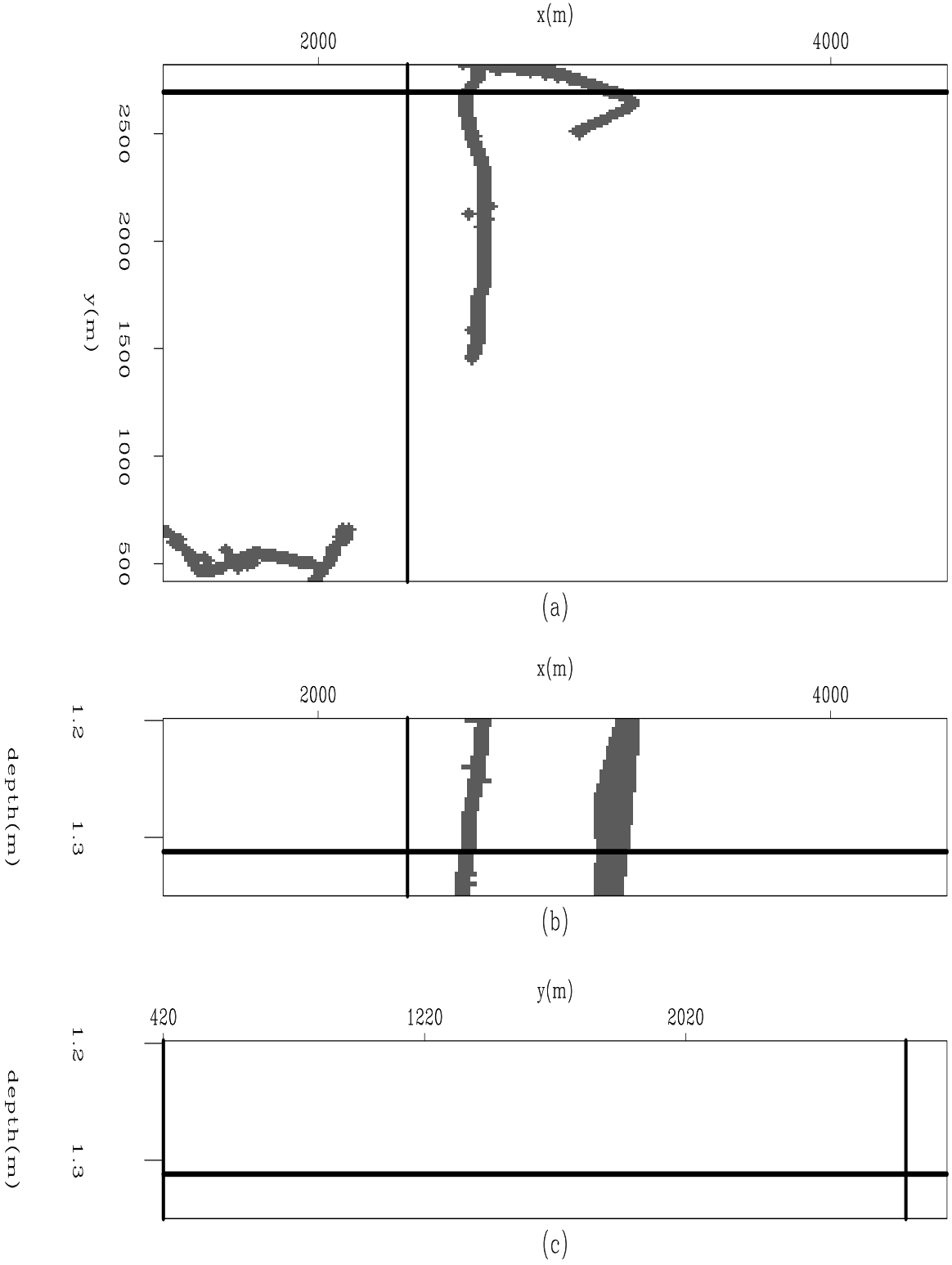

Figure 11: A weight identifying the faults data displayed in 10. (a) The time slice at time=1.32 *s*. (b) An in-line section at y=2693 *m*. (c) A cross-line section at x=2348 *m*. jesse1-shoal.wt [ER]

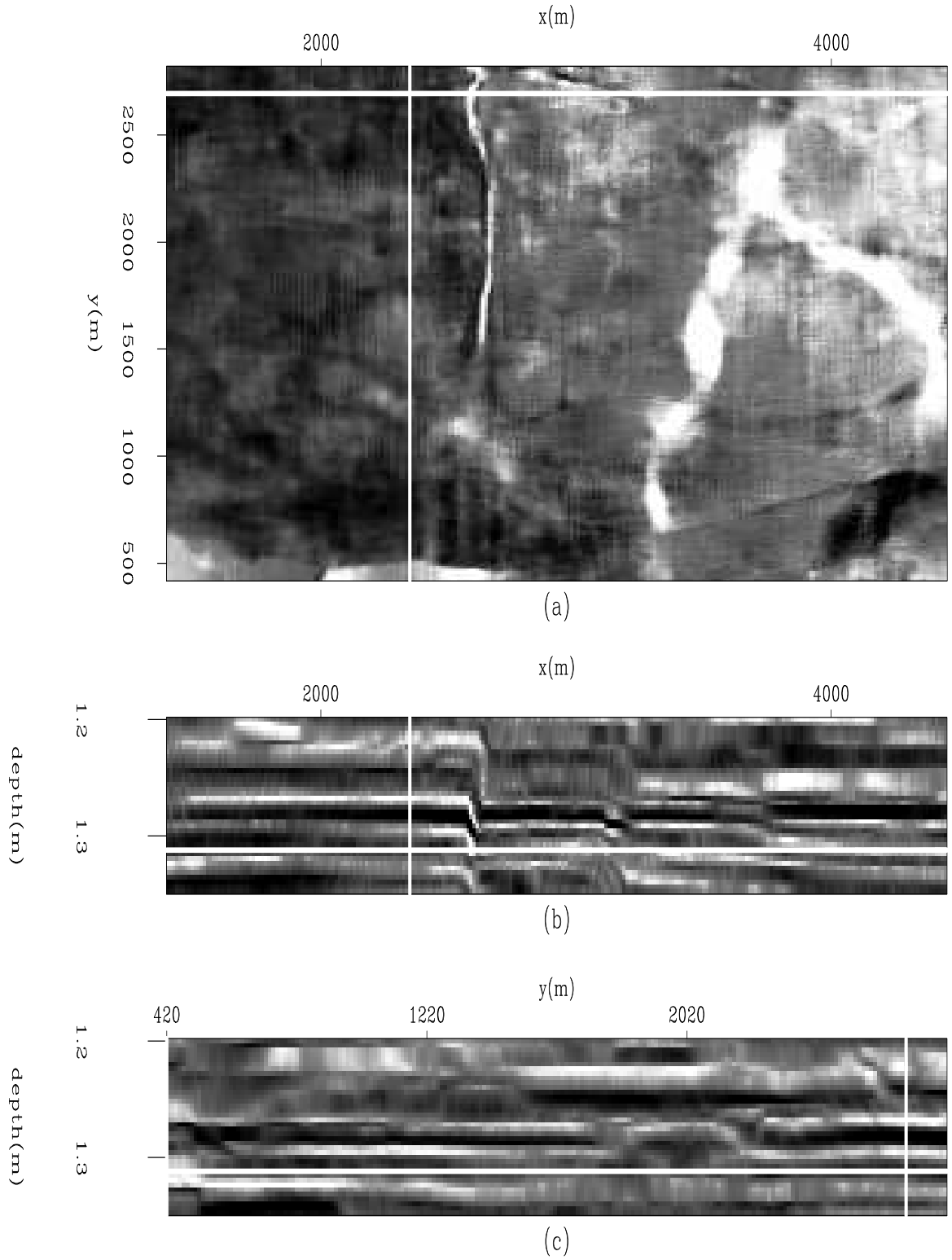

Figure 12: The result of flattening of Figure 10. (a) The horizon slice at time=1.32 *s*. (b) An in-line section at y=2693 *m*. (c) A cross-line section at x=2348 *m*. jesse1-shoal.flat [ER]

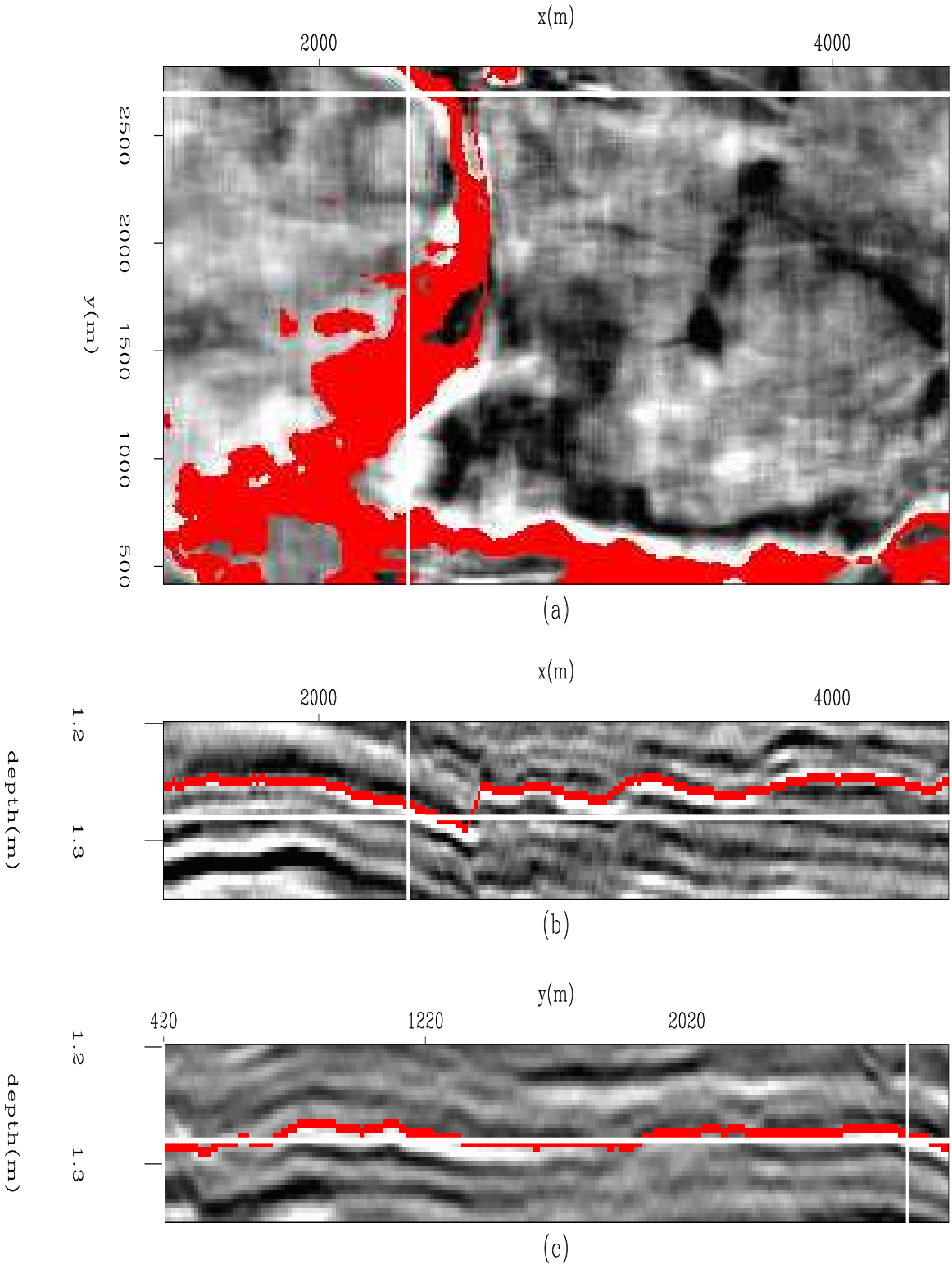

Figure 13: The result of overlaying a tracked horizon on the image in Figure 10. (a) The time slice at time=1.28 *s*. (b) An in-line section at y=2693 *m*. (c) A cross-line section at x=2348 *m*. jesse1-shoal.horizon_overlay [ER]

REFERENCES

Bienati, N., and Spagnolini, U., 1998, Traveltime picking in 3d data volumes: 60th EAGE Meeting, Extended Abstracts.

Bienati, N., and Spagnolini, U., 2001, Multidimensional wavefront estimation from differential delays: IEEE Trans on Geoscience and Remote Sensing, , no. 3, 655–664.

Bienati, N., Nicoli, M., and Spagnolini, U., 1999a, Automatic horizon picking algorithms for multidimensional data: 61st EAGE Meeting, Helsinki, Extended Abstracts.

Bienati, N., Nicoli, M., and Spagnolini, U., 1999b, Horizon picking for multidimensional data: an integrated approach: Proc. 6th International Congress of the Brazilian Geophysical Society, Rio de Janeiro.

Blinov, A., and Petrou, M., 2003, Reconstruction of 3d horizons from 3d seismic data sets: SPIE 10th International Symposium on Remote sensing, Remote Sensing fro environmental monitoring, GIS applications and Geolgy III.

Claerbout, J. F., 1992, Earth Soundings Analysis: Processing Versus Inversion: Blackwell Scientific Publications.

Claerbout, J., 1999, Geophysical estimation by example: Environmental soundings image enhancement: Stanford Exploration Project, `http://sepwww.stanford.edu/sep/prof/`.

Farlow, S. J., 1993, Partial differential equations for scientists and engineers: Dover Publications.

Fomel, S., 2002, Applications of plane-wave destruction filters: Geophysics, **67**, no. 6, 1946–1960.

Ghiglia, D. C., and Romero, L. A., 1994, Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods: Optical Society of America, **11**, no. 1, 107–117.

Guitton, A., Claerbout, J. F., and Lomask, J. M., 2004, First order interval velocity estimates without picking: 74th SEG Meeting, Extended Abstracts.

Guitton, A., Lomask, J., and Fomel, S., 2005, Non-linear estimation of vertical delays with a quasi-Newton method: SEP–**120**, 167–178.

James, H., Peloso, A., and Wang, J., 2002, Volume interpretation of mult-attribute 3d surveys: First Break, **20**, no. 3, 176–180.

Lee, R., 2001, Pitfalls in seismic data flattening: The Leading Edge, **20**, no. 01, 160–164.

Leggett, M., Sandham, W. A., and Durrani, T. S., 1996, 3d horizon tracking using artificial neural networks: First Break, **14**, no. 11, 413–418.

Lomask, J., 2003a, Flattening 3d seismic cubes without picking: 73th SEG Meeting, Extended Abstracts.

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

Stark, T. J., 2004, Relative geologic time (age) volume- relating every seismic sample to a geologically reasonable horizon: The Leading Edge, **23**, no. 9, 928–932.