VII. WIGGINS-TYPE ALGORITHM

In this chapter an algorithm is derived which iteratively estimates an inverse filter such that $U(X,\alpha_1,\alpha_2)$ is maximized. It is called a Wiggins-type algorithm because it closely follows an ingenious procedure applied by Wiggins (26) in development of MED. An efficient implementation of the algorithm in the frequency domain is outlined and its properties are illustrated by its application to synthetic seismograms. Results obtained by deconvolving real seismograms are compared to those achieved using other techniques.

A. Derivation

Let y_{ij} be the i-th sample of the signal recorded on channel j where there are n samples per channel and m channels. Let f_k be the k-th sample of an inverse filter having nf samples. The reflectivity series, x_{ij}, is the convolution of the recorded signal and inverse filter.

$$x_{ij} = \sum_{k=1}^{nf} y_{i-k,j}\, f_k \tag{7-1}$$

The variable norm for the Wiggins-type algorithm is

$$U(X,\alpha_1,2) = \log \prod_{j=1}^{m} \frac{\left(\frac{1}{n}\sum_{i=1}^{n} |x_{ij}|^{\alpha_1}\right)^{n/\alpha_1}}{\left(\frac{1}{n}\sum_{i=1}^{n} |x_{ij}|^{2}\right)^{n/2}} \tag{7-2}$$

where $\alpha_1 > 2$.

To find an inverse filter that gives a relative maximum of $U(X,\alpha_1,2)$ take the partial of U with respect to the filter coefficients. When the

partial vector is zero, a relative maximum is found.

$$\frac{\partial U(X,\alpha_1,2)}{\partial f_k} = \sum_{j=1}^{m} \left[\frac{1}{\mu_{1j}} \sum_{i=1}^{n} |x_{ij}|^{\alpha_1 - 1} \operatorname{sgn}(x_{ij}) \frac{\partial x_{ij}}{\partial f_k} - \frac{1}{\mu_{2j}} \sum_{i=1}^{n} x_{ij} \frac{\partial x_{ij}}{\partial f_k} \right] \tag{7-3}$$

where

$$\mu_{1j} = \frac{1}{n} \sum_{i=1}^{n} |x_{ij}|^{\alpha_1}$$

and

$$\mu_{2j} = \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2 \quad .$$

From Equation (7-1),

$$\frac{\partial x_{ij}}{\partial f_k} = y_{i-k,j} \quad . \tag{7-4}$$

Setting Equation (7-3) to zero and using Equation (7-4) gives a non-linear expression relating an optimal reflectivity series to the recorded signal.

$$\sum_{j=1}^{m} \left(\frac{1}{\mu_{2j}} \sum_{i=1}^{n} x_{ij}\, y_{i-k,j} \right) = \sum_{j=1}^{m} \left(\frac{1}{\mu_{1j}} \sum_{i=1}^{n} |x_{ij}|^{\alpha_1 - 1} \operatorname{sgn}(x_{ij}) y_{i-k,j} \right) \tag{7-5}$$

To solve Equation (7-5) iteratively, assume x_{ij} in the sum on the left is defined by Equation (7-1) and all other occurrences are computed from the previous estimate. Then Equation (7-5) can be written,

$$\sum_{\ell=1}^{nf} \left[\sum_{j=1}^{m} \frac{1}{\mu_{2j}} \sum_{i=1}^{n} y_{i-\ell,j}\, y_{i-k,j} \right] f_{\ell} = \sum_{j=1}^{m} \left[\frac{1}{\mu_{1j}} \sum_{i=1}^{n} |x_{ij}|^{\alpha_1 - 1} \operatorname{sgn}(x_{ij}) y_{i-k,j} \right] \tag{7-6}$$

Using matrix notation Equation (7-6) can be simply expressed as

$$Rf = g \tag{7-7}$$

where

R is a weighted sum of the autocorrelation matrices of the inputs,
f is an unknown filter,

and

g is a weighted sum of crosscorrelations between the inputs and the current estimates of the reflectivity series to some power where their signs are retained.

B. Frequency Domain Implementation

The system represented by Equation (7-7) can be iteratively solved in either the time or frequency domain. The choice of method depends on the number of channels available and the resolution desired. Implementation in the frequency domain is appropriate for estimating long filters to provide high resolution. In this situation the method is computationally most efficient but requires many channels to constrain the solution.

Time domain methods have much greater flexibility, require fewer channels, and are most efficient for estimating short inverse filters. They are conceptually difficult to understand, so a frequency domain implementation is presented first. Time domain methods will be outlined in the next chapter.

Let capital letters represent the Fourier transforms of the corresponding time domain variables. A bar over a variable represents its complex conjugate. The following is a verbal step by step description of a flow chart used to program this algorithm.

Initialization is accomplished by setting the inverse to a unit spike and transforming,

$$f(t) \leftarrow \delta(t-t_0)$$
$$F(\omega) \leftarrow FFT(f(t)) \ . \tag{7-8}$$

The shape of the final inverse filter is invariant to the initial time shift t_0. The length of f(t) should be that of the recorded trace. Vectors R and G are used to accumulate the weighted auto and crosscorrelations of the channels. They are equivalent to R and g of Equation (7-7). At the start of each iteration they are zeroed,

$$R(\omega) \leftarrow 0$$
$$G(\omega) \leftarrow 0 \ . \tag{7-9}$$

Each trace, y(t), is read and transformed.

$$Y(\omega) \leftarrow FFT(y(t)) \ . \tag{7-10}$$

Before transforming the input trace it may be necessary to double its length by zero padding to prevent aliasing due to performing convolution by multiplication in the frequency domain. This is especially important when the inverse filter has a lot of high frequency energy.

The reflectivity series is computed

$$X(\omega) \leftarrow F(\omega) \cdot Y(\omega) \tag{7-11}$$

and inverse transformed

$$x(t) \leftarrow \text{IFFT}(X(\omega)) \quad . \tag{7-12}$$

The scale factors,

$$\mu_1 \leftarrow \frac{1}{\frac{1}{n} \sum_t |x_t|^{\alpha_1}}$$

and

$$\mu_2 \leftarrow \frac{1}{\frac{1}{n} \sum_t x_t^2} \tag{7-13}$$

are found and the current estimate of the reflectivity series weighted,

$$x'(t) \leftarrow |x(t)|^{\alpha_1 - 1} \, \text{sgn}(x(t)) \tag{7-14}$$

and its transform computed,

$$X'(\omega) \leftarrow \text{FFT}(x'(t)) \quad . \tag{7-15}$$

The auto-and cross-correlations are found, then weighted and added into the accumulation vectors,

$$\begin{aligned} R(\omega) &\leftarrow R(\omega) + \mu_1 \, \overline{Y}(\omega) Y(\omega) \\ G(\omega) &\leftarrow G(\omega) + \mu_2 \, \overline{Y}(\omega) X'(\omega) \quad . \end{aligned} \tag{7-16}$$

Steps described by Equations (7-10) to (7-16) are performed on each trace. When all have been processed the inverse filter is found by spectral division,

$$F(\omega) \leftarrow \frac{G(\omega)}{(R(\omega) + \lambda \sum_{\omega'} R(\omega'))} \quad . \tag{7-17}$$

This step is shown by R. Gray (31) to be asymptotically equivalent to solving a Toeplitz system in the time domain. The parameter λ is analogous to the white noise added to the zero lag autocorrelation. It is proportional to the inverse of the squared signal to noise ratio of the recorded data.

An alternate approach is

$$F(\omega) \leftarrow \frac{G(\omega)}{\max[R(\omega), \lambda \sum_{\omega'} R(\omega')]} \quad . \tag{7-18}$$

This has the advantage of only distorting the spectrum below a given threshold. The peaks and flanks are not modified.

In practice it was found that setting the d.c. term of $G(\omega)$ to zero improves the stability of the algorithm. Physically this makes sense because source wavelets are zero mean.

The new inverse filter is scaled so it has unit power,

$$F(\omega) \leftarrow \frac{F(\omega)}{[\sum_{\omega'} \overline{F}(\omega')\ F(\omega')]} \quad . \tag{7-19}$$

The norms for each channel are then compared with the norms of the previous iteration. Convergence is assumed if the differences are less than some prespecified tolerance. If another iteration is required, steps described by Equations (7-9) to (7-19) are repeated.

The final inverse is used to deconvolve the input data by the steps described by Equations (7-10) through (7-12). The deconvolved output must be time shifted by t_0 of Equation (7-8) to correct for the time delay introduced by the inverse filter.

C. Application to Synthetics

Synthetic data sets are used to show the properties of the frequency domain implementation when the density of events and noise levels are varied.

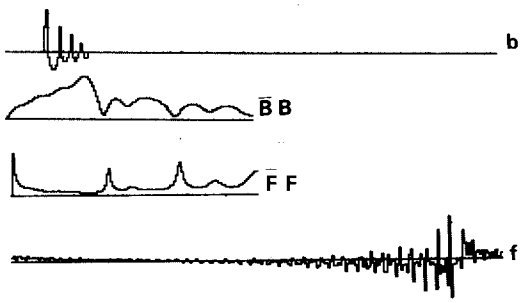


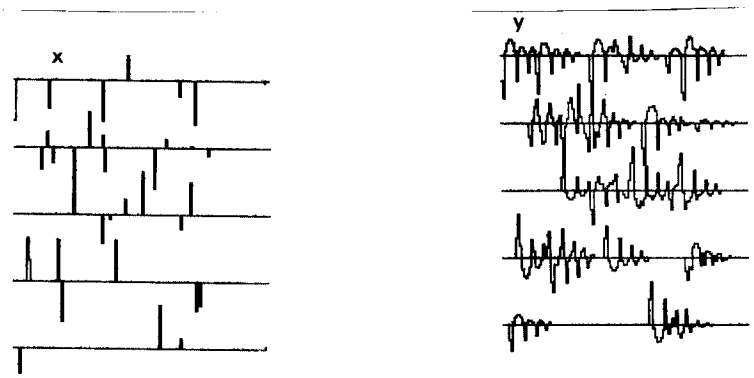
FIGURE 7-1. Synthetic wavelet, its spectrum, estimated inverse spectrum, and inverse wavelet.

The same source wavelet is used for all examples. It is displayed in Figure 7-1 where the inverse transform, $F(\omega)$ was found from

$$F(\omega) = \frac{\overline{B}(\omega)}{\overline{B}(\omega)\ B(\omega) + (1 \times 10^{-8}) \sum_{\omega'} \overline{B(\omega')}\ B(\omega')} \quad . \tag{7-20}$$

The inverse wavelet was computed by inverse transforming $F(\omega)$.

The length of the source wavelet is 23 samples, that of the synthetic seismograms is 128 samples. Events were not allowed after sample 105 to eliminate truncation effects. For real data, the end effects can be minimized by using a triangular or cosine taper. Convergence of the algorithm was assumed when the norm of the estimate of the reflectivity series of the first trace changed by less than .001 from that of the previous iteration. All examples were initialized by a unit spike in the first sample of the inverse filter. The trace lengths were doubled to eliminate convolutional aliasing.

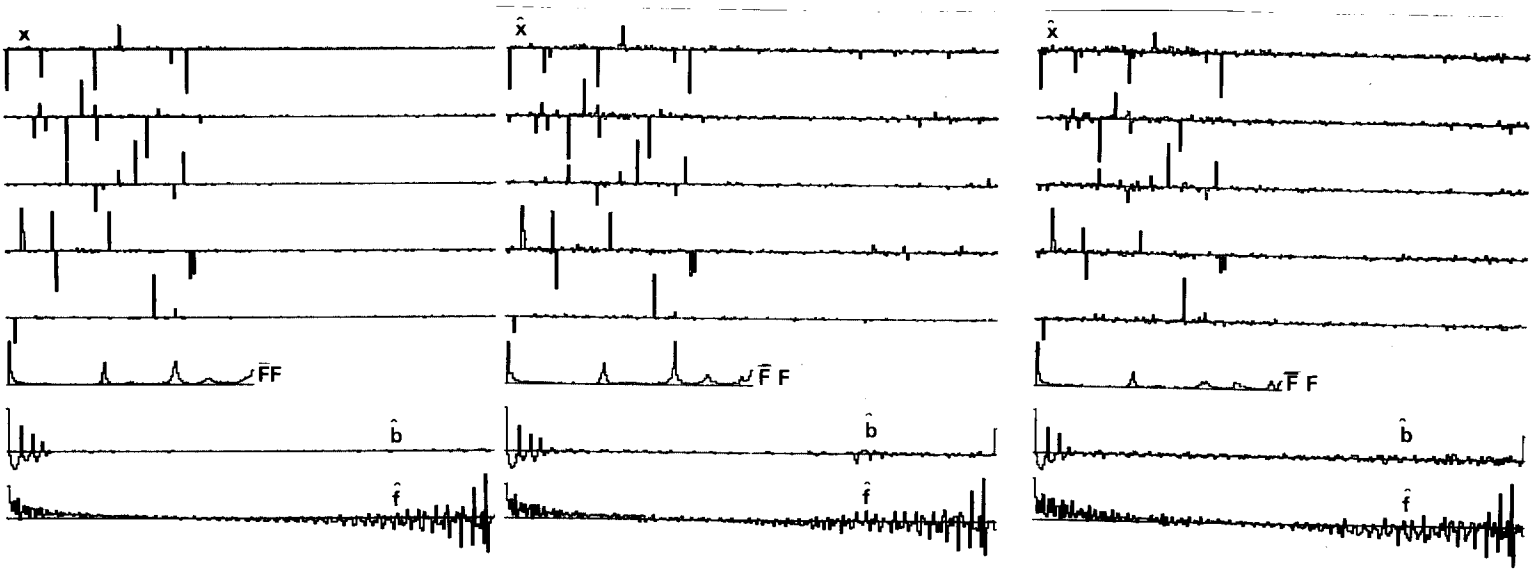


Reflectivity Series
.05 Density

Synthetics
.001 Additive Noise

FIGURE 7-2. Synthetics (right) for low density reflectivity series (left).

The first test was practically noise free and had a low density of events. Figure 7-2 displays the inputs. On the left is the reflectivity series with density (.05) or about one event per twenty samples. Each event has its time and amplitude drawn from a uniform distribution. On the right are the synthetics resulting from convolving the source wavelet (Figure 7-1) with the reflectivity series and adding noise drawn from a uniform distribution. The noise level is about .001 that of the largest event.



α = 2.2
12 iterations

α = 4.
5 iterations

α = 6.
4 iterations

FIGURE 7-3. Results for low density reflectivity series (.05), low noise (.001). Each column displays for a different α the estimated reflectivity series, inverse spectrum, forward wavelet, and inverse wavelet.

The results of the first test are shown in Figure 7-3 for α = 2.2, 4, and 6. The estimates of the source wavelet, inverse spectrum, and inverse filter are displayed below the deconvolutions. The source wavelet was estimated using Equation (7-20) where the roles of the source and inverse are reversed.

For α_1 = 6 the algorithm concentrates on spiking the high amplitude events. The low amplitude noise created by a slightly incorrect estimate of the inverse filter is not considered. For α_1 = 4, the low amplitude noise is reduced and fairly good results are obtained. The best results were achieved with α_1 = 2.2. Close comparison with Figures 7-1 and 7-2 shows a nearly perfect deconvolution and estimate of the source.

These results imply that setting α_1 small yields the best deconvolution. When α_1 = 2 nothing happens as U = 1. When α_1 < 2 the algorithm tries to

equalize the amplitudes on all channels. The results look like uniform noise. It seems that α_1 just barely greater than 2 would yield the best deconvolution.

This is wrong as the next example (Figure 7-4) shows. Letting $\alpha_1 = 4$ and 6 were also tried. When $\alpha_1 = 6$ the algorithm spiked one trace. For $\alpha_1 = 4$ the deconvolution was a little noisier than $\alpha_1 = 2.5$ but converged in eight iterations.

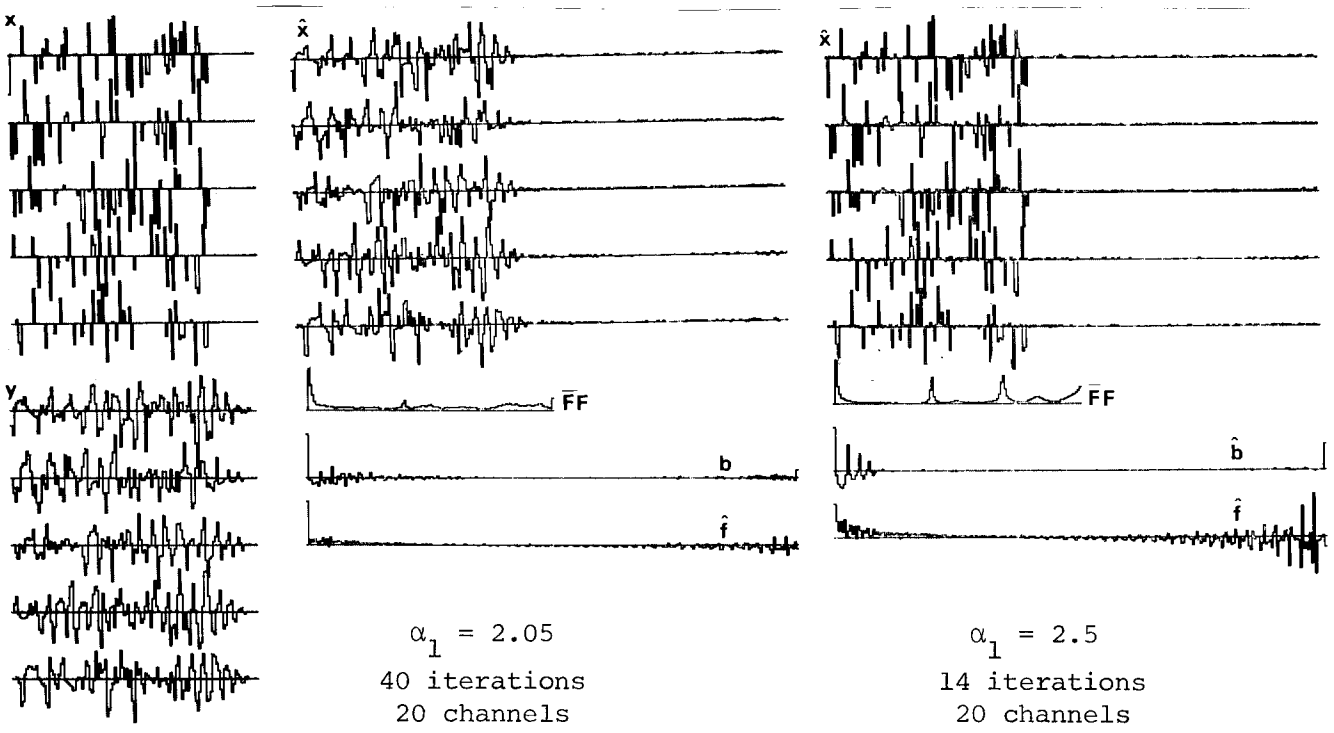


FIGURE 7-4. High density (.3) reflectivity series and synthetics on the left. On the right are results for $\alpha_1 = 2.05$ and 2.5. $\alpha_1 = 2.5$ yields a near perfect deconvolution. $\alpha_1 = 2.05$ hardly changes the inputs.

Several tests on synthetics with varying levels of additive noise were performed. Figure 7-5 shows results from synthetics with an ambient noise level of .1 and reflectivity series density .05. The deconvolution of $\alpha_1 = 2.5$ is better than $\alpha_1 = 4$ for all but the fifth channel. As in Figure 7-3,

when $\alpha_1 = 4$, the small amplitude noise is not reduced. This is because higher norm ratios do not see the smaller amplitudes.

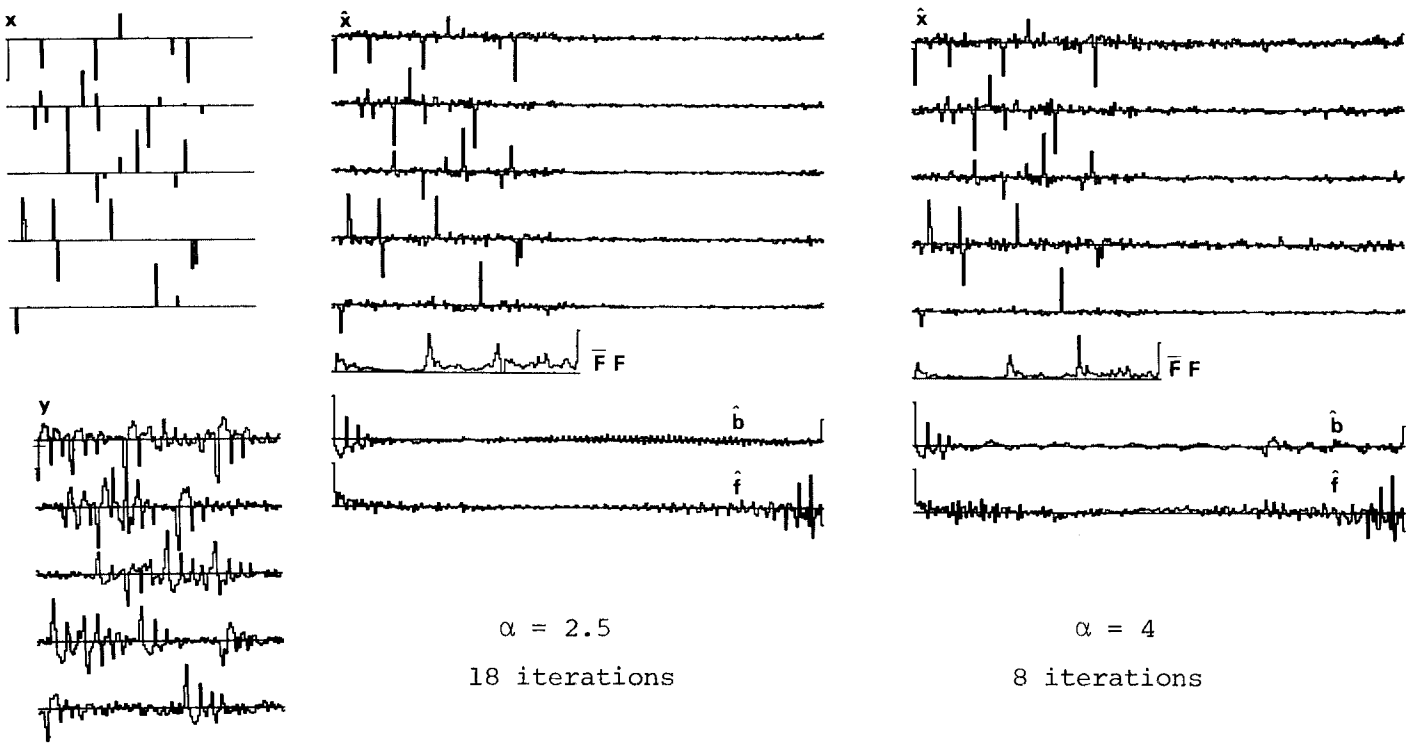


FIGURE 7-5. Results for moderate noise level. RMS signal to noise ratio is 3.8, ambient level is .1.

When the ambient noise level reached about .25 the results started degenerating. Figure 7-6 shows deconvolutions for the same data as in Figure 7-5 but with noise levels of .3 and .7. The inverse spectrums have some similarity to the spectrum of the source wavelet. It seems that at high noise levels the inverse filter that maximizes spikiness is the time reversed source wavelet -- a matched filter.

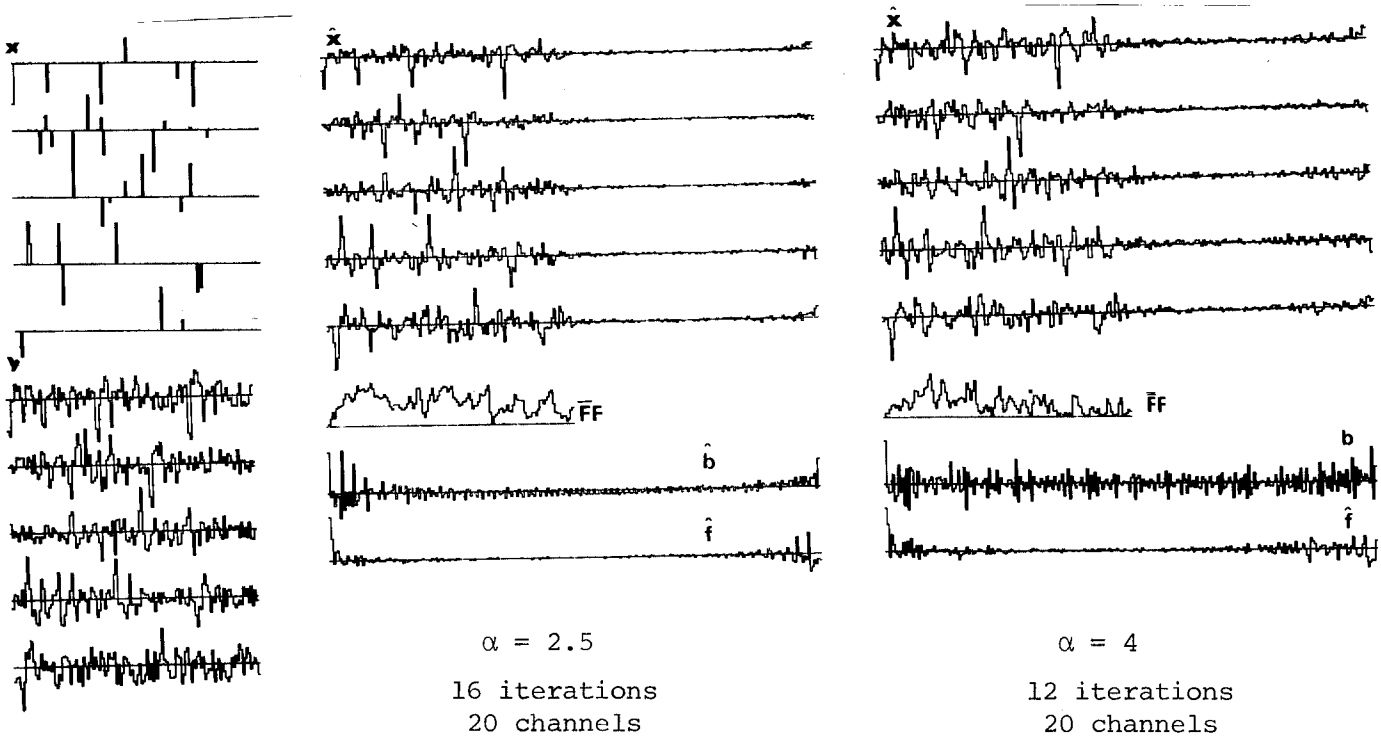


FIGURE 7-6. Results for high noise levels. For the $\alpha = 2.5$ deconvolution the ambient noise level was .3 and RMS signal to noise ratio 1.1. For the $\alpha = 4$ case the ambient noise is .75 and signal to noise ratio .42. Inputs for the $\alpha = 2.5$ case are displayed on the left.

D. Application to Field Recorded Data

Marine seismic data recorded in 1975 for the USGS in the Gulf of Alaska was available for testing. The source was twenty-two airguns of 1200 cu. in. size. Six seconds of data were recorded at a sampling rate of 4 ms on each of forty-eight channels. The near trace offset was 200M and the far trace 2550M.

The data was chosen for deconvolution because bubble oscillations of the reflected pressure pulse have noticeable amplitude for 250 ms after the initial pulse. These low frequency oscillations obscure primaries and are difficult to remove with stacking.

Two seconds of data on each of the near sixteen traces were used as inputs to the algorithm. A triangular taper was applied to reduce truncation effects and the d.c. component of each trace removed. Three deconvolutions for α = 2.5, 4, and 6 are presented in Figures 7-9, 7-10, and 7-11. Only two seconds of the near twenty-four traces are displayed. The same time-varying gain correction was applied to the original data and deconvolutions before plotting.

The inverse spectrums, estimated source wavelets, and inverse filters are displayed in Figure 7-7. The effect of increasing α is clearly related to the power present in the higher frequencies of the spectrums. The estimated source wavelets are ringy because of a hole in the inverse spectrum at 60 Hz.

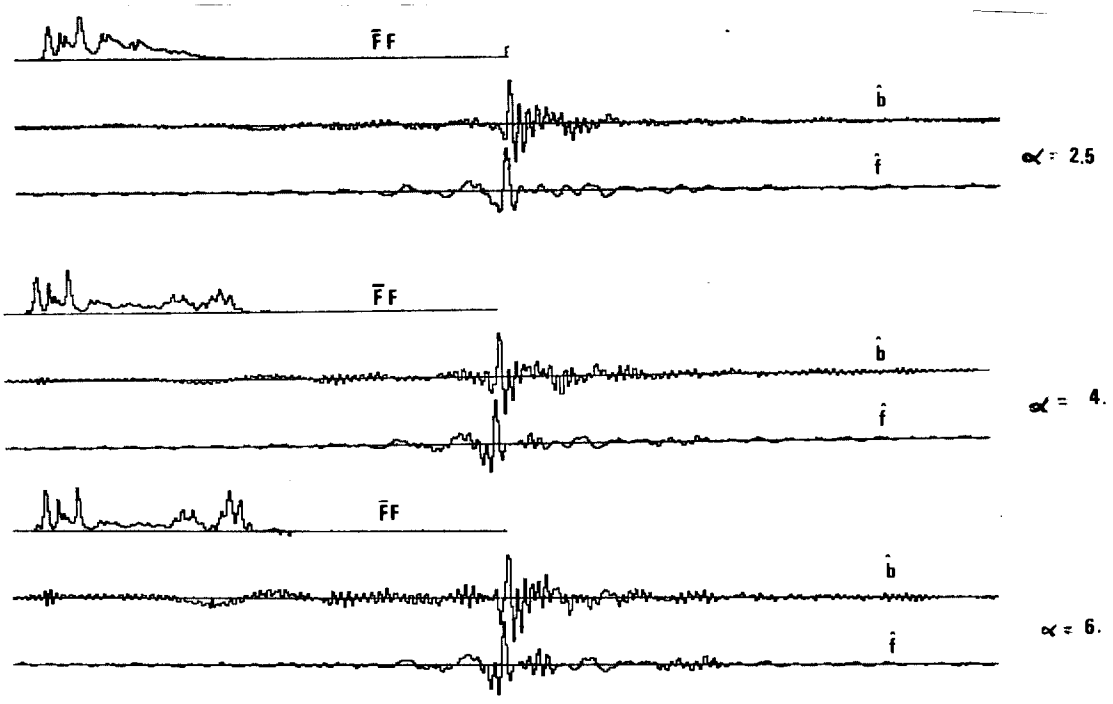


FIGURE 7-7. Spectrums of the inverse filters, estimated source wavelets, and inverse filters for variable norm deconvolutions on field data.

For comparison the original data is presented in Figure 7-8. The deconvolutions have eliminated the prominent bubble oscillation labeled on the original data. The deconvolution for $\alpha = 6$ looks noisy but when viewed from the side at a low angle it shows the layering near the sea floor in greater detail than any of the others. Deeper in the section it seems to have degraded some reflections. This is because the algorithm "sees" only the high amplitude sea floor reflections.

Figure 7-12 is a result obtained using spectral smoothing, a multi-channel predictive deconvolution technique developed by Godfrey and Claerbout. The results obtained using the variable norm algorithm seem to have done a superior job of suppressing noise and eliminating the bubble oscillations.

An implementation of Parsimonious Deconvolution [Equation (6-2)] developed by Godfrey yielded the results presented in Figure 7-13. It is very similar to that obtained using the variable norm method. Godfrey computed the entropy and kurtosis for all the results presented. Those with maximum kurtosis and minimum entropy were obtained using the variable norm method.

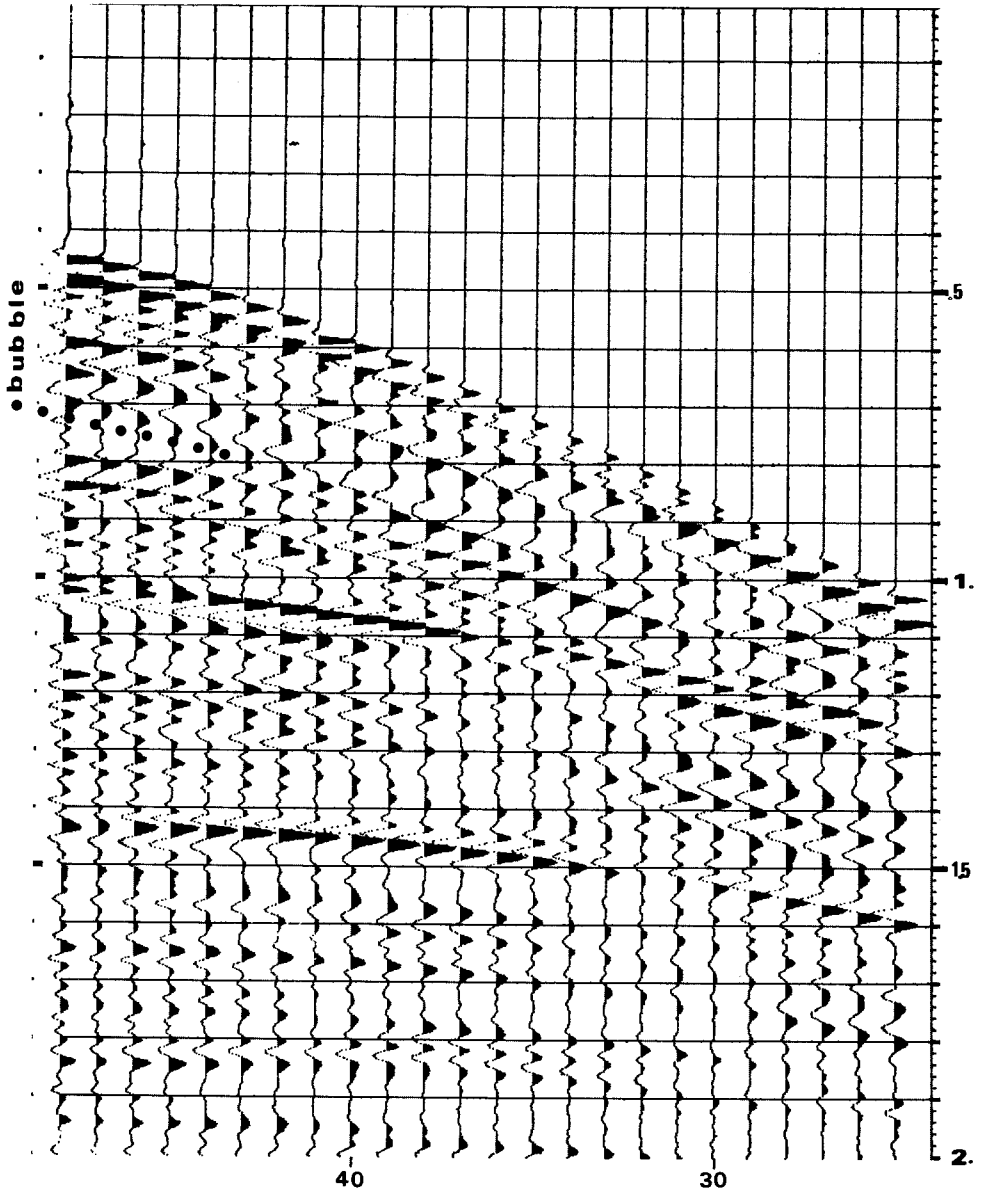


FIGURE 7-8. Original data. Bubble pulse is denoted by the dots between .7 and .8 sec. on the left.

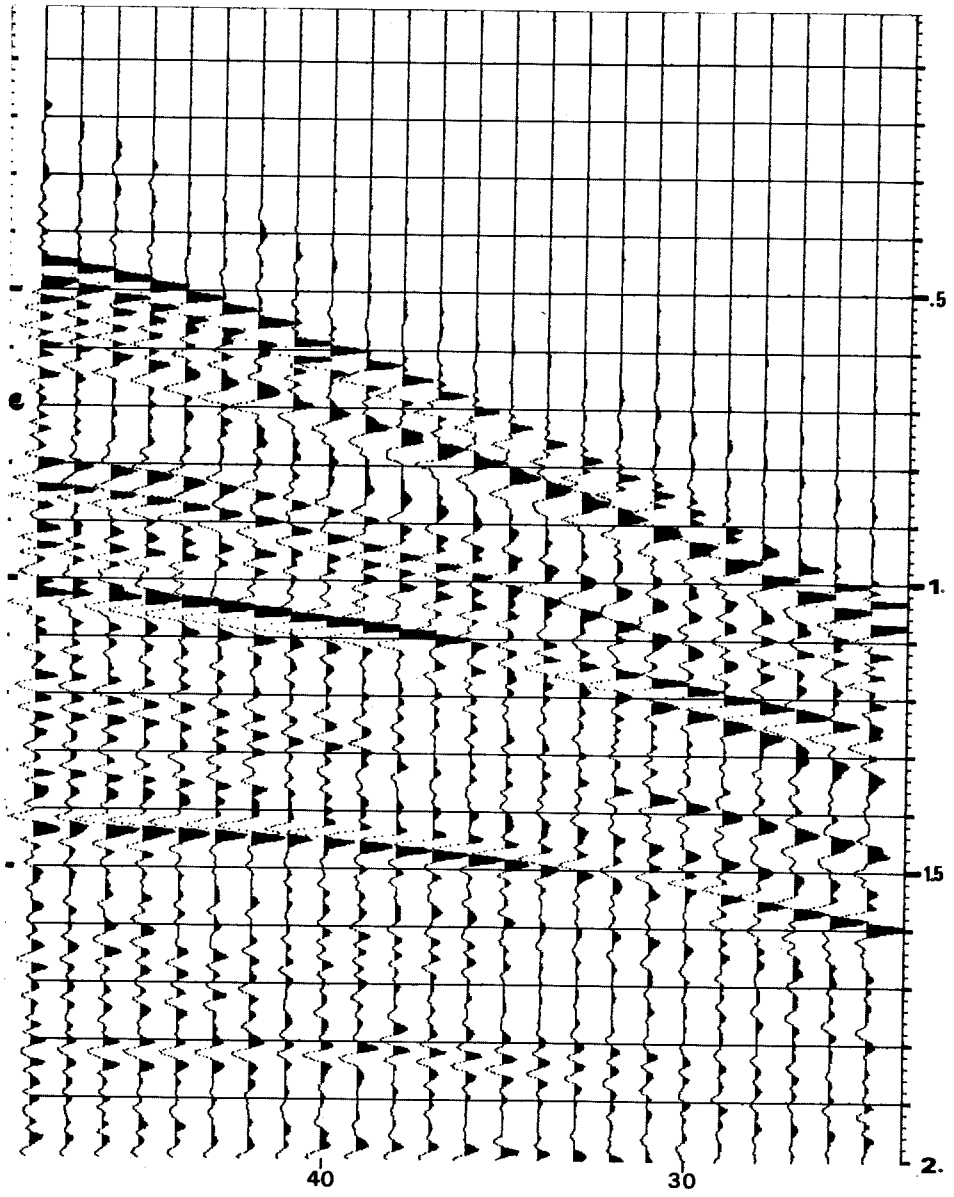


FIGURE 7-9. Variable norm deconvolution for $\alpha = 4$ after 6 iterations.

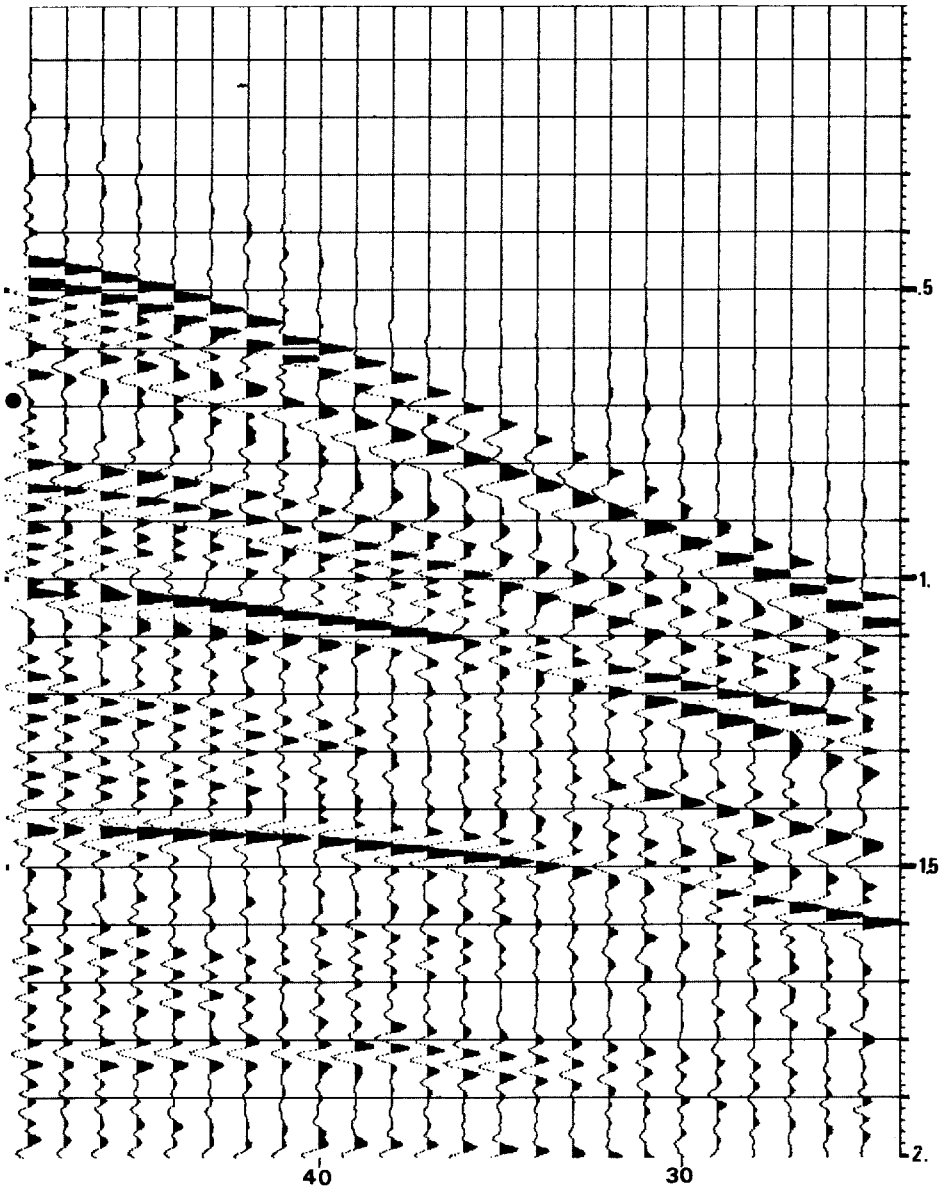


FIGURE 7-10. Variable norm deconvolution for α = 2.5 after 9 iterations.

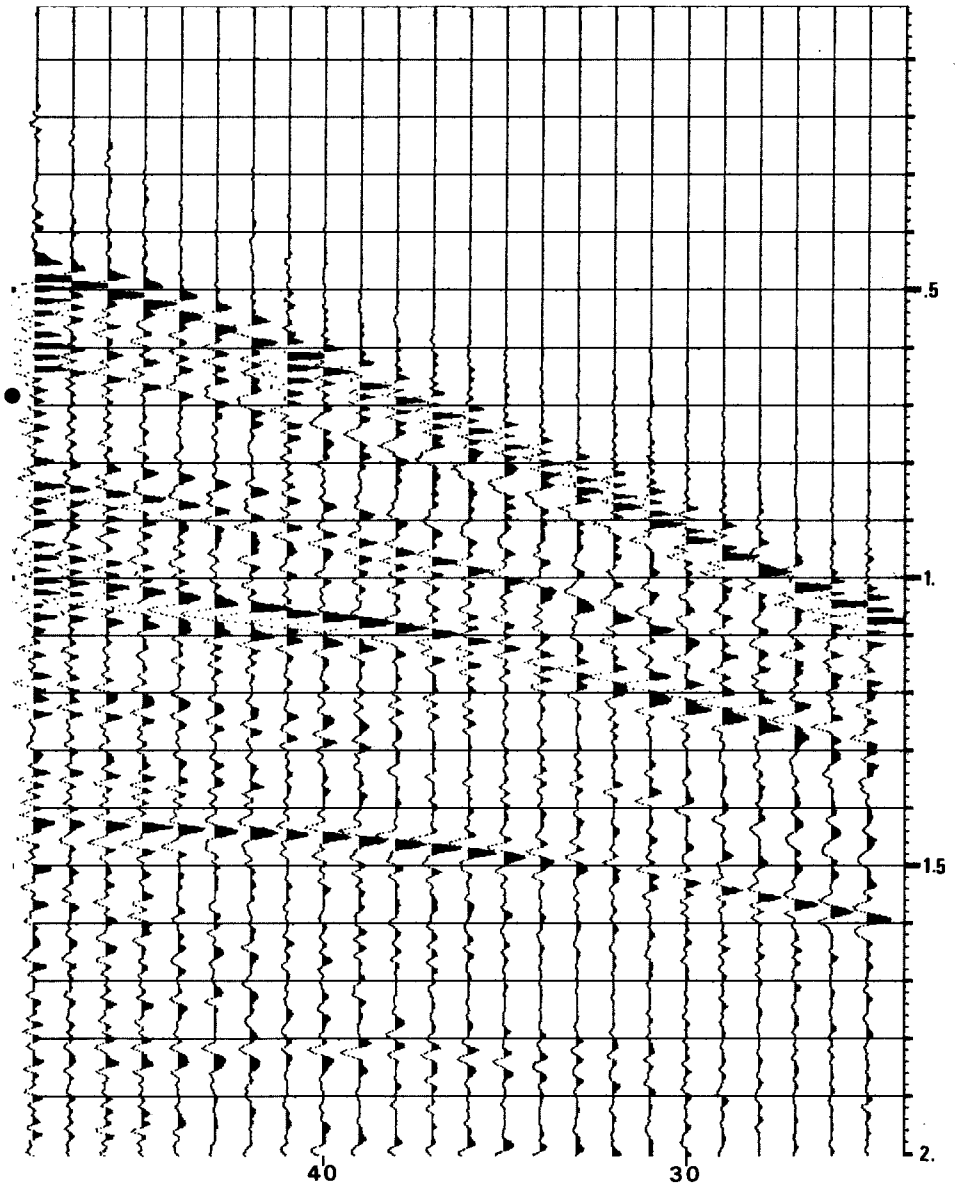


FIGURE 7-11. Variable norm deconvolution for α = 6 after 5 iterations.

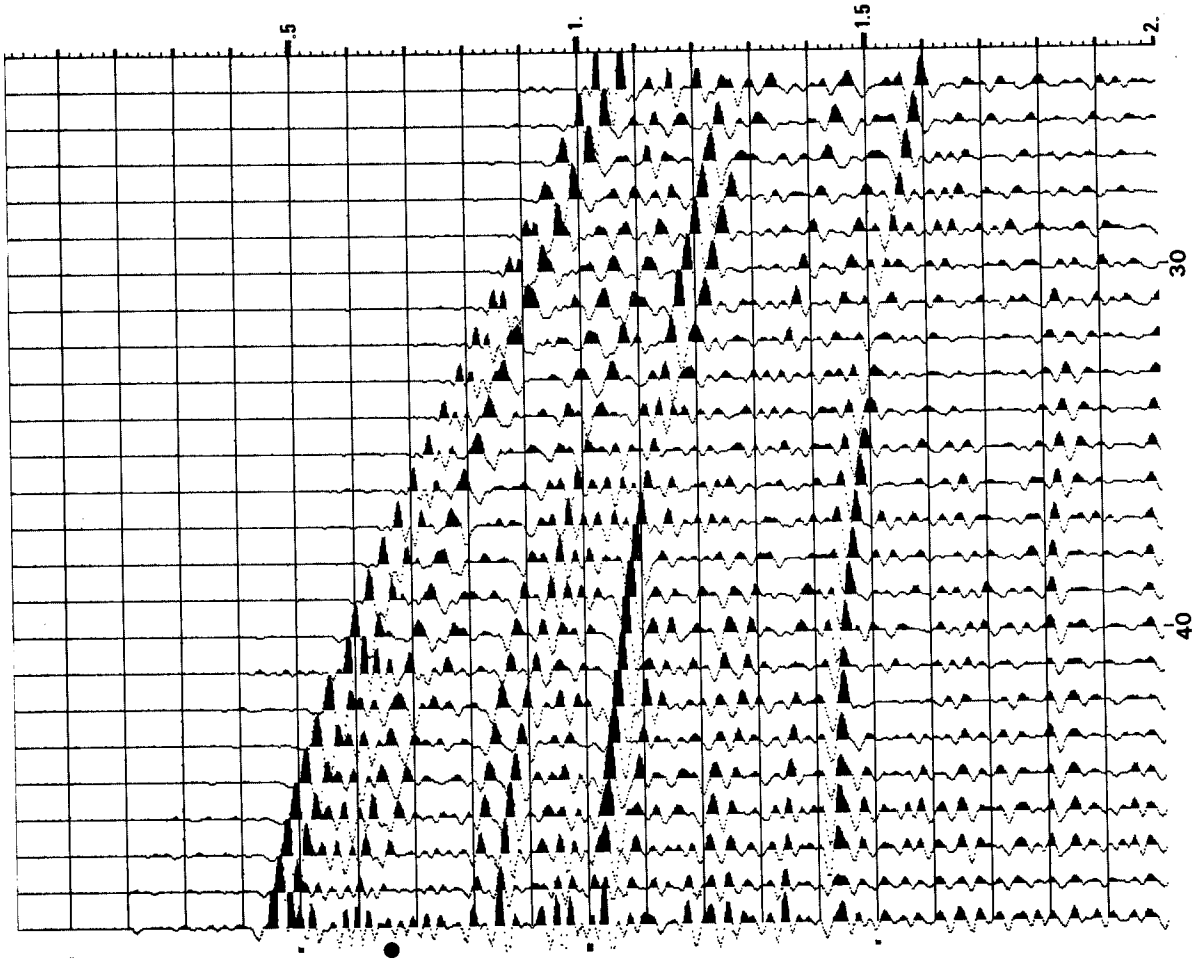


FIGURE 7-13. Parsimonious Deconvolution.

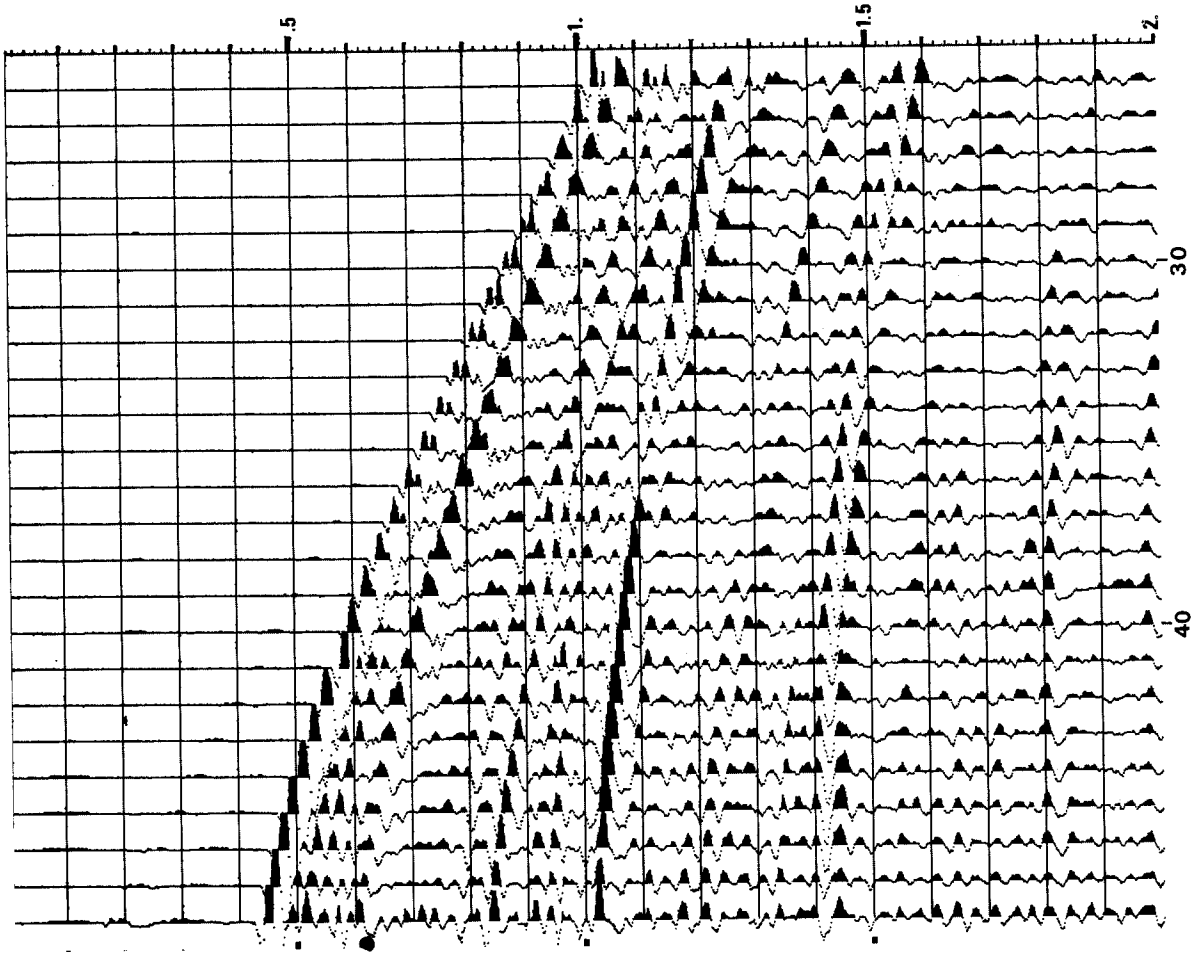


FIGURE 7-12. Spectral Smoothing Deconvolution.