LEAST-SQUARES SEPARATION OF SIGNAL AND NOISE USING MULTIDIMENSIONAL FILTERS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF GEOPHYSICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Raymond Lee Abma
August 1995

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Jon F. Claerbout (Principal Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Gregory C. Beroza

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Francis Muir

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Simon Klemperer

Approved for the University Committee on Graduate Studies:

Least-squares separation of signal and noise with multidimensional filters

Raymond Lee Abma, Ph.D.
Stanford University, 1995

ABSTRACT

Multidimensional filters are used to characterize and separate seismic signal and noise. This separation may be achieved with either simple filtering or by an inversion process that involves solving a system of regressions. The system of regressions describes the expected properties of the noise and signal by using filters and weights.

Signal and noise separation by filtering may be done by either f-x prediction or t-x prediction. Both these techniques are prediction-error filters that define the noise as the prediction error. The f-x prediction is shown to be equivalent to a t-x prediction with a very long filter length in time. While filtering is simple, it can produce spurious events and attenuate the amplitude of the signal. A technique that separates signal and noise with an inversion can eliminate these weaknesses of simple filtering.

An important issue in signal and noise separation is the removal of high-amplitude noise before filtering or inversion, since high-amplitude noise corrupts the estimation of prediction-error filters and impairs least-squares inversion techniques. To detect automatically these high-amplitude noises, trace-to-trace predictions are examined for large residuals that correspond to bad samples. After the bad samples are eliminated, the inversions are arranged to predict the missing data simultaneously with the signal and noise separation. The missing data may be data that has been removed because of the high-amplitude noise removal, or it may be data missing because it was not recorded.

Two general forms of inversion are used. One form requires only a filter that describes the signal. This form of inversion is useful in removing noise that is unpredictable from trace to trace. The other form requires filters that describe both the signal and the noise. This second form allows high-amplitude and coherent noise to be well separated from the signal, but it often requires a more complicated weighting function to properly distribute the data between noise and signal.

These techniques are applied to synthetic and real seismic data to demonstrate the weaknesses and strengths of the various approaches.

Approved for publication:

By _____

For Department of Geophysics

Preface

Most of the figures in this thesis show the result of programs applied to seismic data. In the interest of providing research that is reproducible, a CD-ROM version of this thesis is available containing the input seismic data and the programs used to create most of the figures.

All of the figures in this thesis are marked with one of three labels, [R] [NR] or [CR]. These define whether the figure is reproducible from the information provided on the CD-ROM version of this thesis.

If a figure is marked with an [R] it is fully reproducible. All of the input data and source files required to create the figure are provided on the CD. The postscript file containing the figure can be removed and the cakefile rules provided on the CD will rebuild the figure on any of the supported computer architectures.

If a figure is marked with a [CR] it is conditionally reproducible. All the data and source files needed to recreate the figure are provided on the CD, but the figure will require a fairly long time to recreate.

If a figure is marked with an [NR] it is not reproducible. It cannot be recreated automatically from the data on the CD. In this thesis, a figure marked [NR] will generally be a hand drawn figure. In this case, if you remove the postscript file for that figure, there is no way to recreate it.

All of the figures have a button in the caption. For the reproducible and conditionally reproducible figures, clicking with the center mouse key on the button will bring up a panel that lets you interact with the figure. You can display the figure in a separate window or destroy and rebuild the figure. If the figure is conditionally reproducible you can view the warning message that tells you the conditions under which the figure can be rebuilt.

The advantages of making the research fully reproducible go beyond just the interest in confirming my results. It is my hope that the computer codes available on the CD-ROM

version of this thesis will be useful to those who face the problems addressed here.

To follow the conventions used by the journal *Geophysics*, in this thesis, vectors are in bold print such as $\mathbf{v}$, and matrices are in bold with a tilde underneath like $\underset{\sim}{\mathbf{A}}$.

For the convenience of the reader, a list of the commonly used symbols is given here.

$\dagger$ = indicates the transpose conjugate operation

$*$ = indicates convolution

δ_k = the delta function, or one when k is zero and zero otherwise

$\underset{\sim}{\mathbf{D}}$ = a matrix in which the data is organized so that matrix multiplication appears as a convolution to a filter vector on which $\underset{\sim}{\mathbf{D}}$

$\mathbf{d}$ = data vector

$\mathbf{e}$ = error vector

E = expectation

ϵ = scaling factor, generally for balancing parts of an inversion

$\underset{\sim}{\mathbf{F}}_s$ = multidimensional signal annihilation filter

$\underset{\sim}{\mathbf{F}}_n$ = multidimensional noise annihilation filter

$\mathbf{f}$ = a filter

$\underset{\sim}{\mathbf{I}}$ = the identity matrix, a square matrix with ones on the diagonal and zeros elsewhere

$\underset{\sim}{\mathbf{K}}$ = known data mask

$\mathbf{k}$ = known data

$\underset{\sim}{\mathbf{L}}$ = a lower triangular matrix

$\underset{\sim}{\mathbf{L}}\underset{\sim}{\mathbf{U}}$ decomposition = the decomposition of a matrix into upper and lower triangular matrices

$\underset{\sim}{\mathbf{M}}$ = missing data mask

$\mathbf{m}$ = missing data vector, or as in chapter 3 this indicates a model vector

$\underset{\sim}{\mathbf{N}}$ = noise annihilation filter

$\mathbf{n}$ = noise vector

$\varphi(x)$ = probability density as a function of x

p_{x_i} = probability function of x_i

$\mathbf{r}$ = a residual vector

SEP = Stanford Exploration Project

$\underset{\sim}{\mathbf{S}}$ = signal annihilation filter

$\mathbf{s}$ = signal vector

$\underset{\sim}{\mathbf{U}}$ = an upper triangular matrix

$\underset{\sim}{\mathbf{V}}$ = covariance matrix

ω = angular frequency

$\underset{\sim}{\mathbf{W}}$ = a weighting matrix

Acknowledgments

First, I would like to thank Jon Claerbout for allowing me to be a part of the Stanford Exploration Project. I must also thank my fellow students, present and past, who have made SEP such a pleasant place to work, especially those who made contributions to the software environment here at Stanford. Much work has gone into each of many useful tools built by Jon and his students. A special word of thanks is offered to the people with whom I have shared offices in the past few years and to the many people, both within and outside of the Stanford Exploration Project, with whom I've had many pleasant conversations. In particular, I would like to thank Bill Harlan for reviewing chapter 3 of this thesis.

Thanks are also due to the sponsors of the Stanford Exploration Project for their financial support. ARCO generously provided the three-dimensional dataset used for the random noise removal, Western Geophysical provided the shot gather used to demonstrate the coherent noise separation in chapter 10, and Guy Duncan of the University of Melbourne provided the shot gather used in the single trace noise separation in chapter 10. Oz Yilmaz and Dennis Cumro of Western Geophysical provided the data in chapter 8 as part of a collection of shot gathers from around the world.

I owe many debts to people who have provided encouragement and enlightenment in the past. Some of these people who have had the greatest impact are Frank Levin, Milton Dobrin, and Arthur Friedrich.

Finally, I must thank my wife Debbie for her support of my work and her tolerance of our move to Stanford.

Table of Contents

List of Figures

Chapter 1

Introduction

Separating noise from signal is an old problem in geophysics. Most geophysical work involves some separation of the desired information from the information that is not of interest, that is, noise. Many seismic data, especially those acquired on land, are seriously contaminated with noise that impedes interpretation and interferes with further processing and analysis.

Noise may be either random or coherent. Random noise, as it is considered here, is a noise that has no predictability from one sample to the next. Seismic data often has a background of this random noise. This random noise is recognized by its dissimilarity from trace to trace. Seismic signals, on the other hand, are recognized by their lateral continuity, and this continuity is used to distinguish events of interest from the background random noise. Much of this continuity results from the sedimentary character of the data being considered. Coherent noise may be considered as undesired signal. Examples of coherent noise are ground roll, near-surface scatterers, refracted arrivals, and out-of-plane reflectors. While coherent noise may often be signal in some other context, these noises interfere with the primary use of the data. Eliminating background noise from a seismic section makes reflections significantly easier to recognize, especially when noise is strong.

1.1 Background

In the past, single trace deconvolution and NMO stacking have been the most effective noise attenuation advances. Single trace deconvolution can be thought of as removing the reverberation of the reflection, a coherent noise, from the data. NMO stacking is

an effective noise attenuator of both coherent noise that does not follow the specified hyperbolic trajectory and of random noise. Even today, these two techniques provide the greatest improvement to our seismic records. Unfortunately, these techniques do not always provide sufficient noise attenuation.

When noise has properties that allow it to be distinguished from signal, it may be removed by taking advantage of these properties. Early noise-attenuation techniques were limited to the grouping of geophones to cancel coherent noise (Dobrin and Savit, 1988; Kanasewich, 1990; Stone, 1994), since sophisticated post-acquisition processing was unavailable. Later, along with digital data recording and processing, multichannel techniques such as velocity filters and F-K filtering (Yilmaz, 1987) were used to eliminate noise such as ground roll and air blasts.

For attenuating random noise, a number of techniques involving filtering or mixing adjacent traces were used before the introduction of the prediction technique of Canales (1984). Canales divided the two-dimensional filtering problem into many one-dimensional filtering problems in space, one for each frequency. Canales' idea was further developed by Gulunay (1986) and was referred to as FX-decon. These techniques have been very successful. They were relatively easy to use, reliable, and did not harm reflected events. The f-x prediction techniques of Canales and Gulunay may also be implemented as t-x prediction process, as is done in chapter 5. A technique that is somewhat similar to the t-x prediction process shown here was presented by Hornbostel (1991), although the emphasis in his work was handling time- and space-varying signals. In this thesis, time- and space-varying signals are treated by windowing the seismic data into smaller sections as discussed in Claerbout (1992b). Unfortunately, these prediction techniques may be only partially successful in the presence of strong noise. Furthermore, these techniques can generate spurious events in seismic data, as I will show later in this thesis.

Bad traces in prestack seismic data may be manually removed or edited automatically (Pokhriyal et al., 1991; Pokhriyal, 1993). A similar approach to automatically removing samples with high-amplitude noise will be presented later in this thesis. Sinusoidal noise may be separated from signal without filtering out the signal of the same frequency (Linville and Meek, 1992). Various specialized methods are available to remove coherent noise from prestack data. Along with velocity filtering and F-K filtering, multiple attenuation, stacking, deconvolution, and dip filtering are all efforts to separate noise from signal.

In this thesis, I assume that the signals of interest are linear; that is, signals are lines in 2-D data and planes in 3-D data. For nonlinear events, the data are subdivided, or windowed, into smaller sections where the events of interest are approximately linear. While the human eye recognizes the continuity of nonlinear events, the mathematical tools available to us work best on linear problems. Windowing extends the applicability of the techniques discussed here to data in areas of complex geology and to prestack data. Complex geology requires that the windows be small enough to make the geology look linear within the window, although this requirement may be relaxed somewhat because of the smoothing effect of diffractions. Prestack data generally appears as a series of hyperbolas, which will be more linear at the far offsets than at the near. Once again, windowing these hyperbolas generally will allow them to be treated as linear events within the window.

1.2 Outline of the thesis

This thesis covers two methods of separating noise and signal. The first method is removing noise by prediction filtering. Examples of this method are the traditional predictive deconvolution, Canales' and Gulunay's fx-decon techniques, and the t-x prediction techniques discussed in this thesis and in Hornbostel (1991). The second method is to separate the signal and noise using a least-squares inversion method to predict the signal or the noise.

Additionally, a preprocessing step of removing high-amplitude noise is presented to condition data in preparation for these two methods. Since high-amplitude noise corrupts the calculation of a prediction filter and can overwhelm the least-squares inversions used to predict the signal or noise, this process is an important step in signal and noise separation. As an extension to the signal and noise separation problem, missing data may be predicted with a modification of the inversion method. This prediction of missing data is useful, since high-amplitude noise, which would otherwise overwhelm the least-squares calculations, can be removed and later recovered by the inversion.

1.2.1 Automatic data editing for high-amplitude noise

Chapter 2 addresses the removal of high-amplitude noise. While this topic is somewhat different from the rest of the thesis, data editing is often a step needed before the other

processes can be attempted. Removing these high-amplitude samples is done by comparing a trace with its neighbors using small two-dimensional prediction-error filters. Each prediction is done against a single nearby trace. In the two-dimensional problem, two predictions are done, one for each of a trace's nearest neighbors. The best predictions are taken from these two predictions, and if the best prediction exceeds some value, the corresponding input sample is removed and marked as having been deleted. These muted samples may be later restored as discussed in chapter 8.

1.2.2 Noise removal by filtering

Filtering is a common and effective method of removing noise, especially when the noise has a spectrum (1-D, 2-D, or 3-D) that is sufficiently different from the signal. In the work here, filters are derived from the data to be used as predictors. Predictive deconvolution is an example of this, where the signal is what remains after the predictable part of the trace is removed by the prediction-error filter. In contrast, f-x and t-x prediction treats the predictable part of the data as the signal, and the unpredictable part is considered noise.

Chapter 5 will compare the results of f-x and t-x predictions and extend the two-dimensional results into three dimensions. One of the main results of chapter 5 is that f-x prediction can be considered equivalent to a t-x prediction with a very long filter length in time. This long filter length allows more random noise to remain in the signal and may generate spurious events far from the original reflections. Another important result is that three-dimensional filtering provides important advantages over two-dimensional filtering, especially in areas of complex geology.

The advantage of filtering is that it is simple and easy to understand. The disadvantages of filtering, as shown in chapter 7, are that the filter response of the noise is left in the signal, and because the calculation of the filter is corrupted by the noise, spurious events may be generated and signal amplitudes reduced. To reduce these undesired effects, the signal and noise separation problem is posed as an inversion in the second part of this thesis.

1.2.3 Noise removal by inversion

One problem with noise removal by filtering is that the response of the noise to the filter is left in the signal. As an example, consider a model of the recorded data as the sum of

signal and noise, or $\mathbf{d} = \mathbf{s} + \mathbf{n}$, where $\mathbf{d}$ is the recorded data, $\mathbf{s}$ is signal, and $\mathbf{n}$ is noise. If a signal-prediction filter $\underset{\sim}{\mathbf{S}}$ is calculated from the data $\mathbf{d}$ such that $\underset{\sim}{\mathbf{S}}\mathbf{d} \approx \mathbf{0}$, and the noise is assumed to be what is left after the filter is applied, $\mathbf{n} = \underset{\sim}{\mathbf{S}}\mathbf{d}$, is inconsistent with the original model of $\mathbf{d} = \mathbf{s} + \mathbf{n}$. Instead the output of the filter is $\underset{\sim}{\mathbf{S}}\mathbf{n}$, since $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{s} + \underset{\sim}{\mathbf{S}}\mathbf{n}$, assuming that $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$.

An alternative to accepting this filtering result is to set up the problem as a matrix inversion. If the signal is predicted by a filter $\underset{\sim}{\mathbf{S}}$, such that $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$, the noise is predicted by a filter $\underset{\sim}{\mathbf{N}}$, such that $\underset{\sim}{\mathbf{N}}\mathbf{n} \approx \mathbf{0}$, and the recorded data is a sum of the signal and noise, $\mathbf{d} = \mathbf{s} + \mathbf{n}$, then the signal may be predicted with a system of regressions such as

$$\begin{pmatrix} \mathbf{0} \\ \underset{\sim}{\mathbf{N}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s}. \tag{1.1}$$

In this expression, the assumption has been made that $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ have been previously calculated. Since $\mathbf{s}$ and $\mathbf{n}$ are not available before system (1.1) is solved, $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ must be approximated somehow from the available data. In most of the cases in this thesis, $\underset{\sim}{\mathbf{S}}$ can be calculated by assuming the noise is unpredictable laterally or that $\underset{\sim}{\mathbf{S}}$ may be calculated from some subset of the full dataset. $\underset{\sim}{\mathbf{N}}$ may be calculated from some subset of the full dataset where the noise is expected to dominate.

If $\underset{\sim}{\mathbf{N}}$ is difficult to estimate, or if the noise is unpredictable, an alternative to the previous system of regressions is

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \epsilon\underset{\sim}{\mathbf{S}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \mathbf{n}. \tag{1.2}$$

This system will be examined in greater detail in chapter 7. The calculated signal in this case is $\mathbf{s} = \mathbf{d} - \mathbf{n}$. Here the assumption is made that the actual noise is close to the noise predicted by the prediction filtering process.

1.2.4 Noise removal with missing data

A common problem in solving the previous systems is that high-amplitude noise overwhelms least-squares inversion techniques. Removing the worst of the noise before calculating a signal filter and before attempting to separate signal and noise can significantly improve the results. When doing the inversion after removing data, the samples that have been removed must be accounted for, otherwise the zeroed samples will be just another noise contaminating the process.

To allow for the zeroed samples, the previous inversions are recast to predict missing data while separating signal from the noise. This prediction of missing data is implemented by defining the data $\mathbf{d}$ as the sum of the known data $\mathbf{k}$ and the missing data $\mathbf{m}$, or $\mathbf{d} = \mathbf{k} + \mathbf{m}$. The missing data $\mathbf{m}$ is then calculated along with the signal $\mathbf{s}$ and noise $\mathbf{n}$. These substitutions change the previous systems to

$$\begin{pmatrix} \mathbf{0} \\ \tilde{\mathbf{N}}\tilde{\mathbf{K}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \tilde{\mathbf{S}} & \mathbf{0} \\ \tilde{\mathbf{N}} & -\tilde{\mathbf{N}}\tilde{\mathbf{M}} \end{pmatrix} \begin{pmatrix} \mathbf{s} \\ \mathbf{m} \end{pmatrix} \tag{1.3}$$

and

$$\begin{pmatrix} \tilde{\mathbf{S}}\tilde{\mathbf{K}}\mathbf{d} \\ \epsilon\tilde{\mathbf{S}}\tilde{\mathbf{K}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \tilde{\mathbf{S}} & -\tilde{\mathbf{S}}\tilde{\mathbf{M}} \\ \epsilon & -\epsilon\tilde{\mathbf{S}}\tilde{\mathbf{M}} \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ \mathbf{m} \end{pmatrix}, \tag{1.4}$$

where $\tilde{\mathbf{K}}\mathbf{d} = \mathbf{k}$ and $\tilde{\mathbf{M}}\mathbf{d} = \mathbf{m}$, $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{M}}$ being masks for the data describing the known and missing data positions such that $\tilde{\mathbf{K}} + \tilde{\mathbf{M}} = \tilde{\mathbf{I}}$, where $\tilde{\mathbf{I}}$ is the identity matrix.

An important additional advantage of this extension to the previous inversions is that data missing because of acquisition problems may also be estimated. Prediction filtering techniques have long been attempted in prestack data but have often failed because of missing data. Using the predictions from systems (1.3) and (1.4) allows these missing traces to be predicted while producing results that are not affected by missing traces that are otherwise treated as valid data.

1.2.5 Noise removal by characterizing both signal and noise

Noise often has a distinctive spectrum and amplitude. These can be used to characterize the noise in an inversion process. Chapters 9 and 10 show how inversion may be used to separate signal and noise using filters and amplitude characteristics of both the signal and noise. This can be especially useful when the noise is coherent and is predictable from trace to trace. One of the challenging issues is to avoid an overlap in the characterization of the noise and the signal, and if the overlap occurs, to control it.

Chapter 2

Data editing

This chapter addresses an issue that is somewhat different than the topics covered by the rest of this thesis. This issue is data editing, or the removal of bad data by muting, one of the simplest methods of removing noise. The emphasis in this chapter is eliminating isolated high-amplitude unpredictable events. These events are unpredictable in the sense that they leave high-amplitude residuals when predictable events are removed. This is in contrast to the low-amplitude residuals normally expected as prediction errors.

Data editing was one of the original methods of controlling noise in seismic data. When a seismic trace was dominated by noise, it was simply removed. For prestack data that would only be stacked, removing an offending trace would not significantly affect the stack as long as the stack fold was significant. Even then, only traces completely overwhelmed by high-amplitude noise needed attention, since moderately bad traces generally would not affect the stacked result.

In modern data processing, manual editing of traces is no longer practical since the volume of data is so large that a processor cannot examine all the data in a reasonable time. Automatic editing is especially important for three-dimensional surveys because of the huge data volumes involved. These large data volumes have led to other efforts to edit data automatically (Pokhriyal et al., 1991; Pokhriyal, 1993).

With modern acquisition systems, the dynamic range of the recorded data is much larger than that available with older systems. While this large dynamic range allows more accurate recording of the desired signals, higher amplitude noise is also allowed into the data. This noise may be caused either by recording unwanted signals, such as ground roll, or it may be caused by imperfections in the recording instruments. Modern data

processing techniques preserve these high amplitudes, where some older data processing methods would have limited the highest amplitudes to some "reasonable" limit.

Another problem is that data is no longer simply stacked; clean prestack data is needed for processes such as velocity determination, prestack migration, and AVO measurements. The processes discussed later in this thesis will also be impacted by high-amplitude noise in the data. High-amplitude noise is a special problem when least-squares methods are being used, since the square of a high-amplitude error is likely to overwhelm the smaller errors that are of more interest.

The goal of this chapter is to locate, eliminate, and mark the position of high-amplitude noise so that it will not contaminate the least-squares techniques presented in the following chapters. Marking the locations of bad data is done for two reasons. The first is to avoid mistaking the data that was zeroed for good data. The second reason is to allow restoration of the data that has been removed with an estimate derived from an inversion processes.

The unpredictable noise will be detected when the residual of a prediction-error filter exceeds a limit that is defined from the data. This technique will be demonstrated on real data with a variety of noises.

2.1 Data continuity assumptions

To eliminate noise, some description of the signal is needed. In this case, signal is assumed to be anything that is predictable from trace to trace, while noise is unpredictable between traces. While this predictability might be refined to specify as signal anything outside the evanescent zone (Claerbout, 1985), for the sake of simplicity I will stay with the assumption that signal is the only part of the data that is predictable between traces. Although later I will show an example where some coherent noise was removed, the process described in this chapter is not designed to attack coherent noises such as ground roll. The emphasis here is on removing isolated noise rather than coherent noise.

Another assumption made here is that good data may be predicted using filters. To allow a valid prediction filter to be computed, the number of samples that can be missing must be only a fraction of the original samples available, otherwise there will not be enough data to compute the filter reliably. A related assumption is that there will be enough good data to allow a valid prediction filter to be computed. If the data is completely dominated by noise, calculating a meaningful filter is difficult.

2.2 Algorithm

2.2.1 Editing in one-dimensional data

As an example of editing one-dimensional data, consider a time series such as the following:

$$1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 3 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1. \tag{2.1}$$

It is obvious that one sample does not follow the pattern of the majority of the samples. A prediction may be made on these samples by applying the filter

$$1 \quad -1 \tag{2.2}$$

to the data, giving a result of

$$0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 2 \quad -2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0. \tag{2.3}$$

While the exception is more visible, the residual is now spread over two samples. Applying the same filter in the reverse direction produces

$$0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad -2 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0. \tag{2.4}$$

Much the same residual has been obtained as in (2.3), but it has now been smeared in the other direction. To find the sample that has caused the residual, the maximum of the absolute values of corresponding samples from both filtered series can be taken. For the filtered series in (2.3) and (2.4), the result is

$$0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0. \tag{2.5}$$

The troublesome sample is now easy to distinguish. To detect it automatically, a quantitative criteria for the expected size of the residuals must be used. A number of criteria are available, including some multiple of the average, the median, or the RMS. For the cases likely to appear in seismic data, a multiple of the median of the residuals is likely to produce the most robust measure. This measure will be used as the threshold value indicating a noisy sample in the discussions that follow.

2.2.2 Editing in two-dimensional data

Two-dimensional data provides a more challenging problem, but also a more useful one. Consider the following as an example of two-dimensional data with noise:

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 99 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} . \tag{2.6}$$

Assume that the vertical axis of (2.6) corresponds to the time axis, and that the horizontal axis corresponds to the space axis. This dataset has a horizontal event plus a noise spike. An obvious method of removing the noise is to apply the one-dimensional filter (2.2) to either the rows or the columns of (2.6). Applying filter (2.2) to the rows would be making the assumption that reflection events are flat. This approach would be valid for the dataset in (2.6) since the signal is flat. Applying filter (2.2) to the columns, or along the time axis, would generally not be valid, since reflections are assumed to be unpredictable, and predictable events within a trace indicate undesirable noise.

For a dataset such as

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 99 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} , \tag{2.7}$$

two-dimensional filters are needed to predict the sloping event. One such filter is

$$\begin{matrix} 0 & -1 \\ 1 & 0 \\ 0 & 0 \end{matrix} . \tag{2.8}$$

In the case of real seismic data, the filters are computed from the data. While the filters could be calculated over several traces, I limit the prediction of a trace to include

only its immediate neighbors. These filters have the form

$$\begin{matrix} 0 & a_1 \\ 0 & a_2 \\ 1 & a_3 \\ 0 & a_4 \\ 0 & a_5 \end{matrix} \tag{2.9}$$

to predict a trace from the right, and the form

$$\begin{matrix} b_1 & 0 \\ b_2 & 0 \\ b_3 & 1 \\ b_4 & 0 \\ b_5 & 0 \end{matrix} \tag{2.10}$$

to predict a trace from the left. In these filters, the output point is under the 1 coefficient. The vertical axis of this filter is time, the bs cover the trace used for the prediction, and the column with the single 1 coefficient covers the trace being predicted. Both these filters are calculated as annihilation filters as described in chapter 3, so that if the trace is well predicted, the residual will be almost zero. A small residual indicates that the trace is dominated by signal.

Two filter calculations are done for each trace, one for each filter shown above. When these filters are applied, two sets of residuals are created. These two sets of residuals are combined into a single set of residuals by taking a sample-by-sample minimum of the absolute values of the two residuals. This single residual is then used as the diagnostic for noise. The single residual allows a sample to be predicted from either the right or from the left by taking the best of the two original predictions.

Once the set of minima of the absolute values of the residuals is created, the median of all the minima within a window is calculated. Although measures other than the median, such as the RMS of a window, could be used, the median is likely to provide a better diagnostic of a typical value within a window. Zero values are ignored when calculating the median, so that traces that were not present or muted on the input to the process do not contribute to the value of the median. Any value greater than w times the median value calculated is considered to be caused by high-amplitude noise. The value of w is determined by examining the data processed with a range of ws. In the examples shown

here, a w of five was used, although the process seems to be somewhat insensitive to the exact values of w, as can be seen with Figure 2.3 in the electronic version of this thesis. Samples in the original data are eliminated when the corresponding values in the single residual are greater than w times the median within a window. A flowchart describing the entire process is shown in figure 2.1.

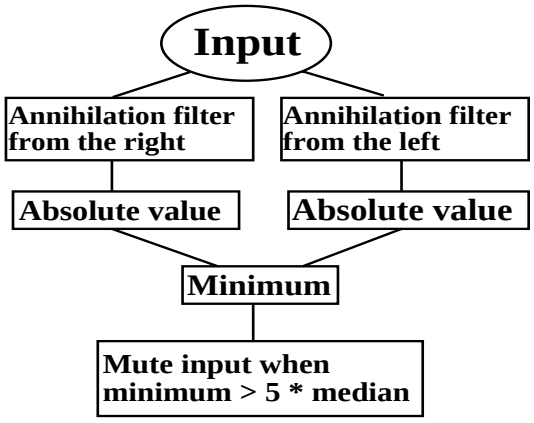

FIG. 2.1. A flowchart of the sample deletion process. dataedit-flowchart [NR]

2.3 Examples

Figure 2.2 shows a shot gather with some obvious bad traces. The data here and in the following plots have been scaled by time squared to show the signal better. Figures 2.3 and 2.4 shows the separation of the shot gather shown in Figure 2.2 into the rejected and the accepted samples with a w of five. In the electronic version of this thesis, pushing the button under Figure 2.3 shows a movie of the data in this figure with a range of values of w. For small values of w, for example 1 to 3, the result of the process changed quickly from one value of w to another. As w increased, the changes in the results for different values of w decreased. An example of a small portion of Figure 2.2 processed with values of w varying from 1 to 9 is shown in Figure 2.5.

The result of this process appears to do a good job of removing bad traces. Figure 2.3 shows very little coherent signal in the rejected samples. The little coherent noise left in Figure 2.3 appears on the near traces, which have anomalous amplitudes.

Figure 2.6 shows another shot gather with some bad traces and some coherent noise. Once again, the data here and in the following plots have been scaled by time squared.

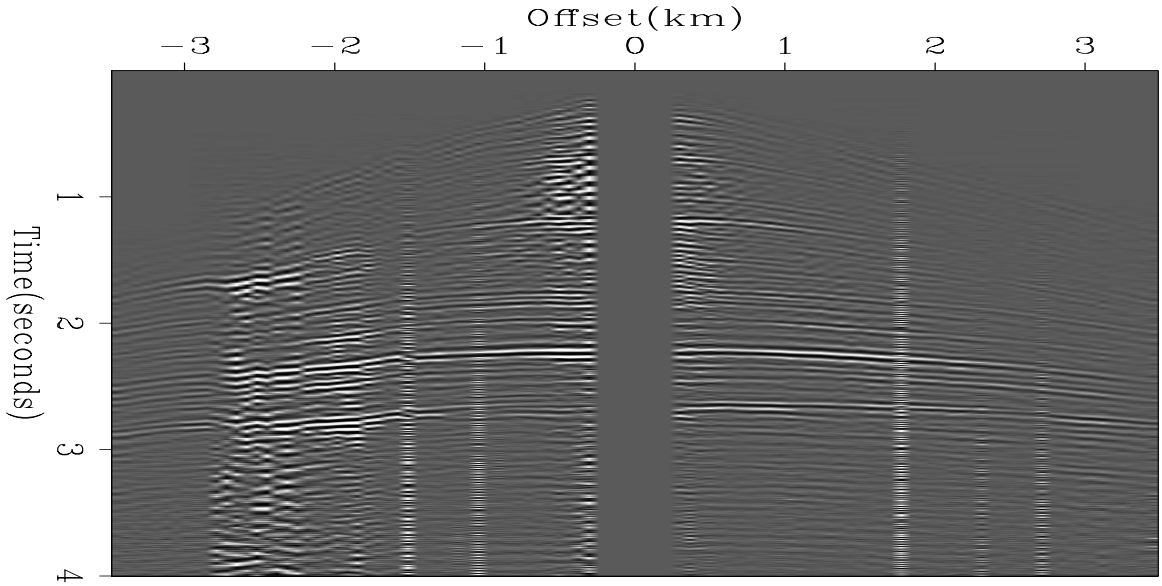

FIG. 2.2. The original data showing some bad traces and other noise. dataedit-original [R]

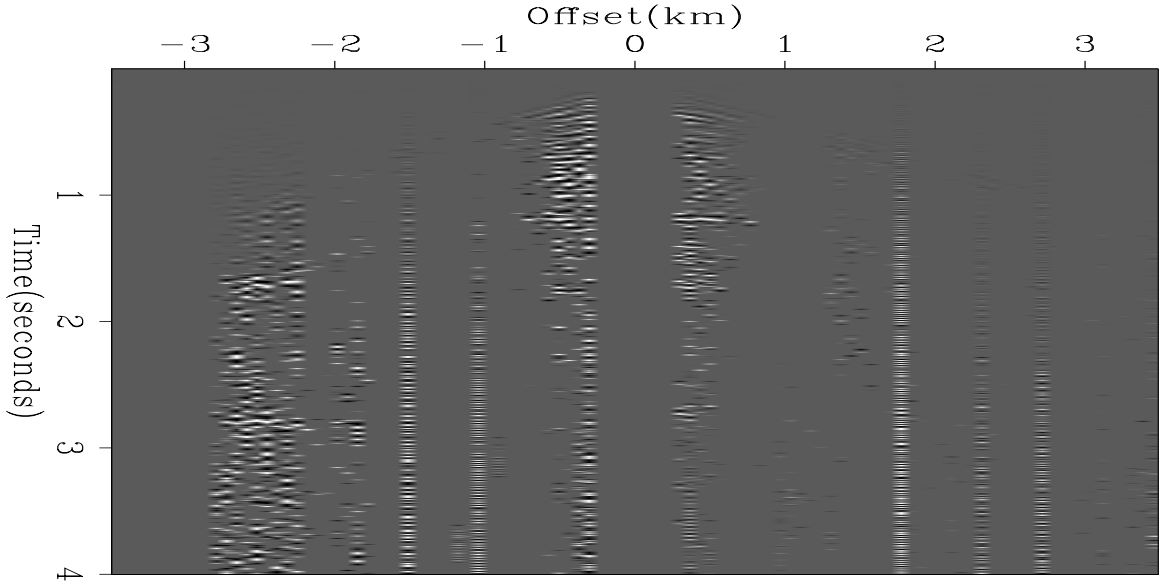

FIG. 2.3. The rejected samples from the data in the previous figure. The value of w was 5 for this result. Push the button to see a movie showing the rejected and accepted samples corresponding to various values of w. dataedit-mpatchn [R]

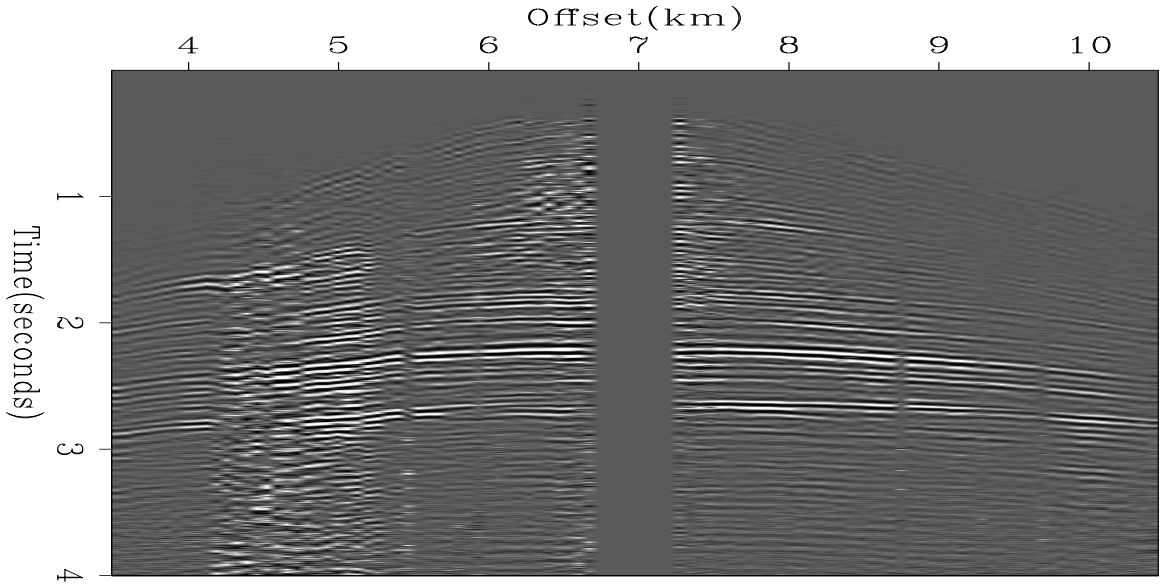

FIG. 2.4. The accepted samples from the data in the previous figure. The value of w was 5 for this result. dataedit-mpatchs [R]

Figures 2.7 and 2.8 show that much of the coherent noise between 2 and 3 seconds is removed. This effect is seen here because the filters used to make the trace-to-trace predictions are short, in this case five samples, and cannot predict much of the steeply dipping energy. While eliminating this particular event is desirable, care must be taken to make the filters long enough to predict all events that are to be preserved. Events such as the diffractions from complex events or overturned rays might not be predicted by very short filters, although generally these events will not be strong enough to be thrown out. In many cases, coherent noise is not as localized as it is in this example and so will not be eliminated.

One advantage of this technique over similar methods is that, since predictions are done from single neighboring traces, static shifts do not affect the predictions. Both examples shown here have traces with static shifts that are passed without problems.

2.4 Extensions

The methods used above appear to work reasonably well, but improvements may be made to take advantage of the type of data or the character of the noise. While these extensions are speculative, it is possible that some of these could produce significant improvements

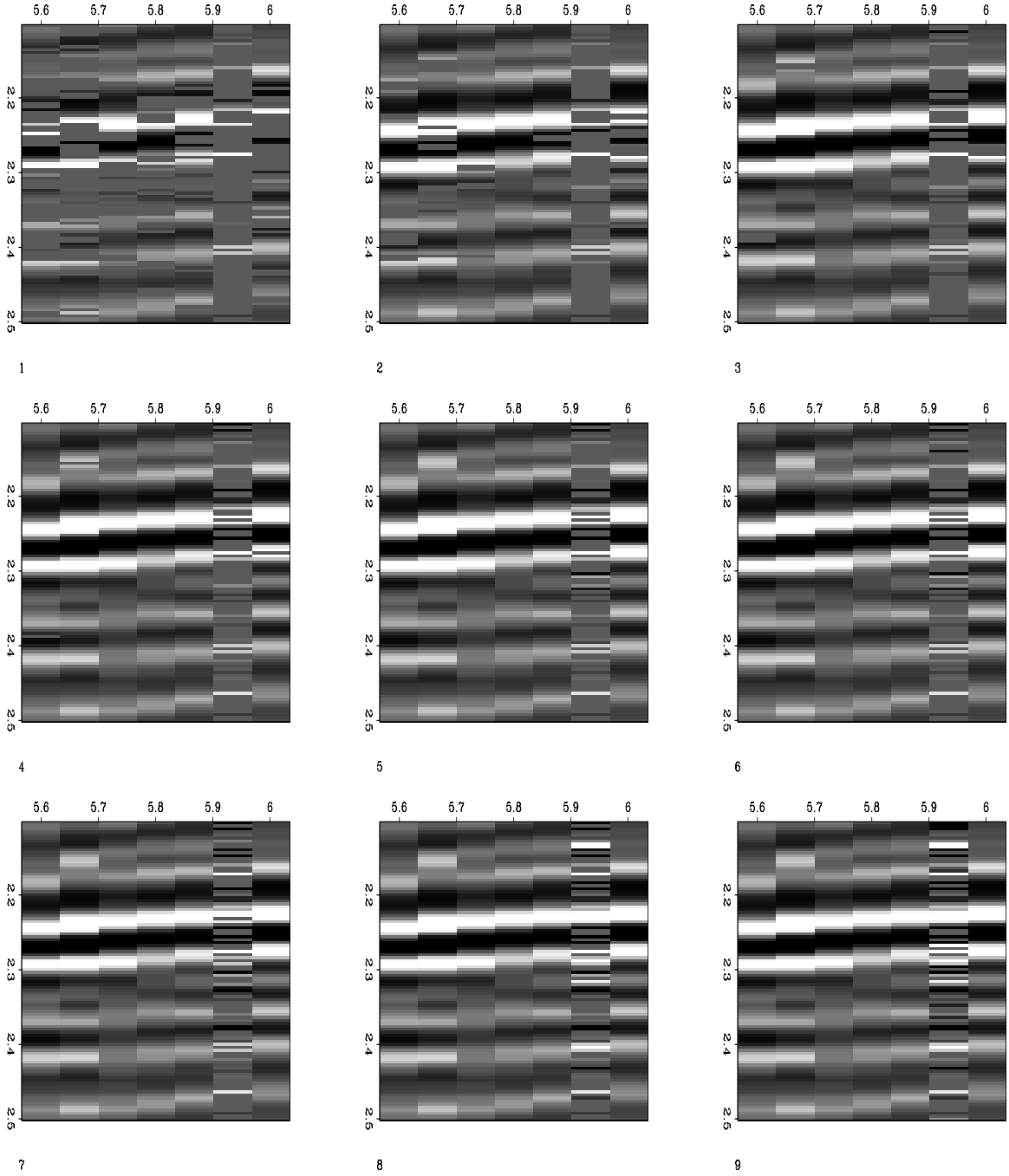

FIG. 2.5. A subsection of the seismic data processed with values of w varying from 1 to 9. dataedit-wplot [R]

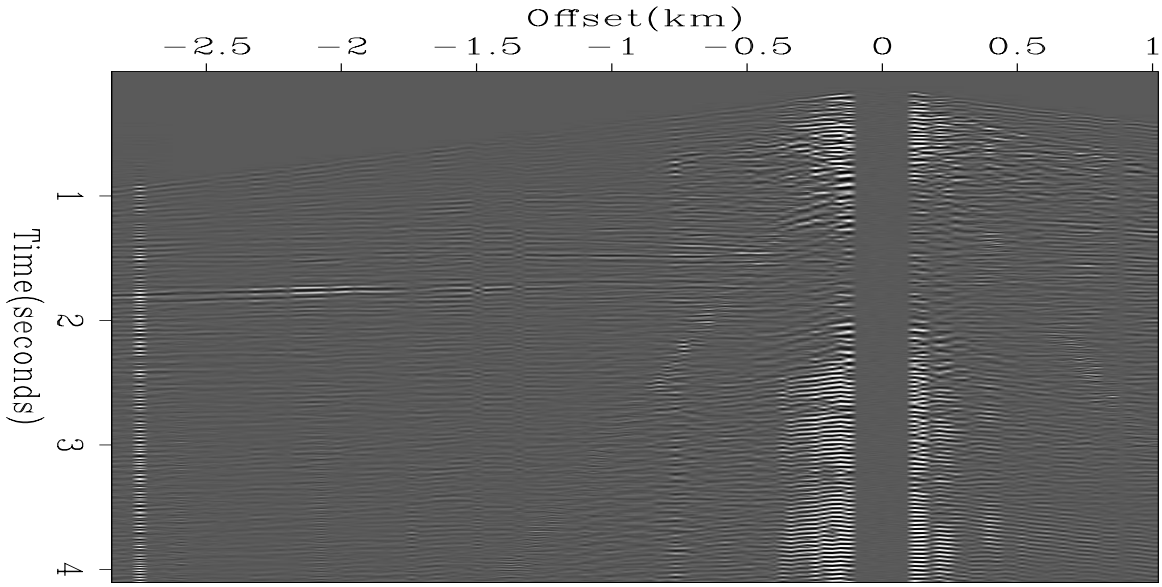

FIG. 2.6. The original data showing some bad traces and other noise. dataedit-original2 [R]

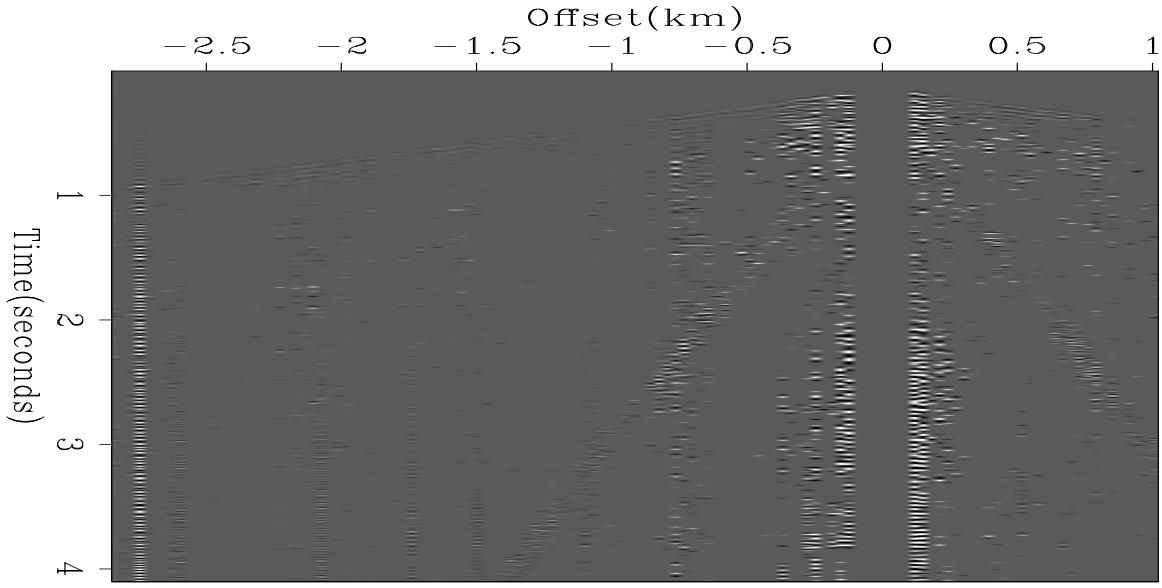

FIG. 2.7. The rejected samples from the data in the previous figure. The value of w was 5 for this result. In the electronic version of this document, push the button to see a movie showing the rejected and accepted samples corresponding to various values of w. dataedit-mpatch2n [R]

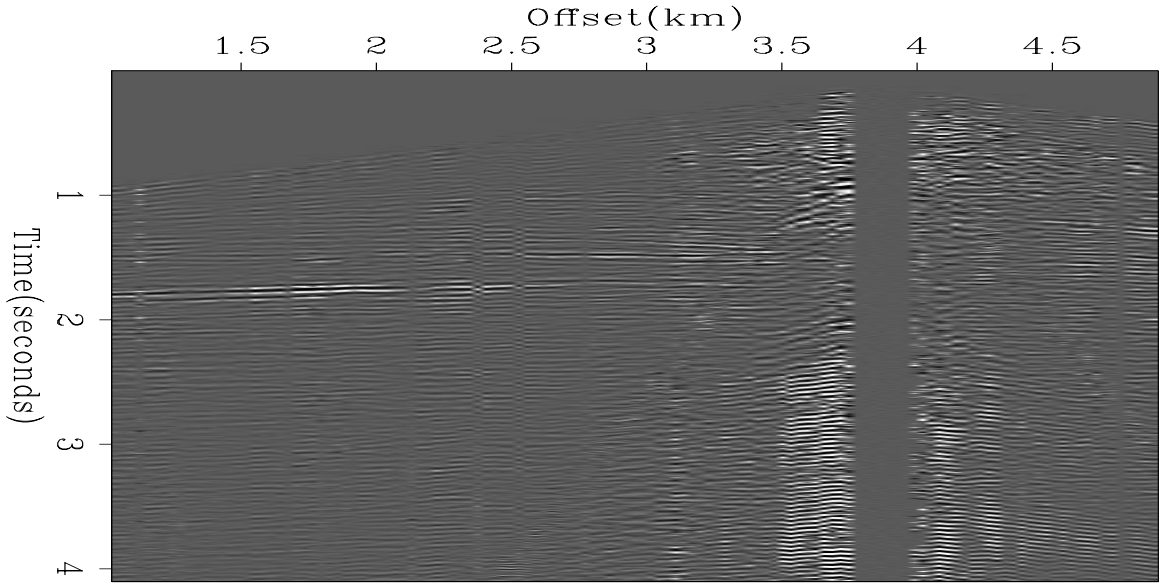

FIG. 2.8. The accepted samples from the data in the previous figure. The value of w was 5 for this result. dataedit-mpatch2s [R]

to the previous method.

2.4.1 Two-dimensional extensions

The algorithm described in the previous sections will work well in the cases where there are few bad traces. In the cases where many traces are bad, this method will fail. One example of a failure would be where a single good trace falls between two bad traces.

For a shot gather with many bad traces, comparing more than just adjacent traces may provide more information than comparing only neighboring traces. For example, comparing each trace to the four nearest traces would allow a single trace between two bad traces to be marked as good, provided at least one of the four nearest traces is good enough to be used to predict the trace being considered.

Although a good trace might be better predicted from more than one or two neighboring traces, as more adjacent traces are used in the comparisons, the possibility of producing spurious good predictions increases. Even for the examples presented in the previous section, some samples in the bad traces remain, not because the samples are good, but because the bad trace happened to have a fairly low amplitude where a low amplitude was predicted. While these small errors are less harmful than the high-amplitude

noise removed from the rest of the trace, removing them would be desirable. An obvious means of eliminating these spurious predictions would be to mute out a range of samples around any bad samples found. This approach is also appealing from a physical point of view, since any high-amplitude noise passing through the recording system is likely to have an lower-amplitude impulse response associated with it. Removing samples near high-amplitude noise could be considered as removing the impulse response of the noise.

Another approach to removing spurious predictions is to modify the comparisons of the residuals. For example, if four predictions are done, and three of the predictions indicate a sample is bad, the sample could be muted. In the previous work, any single good prediction is accepted. This idea of using the majority of the predictions to decide if a sample is eliminated will make the process more complicated and more time consuming than the previous methods, but may produce better editing in case where many of the traces are good. In the cases where many traces are bad, this method would require many more predictions than the technique demonstrated above.

Another alternative to the technique shown in the previous section would be extending the filters from

$$\begin{matrix} b_1 & 0 \\ b_2 & 0 \\ b_3 & 1 \\ b_4 & 0 \\ b_5 & 0 \end{matrix} \tag{2.11}$$

to include more traces to create filters such as

$$\begin{matrix} b_{1,1} & b_{1,2} & 0 \\ b_{2,1} & b_{2,2} & 0 \\ b_{3,1} & b_{3,2} & 1 \\ b_{4,1} & b_{4,2} & 0 \\ b_{5,1} & b_{5,2} & 0 \end{matrix} . \tag{2.12}$$

This filter allows two traces to be used to predict a trace. The neighboring trace that predicts the trace being considered best would automatically be weighted higher by the filter calculation. Although this would reduce the number of calculations in the median comparisons, this technique would have no obvious advantage in speed because the filter would be more expensive to calculate The cost of using this larger filter could be reduced by using the same filter over a range of traces. This extension then produces predictions

similar to the predictions of the noise to be done in chapter 5. A disadvantage of this approach would be that the traces must be fairly clean before a good prediction is made. The advantage of this approach is that it can be used to remove noise with lower amplitudes and to remove traces with static shifts.

2.4.2 Multi-dimensional extensions

So far, I have only discussed predictions done within a single shot gather. Applying the same process demonstrated previously to other gather types, especially common-midpoint gathers and common-receiver gathers, would be easy and would also allow bad shots to be deleted.

With only slight modifications, these techniques can be extended to multiple dimensions. Consider, for example, the gathers in Figure 2.9. The circled trace may be predicted from nearby traces in the shot, receiver, or midpoint gathers, but there is no reason that all the traces in the box around the circled trace cannot be used to predict that trace. This box may also be expanded to include as many traces as are needed.

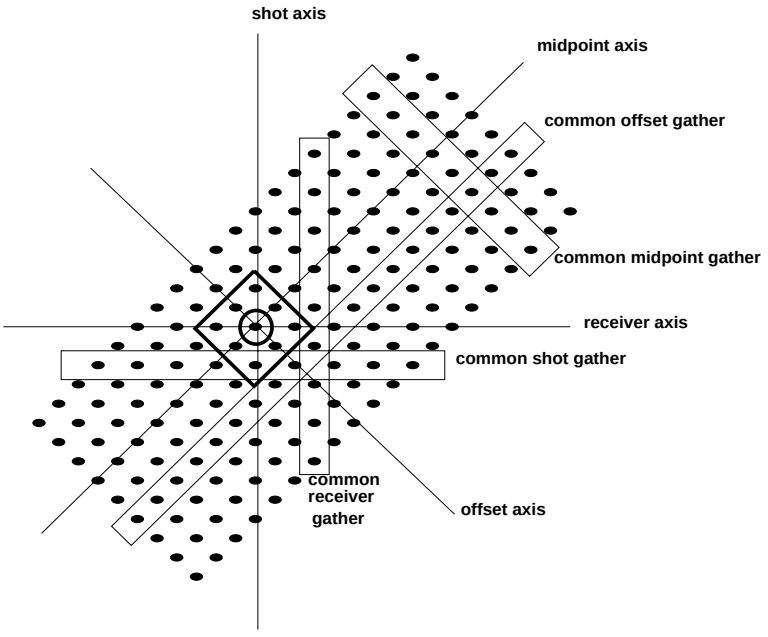

FIG. 2.9. Various gathers used in two-dimensional seismic data. dataedit-gathers [NR]

For three-dimensional acquisition, the predictions done from the nearby traces shown in Figure 2.9 can be extended to include traces in nearby crosslines. Eventually, all traces

within a given distance of a point can be used to predict a trace at that point.

The multiple dimensions available in prestack data allows other uses for these ideas. For example, the cable noise example discussed in Larner et al., (1983) shows the noise is unpredictable in CMP gathers, but is predictable in shot or receiver gathers. The unpredictability in the CMP gathers would allow the trace-to-trace predictions done there to remove this noise. A similar case would be cable noise such as that of crab pots caught in a marine cable. Such noise may be predictable over several traces in a given shot gather, since each hydrophone is being excited with the same noise source. Predictions in CMP gathers or receiver gathers would not see this noise as predictable. Bad or weak shots would also be unpredictable in both the CMP gathers and in the receiver gathers. Noise at a particular receiver position would also often be unpredictable in both the CMP gathers and in the shot gathers.

So far the extensions to multiple dimensions only involve multiple comparisons using two traces. This could be considered as two-dimensional prediction done in multiple directions. These predictions could also be extended to multiple traces within the two-dimensional predictions or be extended to multiple traces in multiple dimensions. This predictions could be seen as generalizing Claerbout's steep-dip deconvolution (Claerbout, 1993) to allow lateral predictions.

These multiple-dimensional predictions could be used in two manners. The first is to calculate a separate filter for every trace to be predicted. The second is to calculate one filter for all traces in a window, then predicting each traces from its neighbors with the filter. The first option is considerably more expensive than the second, since in the first case, one filter is calculated for every trace, while in the second case, a single filter is calculated for all the traces. The first option relaxes the assumptions of stationarity somewhat, since the filter can change from trace to trace, but the second option is likely to characterize the signal better because more traces go into the filter calculation. As in the two-dimensional case, it is not clear what advantages a single prediction with a multiple-dimensional filter has over multiple predictions with the simple two-trace filters. Nevertheless, using a single multi-dimensional filter may allow the prediction of low-amplitude noises since the filter constrains the signal better than the many two-dimensional filters.

2.4.3 Extensions for earthquake seismology

In cases where there is no natural ordering of the traces, such as the global arrays of seismometers used to study earthquakes, the arrangement of the traces is arbitrary, and comparing neighboring traces does not have much meaning. For these earthquake seismographs, a given trace could be compared to all the other traces in the array. Naturally, the prediction filters will be longer in this case than for the examples shown here because larger delays are expected between events, but the same basic algorithm would be used. Although many more comparisons would be made within this array than would be required in a typical exploration seismic gather, the earthquake seismograph array would generally have to be processed only once, while many gathers must be processed in a typical line gathered for exploration seismology.

2.5 Conclusions

Data editing to remove the high-amplitude noise is the first step in more refined signal and noise separation processes, as well as for other data processing. This step will often be required, since high-amplitude noise corrupts the least-squares calculations used in the more sensitive processes presented in later chapters. The technique shown here appears robust enough as a standard processing tool.

While removing high-amplitude noise is necessary for the techniques discussed in this thesis, there will be advantages in removing high-amplitude noise before other processes, even if the samples removed are not accounted for in the processes that follow. While zeroed samples may be considered as noise, this noise may be preferable to the higher-amplitude noise that sometimes occur in real data. One example of where this noise removal is desirable, would be standard single trace deconvolution in the presence of noise spikes. If these spikes have amplitudes that are high enough, the deconvolution operators will be ineffective, since the spikes give the data a white spectrum. Another example would be velocity analysis, where a high-amplitude spike in the input may generate curved noise trains in the velocity analysis.

While the method presented in this chapter appears useful enough, it is not the only approach that could be taken to remove high-amplitude noise. In particular, high-amplitude coherent noise may be simply muted out of a data record. The final editing method will depend on the nature of the noise and on the data. For small data volumes, manual muting

by the processor may be the most effective. The important issue is that high-amplitude noise is removed and that the edited data be marked so it will not be treated as good data and can be restored later. In chapter 8, an inversion to restore the edited data while separating noise and signal will be demonstrated.

Chapter 3

Inverse theory and statistical signal processing

In this chapter, I give some background to inversion techniques, especially as they relate to geophysical signal processing. Although the literature concerning inversion and signal processing is vast, some common references to inversion as applied to geophysical signal processing may be found in Webster (1978), Robinson and Treitel (1980), Kanasewich (1981), Tarantola (1987), Menke (1989), Claerbout (1992a), Parker (1994), and Claerbout (1995). Some more general references on inversion of linear systems may be found in Strang (1986), Strang (1988), and Golub and Van Loan (1989). Another goal of this chapter is to give the reader a feeling for the calculation and use of prediction-error filters both in standard filtering applications and in inversion for signal and noise separation, where the prediction-error filters, also referred to here as annihilation filters, characterize signal and noise.

3.1 Background and definitions

3.1.1 Inversion and statistics

From a purely mathematical viewpoint, inversion of a linear system is solving for a vector $\mathbf{m}$ when a matrix $\underset{\sim}{\mathbf{A}}$ and a vector $\mathbf{d}$ are supplied in an expression $\underset{\sim}{\mathbf{A}}\mathbf{m} = \mathbf{d}$. From an applied point of view, we are generally attempting to derive some information about a physical system when, from this system, a quantitative description of the system is built

by choosing a set of parameters of interest (Tarantola, 1987). This description of the physical system is referred to as the model $\mathbf{m}$ and is represented by a vector of numbers that parameterize the physical model. Also available is a set of measurements, or data $\mathbf{d}$, collected in the effort to derive some information about model $\mathbf{m}$. The relationship between the data $\mathbf{d}$ and the model $\mathbf{m}$ is assumed here to be linear and described by the matrix $\underset{\sim}{\mathbf{A}}$, where $\underset{\sim}{\mathbf{A}}\mathbf{m} = \mathbf{d}$. The problem of taking a model $\mathbf{m}$ and deriving the expected data $\mathbf{d}$ is referred to as the forward problem. This forward problem assumes that the relevant physics of the problem is described in the matrix $\underset{\sim}{\mathbf{A}}$. The inverse problem involves calculating the model $\mathbf{m}$ from a given set of data $\mathbf{d}$. As an example of the use of an inverse system in geophysics, a description of the earth is derived from measurements taken at the surface. The measurements from the surface correspond to $\mathbf{d}$, and the desired description of the earth corresponds to $\mathbf{m}$. Given $\underset{\sim}{\mathbf{A}}$ and $\mathbf{d}$, the description of the earth $\mathbf{m}$ is to be calculated by inversion.

The measured data $\mathbf{d}$ are likely to include some uncertainty, which is generally due to effects not included in the model. For example, when trying to derive an earth model using seismic data, the relevant physical laws to be taken into account would be those governing the propagation of seismic energy through the earth. Noise, or effects not included in the model, would be, for example, wind, local traffic, animals, Brownian motion, and so on. While most extraneous effects are unpredictable, or at least very difficult to predict, these noises can often be assumed to be random. Allowing for these unpredictable effects may then be left to statistical methods where the data $\mathbf{d}$ are considered realizations of random variables. Since the data are random variables, the estimates of the model are also random variables.

In the inversion of the expression $\underset{\sim}{\mathbf{A}}\mathbf{m} = \mathbf{d}$, the noise is considered undesirable and is eliminated as far as possible when calculating the desired $\mathbf{m}$. Much of the work here involves a different system in which the noise, or the unpredictable part of the data is of interest. This system is $\mathbf{r} = \mathbf{f} * \mathbf{d}$, where $\mathbf{d}$ is a recorded data series, $\mathbf{f}$ is a filter to be calculated, $*$ indicates convolution, and $\mathbf{r}$ is the unpredictable reflection series. This can be expressed in terms of the matrix equations $\mathbf{r} = \underset{\sim}{\mathbf{F}}\mathbf{d}$ or $\mathbf{r} = \underset{\sim}{\mathbf{D}}\mathbf{f}$, where $\underset{\sim}{\mathbf{F}}$ is the filter $\mathbf{f}$ expressed as a filter convolution matrix, and $\underset{\sim}{\mathbf{D}}$ is the data $\mathbf{d}$ expressed as a data convolution matrix. The next section addresses the assumptions made about $\mathbf{r}$ and $\mathbf{d}$.

3.1.2 Assumptions about the data and the errors

In this thesis, the emphasis will be on signal processing aspects of inversion. The problems to be dealt with will be time series or collections of time series. A single time series, referred to as a trace, generates a one-dimensional problem. A collection of traces generates multi-dimensional problems. The data recorded $\mathbf{d}$ will be analyzed to produce some information about the reflection series $\mathbf{r}$. The series $\mathbf{r}$ may also be considered to be the error in the expression $\mathbf{0} = \underset{\sim}{\mathbf{F}}\mathbf{d}$, where $\underset{\sim}{\mathbf{F}}$ is a filter that is designed to remove predictable information. Therefore, the reflection series $\mathbf{r}$ is assumed to be the unpredictable part of the series $\mathbf{d}$. The data is assumed to be stationary, Gaussian, and have a zero mean. Stationarity means that the the statistical character of the data does not change in time. This means that any statistical measure is expected to be unchanged if the trace is shifted. This assumption can be enforced by windowing the data so there are no large character changes within a window. The assumption of a Gaussian distribution might be more difficult to confirm but is still reasonable, since errors that are the result of some kind of summation tend to have Gaussian distributions. The zero mean is generally not a problem, since seismic data normally have had some kind of filter applied, and there is no reason to assume that the errors have a bias.

The reflection series, or error $\mathbf{r}$, produced from the filtering operation is also assumed to be stationary, Gaussian, and have a zero mean, but in addition, the samples of the errors are assumed to be independent from the other samples in $\mathbf{r}$. These assumptions may be better described by expectations.

The errors may be considered to be realizations of a random process drawn from a population, or ensemble. The expectation of a function of a random variable x is expressed as

$$E[f(x)] = \int_{-\infty}^{\infty} f(x)\varphi(x)dx \tag{3.1}$$

in the continuous case, where $\varphi(x)dx$ is the probability density (Korn and Korn, 1968). In the discrete case

$$E[f(x_i)] = \sum_i f(x_i)p_{x_i}, \tag{3.2}$$

where p_{x_i} is the probability function of x_i. The summation is over all x_i, and p_{x_i} sums to unity. For Gaussian distributions with zero mean, the expectation may considered to be an average as the number of samples goes to infinity, so that the expectation of the sample values is zero.

For the errors $\mathbf{r}$ and the data $\mathbf{d}$, the assumption of a zero mean is expressed as $E[\mathbf{r}] = 0$ and $E[\mathbf{d}] = 0$. The assumption that the samples of $\mathbf{r}$ are uncorrelated becomes $E[\mathbf{r}\mathbf{r}^\dagger] = \sigma_r^2 \underset{\sim}{\mathbf{I}}$, where $\underset{\sim}{\mathbf{I}}$ is the identity matrix, and $\dagger$ indicates the conjugate transpose, or adjoint. The dependence of the data values on each other is expressed as $E[\mathbf{d}\mathbf{d}^\dagger] = \underset{\sim}{\mathbf{V}}$, where $\underset{\sim}{\mathbf{V}}$ is the covariance matrix. The Gaussian distribution of $\mathbf{r}$ and $\mathbf{d}$ may be expressed as $p(\mathbf{r}) \propto e^{-\mathbf{r}^\dagger \sigma_r^{-2} \underset{\sim}{\mathbf{I}} \mathbf{r}/2}$ and $p(\mathbf{d}) \propto e^{-\mathbf{d}^\dagger \underset{\sim}{\mathbf{V}}^{-1} \mathbf{d}/2}$, where σ_r is a scalar and $\underset{\sim}{\mathbf{I}}$ is the identity matrix. Notice that these probability functions are distributions that satisfy the zero mean assumption. Also note that the independence of the samples in the errors is seen in the $\sigma_r^{-1} \underset{\sim}{\mathbf{I}}$ factor in $p(\mathbf{r})$, whereas the dependence of the data samples is seen in the inverse covariance matrix $\underset{\sim}{\mathbf{V}}^{-1}$ in $p(\mathbf{d})$.

3.1.3 Least-squares solutions to inverse problems

When solving an inverse problem, the effects not accounted for in the model may make the problem impossible to solve exactly. For example, if some component of $\mathbf{d}$ is in the left null space of $\underset{\sim}{\mathbf{A}}$, no model $\mathbf{m}$ can perfectly predict $\mathbf{d}$ (Strang, 1986). In such cases, a solution that is close to the actual model is the best solution that can be obtained. For least-squares methods, the sum of the squares of the errors between the data recorded and the data that the model should have produced is taken as the measure of closeness.

In the problems considered here, it is assumed that a large number of measurements have been made and that the solution to the inversion problem is either over-determined or mixed-determined (Menke, 1989). An over-determined problem is one in which all the components of the solution are over-determined, so that there will be some inconsistency, or error, in the data. A mixed-determined problem is one in which some of the components in the solution are over-determined, while other components are under-determined, so the problem has errors due to inconsistent measurements and model parameters that cannot be determined from the data. Since the problem is at least partially over-determined, there will generally be some error between the data calculated from a model $\mathbf{d}_{\text{calc}} = \underset{\sim}{\mathbf{A}}\mathbf{m}$ and the data recorded $\mathbf{d}$.

In the case of a system $\underset{\sim}{\mathbf{A}}\mathbf{m} = \mathbf{d}$, the least-squares solution is the one with the smallest sum of the squares of the difference between the actual data and the data derived from the model to be calculated. This difference to be minimized, the vector of errors $\mathbf{e}$, is defined as $\mathbf{e} = \underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{d}$, where $\mathbf{m}$ is the model and $\mathbf{d}$ is the data. The sum of the squares of the error is $\mathbf{e}^\dagger\mathbf{e}$, where $\dagger$ indicates the conjugate transpose, or adjoint. (For purely real

$\mathbf{e}$, $\dagger$ just indicates the transpose).

While $\underset{\sim}{\mathbf{A}}$ will later be considered as a matrix operation, for the moment, $\underset{\sim}{\mathbf{A}}$ may be considered to be any linear operator relating $\mathbf{m}$ to $\mathbf{d}$. To derive a model $\mathbf{m}$, the squared error $\mathbf{e}^\dagger\mathbf{e}$ is minimized. Expressed in terms of $\underset{\sim}{\mathbf{A}}$, $\mathbf{d}$, and $\mathbf{m}$, this becomes

$$\mathbf{e}^\dagger\mathbf{e} = (\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{d})^\dagger(\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{d}). \tag{3.3}$$

Expanding this produces

$$\mathbf{e}^\dagger\mathbf{e} = \mathbf{m}^\dagger\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} + \mathbf{d}^\dagger\mathbf{d} - \mathbf{d}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{m}^\dagger\underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}. \tag{3.4}$$

To minimize $\mathbf{e}^\dagger\mathbf{e}$ with respect to $\mathbf{m}$, we can find where the derivative is zero for either $\mathbf{m}^\dagger$ or $\mathbf{m}$, but $\mathbf{m}^\dagger$ is more convenient. The derivative of the previous expression then becomes

$$\frac{\partial}{\partial\mathbf{m}^\dagger}(\mathbf{e}^\dagger\mathbf{e}) = \frac{\partial}{\partial\mathbf{m}^\dagger}(\mathbf{m}^\dagger\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} + \mathbf{d}^\dagger\mathbf{d} - \mathbf{d}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{m}^\dagger\underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}) = \underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} - \underset{\sim}{\mathbf{A}}^\dagger\mathbf{d} = 0, \tag{3.5}$$

producing $\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}}\mathbf{m} = \underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}$. The value of $\mathbf{m}$ that minimizes $\mathbf{e}^\dagger\mathbf{e}$ is then $\mathbf{m} = (\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}})^{-1}\underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}$.

For the $\mathbf{r} = \mathbf{f} * \mathbf{d}$ system, there is an interesting connection between taking the minimum of the sum of the squares and the assumption that the errors are independent of each other. It can be shown that the two approaches are equivalent. The least-squares solution can be seen to be the solution that best fits the Gaussian distribution of the error $p(\mathbf{r}) \propto e^{-\mathbf{r}^\dagger\sigma_r^{-2}\underset{\sim}{\mathbf{I}}\mathbf{r}}$ seen above, where the samples of $\mathbf{r}$ are independent. Maximizing $p(\mathbf{r})$ is equivalent to minimizing $-\log(p(\mathbf{r}))$ or $\mathbf{r}^\dagger\sigma_r^{-2}\underset{\sim}{\mathbf{I}}\mathbf{r}$. This becomes the minimization of $\mathbf{r}^\dagger\mathbf{r}$, which is just the least-squares result for $\mathbf{r} = \underset{\sim}{\mathbf{F}}\mathbf{d}$. While I will continue with the least-squares approach, the independence of the errors will be emphasized more in section 3.1.5.

If $\underset{\sim}{\mathbf{A}}$ is a matrix and $\mathbf{d}$ and $\mathbf{m}$ are vectors, we get the minimum of $\mathbf{e}^\dagger\mathbf{e}$ by minimizing $(\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{d})^\dagger(\underset{\sim}{\mathbf{A}}\mathbf{m} - \mathbf{d})$. Once again this minimum occurs when $(\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}})\mathbf{m} = \underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}$, which is the expression for the least-squares inverse referred to as the normal equations (Strang, 1988). To find $\mathbf{m}$, the inverse of $(\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}})$ must be taken to get $\mathbf{m} = (\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}})^{-1}\underset{\sim}{\mathbf{A}}^\dagger\mathbf{d}$. This leaves the somewhat simpler problem of calculating $(\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}})^{-1}$.

3.1.4 Solving $(\mathbf{A}^\dagger\mathbf{A})^{-1}$ — methods and problems

In geophysical filtering applications, the matrix $\underset{\sim}{\mathbf{A}}^\dagger\underset{\sim}{\mathbf{A}}$ is often, although not always, Toeplitz. (A Toeplitz matrix is a matrix in which each row is a shifted version of the others and has constant diagonals). In general, if the type of filter application is transient convolution,

convolution with the data sequence padded with zeros, $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is Toeplitz, but if the filter application is internal (unpadded convolution) or truncated-transient (transient convolution truncated to the input length), $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is no longer in a Toeplitz form (Claerbout, 1995). When $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is Toeplitz, the matrix can be represented by a single row of the matrix, since a Toeplitz matrix is made up of a single row shifted so that a particular element is always on the diagonal. This makes the matrix easy to store in a computer memory, and easy to solve, since an efficient algorithm for solving Toeplitz systems, that of Levinson recursion, exists (Kanasewich, 1981).

In the cases where $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is not Toeplitz, it is still symmetric, and this symmetry can be useful in calculating $(\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}})^{-1}$. Another useful feature of $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is that it is generally positive definite. Although $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is guaranteed to only be positive semidefinite, in practice a small stabilizer $\epsilon\underset{\sim}{\mathbf{I}}$ is added to $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ to force it to be positive definite. The idea of adding $\epsilon\underset{\sim}{\mathbf{I}}$ can be viewed as forcing all the eigenvalues to be greater than ϵ and eliminating any zero eigenvalues. That $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is positive definite can be seen from the definition of a positive definite matrix, where $\underset{\sim}{\mathbf{B}}$ is positive definite if $\mathbf{x}^{\dagger}\underset{\sim}{\mathbf{B}}\mathbf{x}$ is always positive (Strang, 1986). If $\underset{\sim}{\mathbf{B}} = \underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ and $\underset{\sim}{\mathbf{A}}$ is full rank, then $\mathbf{x}^{\dagger}\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}\mathbf{x} = (\underset{\sim}{\mathbf{A}}\mathbf{x})^{\dagger}(\underset{\sim}{\mathbf{A}}\mathbf{x})$, which is always positive for non-zero $\mathbf{x}$. Since $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is positive definite, it can be solved by Cholesky factorization (Strang, 1988). Cholesky factorization allows a positive definite matrix $\underset{\sim}{\mathbf{B}}$ to be factored as $\underset{\sim}{\mathbf{B}} = \underset{\sim}{\mathbf{L}}\underset{\sim}{\mathbf{L}}^{\dagger}$, where $\underset{\sim}{\mathbf{L}}$ is a lower triangular matrix and $\underset{\sim}{\mathbf{L}}^{\dagger}$ is the corresponding upper triangular matrix. The cost of solving a system with Cholesky factorization is about half that of a general linear system solver, and the method is generally accurate (Makhoul (1975) in Childers (1978)). Efficient methods for Cholesky factorization to solve such systems are given in LINPACK (Dongarra et al., 1979) and in Tarantola (1987).

For a Toeplitz matrix, the storage requirements are small, since only one row of the matrix needs to be stored. In the more general case, where $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is symmetric positive definite, half the matrix needs to be stored for Cholesky factorization. When the inversion requires a fairly small matrix $\underset{\sim}{\mathbf{A}}$, Levinson recursion or Cholesky factorization methods are sufficient. Later in this thesis, larger problems than those that can be practically solved with these methods will be addressed, but it can be seen that even for filter calculations, these computations may become difficult because of their size. For example, when calculating a one-dimensional filter, the number of elements in the matrix $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is required to be the square of the number of elements in the filter. If the type of filter application is transient convolution, only one column of $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ needs to be stored, but if the filter

application is internal or truncated-transient convolution, $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is no longer Toeplitz and half of the elements in the matrix need to be stored. For a short one-dimensional filter, this is generally a reasonable requirement.

For two- or three-dimensional filters, the filters could be calculated by getting the inverse of $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$, but the size of this matrix will tend to be very large. Consider, for example, a small two-dimensional problem, where the filter is

$$\begin{pmatrix} f_{11} & f_{12} \\ f_{21} & f_{22} \end{pmatrix}, \tag{3.6}$$

and the data with which the filter will be convolved is

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix}. \tag{3.7}$$

By unwrapping and padding the data into a one-dimensional vector,

$$\begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{41} \\ 0 \\ 0 \\ x_{12} \\ x_{22} \\ \cdot \\ \cdot \\ \cdot \\ x_{43} \\ 0 \\ 0 \\ x_{14} \\ x_{24} \\ x_{34} \\ x_{44} \end{pmatrix}, \tag{3.8}$$

then unwrapping and padding the filter into a one-dimensional vector,

$$\begin{pmatrix} f_{11} \\ f_{21} \\ 0 \\ 0 \\ 0 \\ 0 \\ f_{12} \\ f_{22} \end{pmatrix}, \tag{3.9}$$

the two-dimensional convolution may be expressed as a one-dimensional convolution. To unwrap the filter into one dimension, the filter has been internally padded with zeros to fill out the first dimension of the data so the filtering does not overlap between columns. This padding will generally make the size of a two-dimensional filter much larger than a typical one-dimensional filter, although some space could be saved since the $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ matrix will be block diagonal. For three-dimensional filters, the first two dimensions must be filled out with zeros, forming an extremely long one-dimensional filter. In spite of the advantages of Levinson recursion, since the size of the matrix $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ depends on the square of the length of the one-dimensional filter formed by the padding, the $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ matrix will soon become so large that it becomes impractical to solve for multi-dimensional problems.

In addition to the large size of the $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ matrix, multi-dimensional filters tend to have many more coefficients than one-dimensional filters because of the higher number of dimensions involved. This increases the cost of solving for the filter by either the square or the cube of the number of coefficients, depending on the solution method. When internal convolutions with multi-dimensional filters are desired, $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ is no longer Toeplitz, and half of the huge $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ matrix must be stored and solved.

When inversions need to be done to predict full seismic datasets rather than just small filters, the number of elements to be calculated may be in the thousands or millions, making the number of elements in the matrix $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ in the millions or billions. Solving, or even storing, such large matrices will generally be impractical.

Later I will discuss the conjugate-gradient technique, which allows an iterative solution to these inversion problems. The conjugate-gradient technique has the advantage that the matrix $\underset{\sim}{\mathbf{A}}^{\dagger}\underset{\sim}{\mathbf{A}}$ does not need to be stored, and a good approximation to the answer may be had by limiting the number of iterations, thus reducing the cost. Also, the programming

of the problem tends to be simpler, since the conjugate-gradient technique requires just the coding of the convolution and its conjugate-transpose, or adjoint.

3.1.5 Prediction-error or annihilation filters

One of the most common geophysical applications of inversion is the calculation of prediction-error filters. A prediction-error $\mathbf{r}$ is defined as

$$r_j = \sum_{i=0}^{n} f_i d_{j-i}, \tag{3.10}$$

where $\mathbf{f}$ is the prediction-error filter of length $\mathbf{n}$, and $\mathbf{d}$ is an input data series. The filtering operation may also be expressed as $\mathbf{f} * \mathbf{d} = \mathbf{r}$, where $*$ indicates convolution. For the sake of the discussion here, $\mathbf{d}$ is an infinitely long time series. As in the previous discussion, the error $\mathbf{r}$ and the data $\mathbf{d}$ will be assumed to be stationary and have a Gaussian distribution with a zero mean. The error $\mathbf{r}$ will also be assumed to be uncorrelated so that $E[\mathbf{r}\mathbf{r}^\dagger] = \sigma_r^2 \underset{\sim}{\mathbf{I}}$.

Application of a prediction-error filter removes the predictable information from a dataset, leaving the unpredictable information, that is, the prediction error. A typical use of prediction-error filters is seen in the deconvolution problem, where the predictable parts of a seismic trace, such as the source wavelet and multiples, are removed, leaving the unpredictable reflections.

The condition that $\mathbf{r}$ contains no predictable information may be expressed in several ways. One method is by minimizing $\mathbf{r}^\dagger \mathbf{r}$, where $\mathbf{r}^\dagger$ is the conjugate transpose of $\mathbf{r}$, by calculating a filter that minimizes $(\mathbf{f} * \mathbf{d})^\dagger (\mathbf{f} * \mathbf{d})$. This minimization reduces $\mathbf{r}$ to have the least energy possible, where the smallest $\mathbf{r}$ is assumed to contain only unpredictable information.

Another equivalent expression of unpredictability is that the non-zero lags of the normalized autocorrelation are zero, or that

$$\frac{\sum_{i=-\infty}^{\infty} r_i r_{i+k}}{\sum_{i=-\infty}^{\infty} r_i r_i} = \delta_k, \tag{3.11}$$

where δ_k is one when k is zero and is zero otherwise. This approximation may also be expressed as $E[r_i r_{i+k}] = \sigma_r^2 \delta(k)$, where σ_r^2 is a scale factor that may be ignored. These two methods of expressing unpredictability are the basis for the sample calculations of the prediction-error presented in the next section.

The prediction-error filter may also be defined in the frequency domain using the condition that the expectation $E[r_i r_j] = 0$ for $i \neq j$. Transforming the autocorrelation into the frequency domain gives $E[\overline{r(\omega)} r(\omega)] = 1$, where $\overline{r(\omega)}$ is the complex conjugate of $r(\omega)$. Since $\mathbf{r}$ is the convolution of $\mathbf{f}$ and $\mathbf{d}$, $r(\omega) = f(\omega) d(\omega)$,

$$E[\overline{r(\omega)} r(\omega)] = 1 = E[\overline{f(\omega) d(\omega)} f(\omega) d(\omega)]. \tag{3.12}$$

Since the filter $\mathbf{f}$ is a linear operator, it can be taken outside of the expectation (Papoulis, 1984) to make the previous expression become

$$\frac{1}{\overline{f(\omega)} f(\omega)} = E[\overline{d(\omega)} d(\omega)]. \tag{3.13}$$

Thus, the power spectrum of $\mathbf{f}$ is the inverse of the power spectrum of $\mathbf{d}$. Although the phase of the data $\mathbf{d}$ is lost when creating the cross-correlation of $\mathbf{d}$ to get $\overline{d(\omega)} d(\omega)$, the phase of the filter is generally unimportant when the filter is being used as an annihilation filter in an inversion. For applications where a minimum phase filter is required, Kolmogoroff spectral factorization (Claerbout, 1992a) may be used.

Another way of expressing the unpredictability of $\mathbf{r}$ is $E[\mathbf{r}\mathbf{r}^\dagger] = \underset{\sim}{\mathbf{I}}$, that is, the expectation of $\mathbf{r}\mathbf{r}^\dagger$ is the identity matrix. This states that the expectation of the cross-terms of the errors are zero, that is, the errors are uncorrelated. This also states that the variances of the errors have equal weights. To make the matrices factor with an $\underset{\sim}{\mathbf{L}}\underset{\sim}{\mathbf{U}}$ decomposition (Strang, 1988), the expression $\mathbf{f} * \mathbf{d} = \mathbf{r}$ needs to be posed as a matrix operation $\underset{\sim}{\mathbf{F}}\mathbf{d} = \mathbf{r}$, with $\underset{\sim}{\mathbf{F}}$ as an upper triangular matrix. To do this, the indices of $\mathbf{d}$ and $\mathbf{r}$ are reversed from the usual order in their vector representations. A small example of $\underset{\sim}{\mathbf{F}}\mathbf{d} = \mathbf{r}$ is

$$\begin{pmatrix} f_1 & f_2 & f_3 & f_4 \\ 0 & f_1 & f_2 & f_3 \\ 0 & 0 & f_1 & f_2 \\ 0 & 0 & 0 & f_1 \end{pmatrix} \begin{pmatrix} d_4 \\ d_3 \\ d_2 \\ d_1 \end{pmatrix} = \begin{pmatrix} r_4 \\ r_3 \\ r_2 \\ r_1 \end{pmatrix} \tag{3.14}$$

Building one small realization of $\mathbf{r}\mathbf{r}^\dagger$ gives

$$\begin{pmatrix} r_4 r_4 & r_4 r_3 & r_4 r_2 & r_4 r_1 \\ r_3 r_4 & r_3 r_3 & r_3 r_2 & r_3 r_1 \\ r_2 r_4 & r_2 r_3 & r_2 r_2 & r_2 r_1 \\ r_1 r_4 & r_1 r_3 & r_1 r_2 & r_1 r_1 \end{pmatrix}. \tag{3.15}$$

To get an estimate of the expectation of this expression, it must be remembered that the data series is stationary, and the error will also be stationary. Only the differences in the indices are important, the locations are not. It can be seen that the elements of (3.15) are the elements of the autocorrelation at various lags. Using equation (3.11) makes the expectation of (3.15) become the identity matrix $\sigma_r \underset{\sim}{\mathbf{I}}$, where the σ_r may be dropped, since it is only a scale factor that may be incorporated into the filter or as a normalization of the autocorrelation.

Starting from the expression $E[\mathbf{r}\mathbf{r}^\dagger] = \underset{\sim}{\mathbf{I}}$ and substituting $\underset{\sim}{\mathbf{F}}\mathbf{d}$ for $\mathbf{r}$ gives

$$E[(\underset{\sim}{\mathbf{F}}\mathbf{d})(\underset{\sim}{\mathbf{F}}\mathbf{d})^\dagger] = \underset{\sim}{\mathbf{I}} \tag{3.16}$$

or

$$E[\underset{\sim}{\mathbf{F}}\mathbf{d}\mathbf{d}^\dagger\underset{\sim}{\mathbf{F}}^\dagger] = \underset{\sim}{\mathbf{I}}. \tag{3.17}$$

Once again, since $\underset{\sim}{\mathbf{F}}$ and $\underset{\sim}{\mathbf{F}}^\dagger$ are linear operators, they can be taken outside of the expectation (Papoulis, 1984)

$$\underset{\sim}{\mathbf{F}}E[\mathbf{d}\mathbf{d}^\dagger]\underset{\sim}{\mathbf{F}}^\dagger = \underset{\sim}{\mathbf{I}}. \tag{3.18}$$

Moving the $\underset{\sim}{\mathbf{F}}$s to the right-hand side gives

$$E[\mathbf{d}\mathbf{d}^\dagger] = \underset{\sim}{\mathbf{F}}^{-1}(\underset{\sim}{\mathbf{F}}^\dagger)^{-1} = (\underset{\sim}{\mathbf{F}}^\dagger\underset{\sim}{\mathbf{F}})^{-1}. \tag{3.19}$$

Expanding one realization of a small example of $E[\mathbf{d}\mathbf{d}^\dagger]$ gives

$$\begin{pmatrix} d_4d_4 & d_4d_3 & d_4d_2 & d_4d_1 \\ d_3d_4 & d_3d_3 & d_3d_2 & d_3d_1 \\ d_2d_4 & d_2d_3 & d_2d_2 & d_2d_1 \\ d_1d_4 & d_1d_3 & d_1d_2 & d_1d_1 \end{pmatrix}. \tag{3.20}$$

Once again, to get an estimate of the expectation of this expression, it should be remembered that $\mathbf{d}$ is stationary, and only the differences in the indices are important. It can then be seen that $E[d_id_{i-j}]$ are elements of the autocorrelation, and $E[\mathbf{d}\mathbf{d}^\dagger]$ is the autocorrelation matrix of $\mathbf{d}$. Setting $\underset{\sim}{\mathbf{A}} = E[\mathbf{d}\mathbf{d}^\dagger]$ gives

$$\underset{\sim}{\mathbf{F}}^\dagger\underset{\sim}{\mathbf{F}} = \underset{\sim}{\mathbf{A}}^{-1}. \tag{3.21}$$

Generally, $E[\mathbf{d}\mathbf{d}^\dagger]$ will be invertible and positive definite for real data. There are some special cases where $E[\mathbf{d}\mathbf{d}^\dagger]$ is not invertible and positive definite, for example, when $\mathbf{d}$

contains a single sine wave. To avoid these problems, the stabilizer $\epsilon\underset{\sim}{\mathbf{I}}$ is often added to the autocorrelation matrix, where ϵ is a small number and $\underset{\sim}{\mathbf{I}}$ is the identity matrix. In the geophysical industry, this is referred to as adding white noise, or whitening, since adding $\epsilon\underset{\sim}{\mathbf{I}}$ to the autocorrelation matrix is equivalent to adding noise to the data $\mathbf{d}$.

The matrix $\underset{\sim}{\mathbf{F}}$ may be obtained from the matrix $\underset{\sim}{\mathbf{F}}^\dagger\underset{\sim}{\mathbf{F}}$ by Cholesky factorization (Strang, 1988), since $\underset{\sim}{\mathbf{F}}^\dagger\underset{\sim}{\mathbf{F}}$ is symmetric positive definite. Cholesky factorization factors a matrix into $\underset{\sim}{\mathbf{L}}\underset{\sim}{\mathbf{L}}^\dagger$, where $\underset{\sim}{\mathbf{L}}$ looks like

$$\begin{pmatrix} l_{1,1} & 0 & 0 & 0 \\ l_{2,1} & l_{1,2} & 0 & 0 \\ l_{3,1} & l_{2,2} & l_{1,3} & 0 \\ l_{4,1} & l_{3,2} & l_{2,3} & l_{1,4} \end{pmatrix}. \tag{3.22}$$

The matrix $\underset{\sim}{\mathbf{F}}$ obtained from this factorization will be upper triangular, as seen in (3.14), so the filter is seen to predict a given sample of $\mathbf{d}$ from the past samples, and the maximum filter length is the length of the window. This matrix could be considered as four filters of increasing length. The longest filter, assuming it is the most effective filter, could be taken as the prediction-error filter.

Another way of looking at this definition of a prediction-error filter is to consider the filter as a set of weights $\underset{\sim}{\mathbf{W}}$ producing a weighted least-squares solution. Following Strang (1986), the weighted error $\hat{\mathbf{e}}$ is $\underset{\sim}{\mathbf{W}}\mathbf{e}$, where $\underset{\sim}{\mathbf{W}}$ is to be determined, and $\mathbf{e}$ is the error of the original system. The best $\hat{\mathbf{e}}$ will make

$$E[\hat{\mathbf{e}}\hat{\mathbf{e}}^\dagger] = \underset{\sim}{\mathbf{I}}. \tag{3.23}$$

Since $\hat{\mathbf{e}} = \underset{\sim}{\mathbf{W}}\mathbf{e}$,

$$E[\underset{\sim}{\mathbf{W}}\mathbf{e}(\underset{\sim}{\mathbf{W}}\mathbf{e})^\dagger] = \underset{\sim}{\mathbf{W}}E[\mathbf{e}\mathbf{e}^\dagger]\underset{\sim}{\mathbf{W}}^\dagger, \tag{3.24}$$

where $E[\mathbf{e}\mathbf{e}^\dagger]$ is the covariance matrix of $\mathbf{e}$. Strang, quoting Gauss, says the best $\underset{\sim}{\mathbf{W}}^\dagger\underset{\sim}{\mathbf{W}}$ is the inverse of the covariance matrix of $\mathbf{e}$. Setting $\underset{\sim}{\mathbf{F}} = \underset{\sim}{\mathbf{W}}$ and $\mathbf{d} = \mathbf{e}$ makes the weight $\underset{\sim}{\mathbf{W}}$ become the prediction-error filter seen in equation (3.19).

While I've neglected a number of issues, such as the invertability of $\mathbf{d}\mathbf{d}^\dagger$, the finite length of $\mathbf{d}$, and the quality of the estimations of the expectations, in practice the explicit solution to (3.14) will not be used to calculate a prediction-error filter. Most practical prediction-error filters will be calculated using other, more efficient, methods. A traditional method for calculating a short prediction-error filter using Levinson recursion will be shown

in the next section. In this thesis, most of the filters are calculated using a conjugate-gradient technique such as that shown in Claerbout (1992a).

Most prediction-error filters are used as simple filters, and the desired output is just the error $\mathbf{r}$ from the application of the filter $\underset{\sim}{\mathbf{F}}\mathbf{d} = \mathbf{r}$. Another use for these prediction-error filters is to describe some class of information in an inversion, such as signal or noise. In this case, these filters are better described as annihilation filters, since the inversion depends on the condition that the filter applied to some data annihilates, or zeros, a particular class of information to a good approximation. For example, a signal $\mathbf{s}$ may be characterized by a signal annihilation filter expressed as a matrix $\underset{\sim}{\mathbf{S}}$, so that $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$ which may be expressed as $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{r}$, where $\mathbf{r}$ is small compared to $\mathbf{s}$. A noise $\mathbf{n}$ may be characterized by a noise annihilation filter $\underset{\sim}{\mathbf{N}}$, so that $\underset{\sim}{\mathbf{N}}\mathbf{n} \approx \mathbf{0}$. Examples of annihilation filters used to characterize signal and noise will be shown in chapters 7, 8, 9, and 10.

In this thesis, prediction-error filters will be referred to as annihilation filters when used in an inversion context. While prediction-error filters and annihilation filters are used in different manners, they are calculated in the same way. In spite of the similarities in calculating the filters involved, the use of these filters in simple filtering and in inversion is quite different. Simple filtering, whether one-, two-, or three-dimensional, involves samples that are relatively close to the output point and makes some simplifying assumptions. An important assumption is that the prediction error is not affected by the application of the filter. Inversion requires that the filters describe the data, and the characterization of the data is less local than it is with simple filtering. The assumption that the prediction error is not affected by the filter can be relaxed in inversion, a topic to be further considered in chapters 4 and 7.

3.1.6 An example of calculating a prediction-error filter

The problem of calculating a prediction-error filter $\mathbf{f}$ can be set up by describing the convolution process as a matrix multiplication $\underset{\sim}{\mathbf{A}}\mathbf{f}$. The matrix $\underset{\sim}{\mathbf{A}}$ is made up of shifted versions of the signal $\mathbf{x}$ to be multiplied by the filter vector $\mathbf{f}$ as shown below. If $\mathbf{x}$ is perfectly predictable, a filter $\mathbf{f}$ will be calculated to give a result of zero, $\underset{\sim}{\mathbf{A}}\mathbf{f} = \mathbf{0}$. With most real data, $\mathbf{x}$ will not be perfectly predictable, and the result of $\underset{\sim}{\mathbf{A}}\mathbf{f}$ will not be zero, but $\underset{\sim}{\mathbf{A}}\mathbf{f} = \mathbf{r}$, where $\mathbf{r}$ is the error or unpredictable part. This error is minimized to get the desired $\mathbf{f}$. The idea of an imperfect prediction may also be expressed as a system of regressions, where $\underset{\sim}{\mathbf{A}}\mathbf{f} \approx \mathbf{0}$.

Here, I will show the traditional method of setting up a one-dimensional prediction-error filter calculation problem. The linear system $\underset{\sim}{\mathbf{A}}\mathbf{f} \approx \mathbf{0}$, expanded to show its elements, is

$$\begin{pmatrix} x_1 & 0 & 0 & 0 \\ x_2 & x_1 & 0 & 0 \\ x_3 & x_2 & x_1 & 0 \\ x_4 & x_3 & x_2 & x_1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ x_n & x_{n-1} & x_{n-2} & x_{n-3} \\ 0 & x_n & x_{n-1} & x_{n-2} \\ 0 & 0 & x_n & x_{n-1} \\ 0 & 0 & 0 & x_n \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \approx \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} . \tag{3.25}$$

Notice how the matrix $\underset{\sim}{\mathbf{A}}$ is made up of the vector $\mathbf{x}$ shifted against the filter to produce the convolution of $\mathbf{x}$ and $\mathbf{f}$. At least one element of $\mathbf{f}$ is constrained to be non-zero to prevent the trivial solution where all elements of $\mathbf{f}$ are zero. In this case, f_1 is constrained to be one so we can modify the equation above to move the constrained portion to the right-hand side to get

$$\begin{pmatrix} 0 & 0 & 0 \\ x_1 & 0 & 0 \\ x_2 & x_1 & 0 \\ x_3 & x_2 & x_1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ x_{n-1} & x_{n-2} & x_{n-3} \\ x_n & x_{n-1} & x_{n-2} \\ 0 & x_n & x_{n-1} \\ 0 & 0 & x_n \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \end{pmatrix} \approx \begin{pmatrix} -x_1 \\ -x_2 \\ -x_3 \\ -x_4 \\ \cdot \\ \cdot \\ \cdot \\ -x_n \\ 0 \\ 0 \\ 0 \end{pmatrix} . \tag{3.26}$$

The new matrix without the first row will be referred to as $\underset{\sim}{\mathbf{B}}$, and the set of filter coefficients without f_1 will be referred to as $\mathbf{f}'$. The system to be solved is then $\underset{\sim}{\mathbf{B}}\mathbf{f}' \approx -\mathbf{x}$. Using the normal equations, the solution for $\mathbf{f}'$ becomes $\mathbf{f}' = (\underset{\sim}{\mathbf{B}}^{\dagger}\underset{\sim}{\mathbf{B}})^{-1}\underset{\sim}{\mathbf{B}}^{\dagger}(-\mathbf{x})$.

$\underset{\sim}{\mathbf{B}}^\dagger\underset{\sim}{\mathbf{B}}$ is the autocorrelation matrix, which is also referred to as the inverse covariance matrix. The reason for calling $\underset{\sim}{\mathbf{B}}^\dagger\underset{\sim}{\mathbf{B}}$ an autocorrelation matrix is obvious, since the rows of the matrix are the shifted autocorrelation of $\mathbf{x}$. The description of $(\underset{\sim}{\mathbf{B}}^\dagger\underset{\sim}{\mathbf{B}})^{-1}$ as a covariance matrix comes from the idea that $\mathbf{x}$ is a function of random variables. The relationship between the values of $\mathbf{x}$ at various lags are described by the expectation E of the dependence of the values of $\mathbf{x}$ with different delays. This is expressed as $E(x_i x_j)$, where i and j are indices into the series $\mathbf{x}$. If the expectation $E(x_i x_j)$ is zero when $i \neq j$, the spectrum of $\mathbf{x}$ would be white, and the sample values of $\mathbf{x}$ would be unrelated to each other. The autocorrelation would be zero except at zero lag.

The solution for $\mathbf{f}$ requires storing only the autocorrelation of $\mathbf{x}$ and can be solved quickly and efficiently using Levinson recursion. Once again, the efficiency of this technique depends on $\underset{\sim}{\mathbf{B}}^\dagger\underset{\sim}{\mathbf{B}}$ being Toeplitz, which in turn depends on the type of convolution being transient, rather than internal or truncated-transient. When $\underset{\sim}{\mathbf{B}}^\dagger\underset{\sim}{\mathbf{B}}$ is not Toeplitz, it can still be solved using techniques such as Cholesky factorization, but at a higher cost of storage and calculation.

3.1.7 Prediction-error filtering in the frequency domain

If the convolution $\mathbf{f} * \mathbf{d} = \mathbf{r}$ is expressed in matrix form as $\underset{\sim}{\mathbf{D}}\mathbf{f} = \mathbf{r}$, where $\underset{\sim}{\mathbf{D}}$ is the convolution matrix of $\mathbf{d}$, the filter $\mathbf{f}$ can be solved for to get the least-squares minimum of $\mathbf{r}$. The normal equations expression for the least-squares inverse is $(\underset{\sim}{\mathbf{D}}^\dagger\underset{\sim}{\mathbf{D}})\mathbf{f} = \underset{\sim}{\mathbf{D}}^\dagger\mathbf{r}$, or $\mathbf{f} = (\underset{\sim}{\mathbf{D}}^\dagger\underset{\sim}{\mathbf{D}})^{-1}\underset{\sim}{\mathbf{D}}^\dagger\mathbf{r}$. This expression for $\mathbf{f}$ may be decomposed into simpler expressions in the frequency domain since $\mathbf{f} * \mathbf{d} = \mathbf{r}$ may be expressed as $f(\omega)d(\omega) = r(\omega)$. The expression $\mathbf{f} = (\underset{\sim}{\mathbf{D}}^\dagger\underset{\sim}{\mathbf{D}})^{-1}\underset{\sim}{\mathbf{D}}^\dagger\mathbf{r}$ may be transformed into the frequency domain as $f(\omega) = (\overline{d(\omega)}d(\omega))^{-1}\overline{d(\omega)}r(\omega)$ or $f(\omega) = \overline{d(\omega)}r(\omega)/(\overline{d(\omega)}d(\omega))$. (Here $d(\omega)$ indicates a component of the Fourier transform of the data, and $\overline{d(\omega)}$ indicates the complex conjugate of $d(\omega)$). Canceling out $\overline{d(\omega)}$ gives $f(\omega) = r(\omega)/d(\omega)$. Thus in the frequency domain, where filtering is described as a multiplication such as $f(\omega)d(\omega) = r(\omega)$, inversion is simply division, or $f(\omega) = r(\omega)/d(\omega)$. The values of $f(\omega)$, $d(\omega)$, and $r(\omega)$ are scalars (although they are complex numbers).

The $\overline{d(\omega)}d(\omega)$ term in the denominator is the Fourier transform of the autocorrelation of $\mathbf{d}$. If $\mathbf{d}^\dagger\mathbf{d}$ is the identity matrix $\underset{\sim}{\mathbf{I}}$, $\overline{d(\omega)}d(\omega)$ will be constant. This corresponds to an input with a white spectrum. If all the terms of $\mathbf{d}^\dagger\mathbf{d}$ are constant, $\overline{d(\omega)}d(\omega)$ will be non-zero only at $\omega = 0$, and the inversion will be unstable. This corresponds to a data series

$\mathbf{d}$ containing a constant. It can be seen that $\overline{d(\omega)}A(\omega)$ is a measure of the information available at ω, and $(\overline{d(\omega)}d(\omega))^{-1}$ is a function of the uncertainty, or variance, at ω. The original autocorrelation matrix $\mathbf{d}^\dagger\mathbf{d}$ is the information matrix, and its inverse $(\mathbf{d}^\dagger\mathbf{d})^{-1}$ is the covariance matrix (Strang, 1986).

The expression $f(\omega) = \overline{d(\omega)}r(\omega)/(\overline{d(\omega)}d(\omega))$ will generally have a stabilizer in the denominator to avoid having $m(\omega)$ approach infinity when $\overline{d(\omega)}d(\omega)$ gets small. Adding this stabilizer in the frequency domain corresponds to adding a small value to the diagonal of the autocorrelation matrix. In the cases discussed here, the stabilizer will seldom be needed since random noise in the data generally keeps $\overline{d(\omega)}d(\omega)$ from going to zero.

3.2 Inverse Theory for signal and noise separation

3.2.1 Systems describing signal and noise separation

Claerbout (1995) provides a geophysical linear inverse structure of

$$\mathbf{0} \approx \underset{\sim}{\mathbf{W}}(\underset{\sim}{\mathbf{L}}\mathbf{m} - \mathbf{d}) \tag{3.27}$$

$$\mathbf{0} \approx \epsilon\underset{\sim}{\mathbf{A}}\mathbf{m}, \tag{3.28}$$

where $\underset{\sim}{\mathbf{W}}$, $\underset{\sim}{\mathbf{L}}$, and $\underset{\sim}{\mathbf{A}}$ are linear operators, $\mathbf{m}$ and $\mathbf{d}$ correspond to a model and to the data, and ϵ is a scale factor determining the relative weights of these two systems. The first regression involves how well the model fits the data. The second regression involves limitations on what the model is expected to be. The matrix $\underset{\sim}{\mathbf{A}}$ often enforces a smoothness on the model.

These are systems of regressions, rather than systems of equations. It is not expected that either $\underset{\sim}{\mathbf{W}}(\underset{\sim}{\mathbf{L}}\mathbf{m} - \mathbf{d})$ or $\epsilon\underset{\sim}{\mathbf{A}}\mathbf{m}$ should ever become exactly zero, but it is expected that some minimum of these expressions can be found by adjusting the model $\mathbf{m}$. Alternatively, the regressions in (3.27) and (3.28) can be expressed as the system of equations

$$\mathbf{e}_1 = \underset{\sim}{\mathbf{W}}(\underset{\sim}{\mathbf{L}}\mathbf{m} - \mathbf{d}) \tag{3.29}$$

$$\mathbf{e}_2 = \epsilon\underset{\sim}{\mathbf{A}}\mathbf{m}, \tag{3.30}$$

where $\mathbf{e}_1$ and $\mathbf{e}_2$ are to be minimized.

Paralleling the arguments in Claerbout (1995), if $\underset{\sim}{\mathbf{W}}$ is replaced with a filtering operator $\underset{\sim}{\mathbf{S}}$ that annihilates any signal s on which it operates so that $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$, $\underset{\sim}{\mathbf{I}}$ is replaced by the identity matrix $\underset{\sim}{\mathbf{I}}$, $\underset{\sim}{\mathbf{A}}$ is replaced by a filtering operator $\underset{\sim}{\mathbf{N}}$ that annihilates any noise $\mathbf{n}$ on which it operates, so that $\underset{\sim}{\mathbf{N}}\mathbf{n} \approx \mathbf{0}$, and $\mathbf{m}$ is replaced with the noise $\mathbf{n}$ that is to be calculated and removed from the data $\mathbf{d}$, equations (3.27) and (3.28) become

$$0 \approx \underset{\sim}{\mathbf{S}}(\mathbf{n} - \mathbf{d}) \tag{3.31}$$

$$0 \approx \epsilon \underset{\sim}{\mathbf{N}}\mathbf{n}. \tag{3.32}$$

An alternative method of getting equations (3.31) and (3.32) would be to start from the definitions of the signal annihilation filter and the noise annihilation filter:

$$0 \approx \underset{\sim}{\mathbf{S}}\mathbf{s} \tag{3.33}$$

$$0 \approx \underset{\sim}{\mathbf{N}}\mathbf{n}, \tag{3.34}$$

that is, the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ applied to the signal $\mathbf{s}$ produces something that is almost zero, and the noise annihilation filter $\underset{\sim}{\mathbf{N}}$ applied to the noise $\mathbf{n}$ produces something that is almost zero. Since the data $\mathbf{d}$ is defined as the sum of the signal $\mathbf{s}$ and the noise $\mathbf{n}$, that is, $\mathbf{d} = \mathbf{s} + \mathbf{n}$, $\mathbf{s}$ in equation (3.33) may be replaced with $\mathbf{d} - \mathbf{n}$. Making this substitution and allowing for a scale factor ϵ once more gives equations (3.31) and (3.32).

Combining the systems shown in (3.31) and (3.32) into a single system gives

$$\mathbf{0} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} & -\underset{\sim}{\mathbf{S}} \\ \epsilon\underset{\sim}{\mathbf{N}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ \mathbf{d} \end{pmatrix}. \tag{3.35}$$

Moving the expressions that depend on the data to the left-hand side and keeping those that depend on the unknown noise on the right-hand side gives

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ 0 \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon\underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{n}. \tag{3.36}$$

This system may be solved by either minimizing the error as in Claerbout (1995) or by substituting directly into the solution for the normal equations. In either case the solution for $\mathbf{n}$ is

$$\mathbf{n} = \left(\underset{\sim}{\mathbf{S}}^{\dagger}\underset{\sim}{\mathbf{S}} + \epsilon\underset{\sim}{\mathbf{N}}^{\dagger}\underset{\sim}{\mathbf{N}}\right)^{-1} \underset{\sim}{\mathbf{S}}^{\dagger}\underset{\sim}{\mathbf{S}}\mathbf{d}. \tag{3.37}$$

Once again, $\dagger$ indicates the conjugate-transpose, or adjoint, which is simply the transpose when all values are real.

Since the signal $\mathbf{s}$ may be expressed as $\mathbf{d} - \mathbf{n}$, the solution for the signal is

$$\mathbf{s} = \left(\underset{\sim}{\mathbf{S}}^{\dagger}\underset{\sim}{\mathbf{S}} + \epsilon\underset{\sim}{\mathbf{N}}^{\dagger}\underset{\sim}{\mathbf{N}}\right)^{-1}\epsilon\underset{\sim}{\mathbf{N}}^{\dagger}\underset{\sim}{\mathbf{N}}\mathbf{d}. \tag{3.38}$$

3.2.2 Frequency-wavenumber domain expression of the systems

Just as the inverse filtering expressions in frequency in 3.1.7 are converted to simple scalar systems, the expressions for the signal and noise are appealing when expressed in the frequency or in the frequency-wavenumber domains. The expressions for noise can be considered scalar expressions at a constant frequency in one dimension, and at a constant frequency and a constant wavenumber in two-dimensions. For example, in one dimension, equation (3.37) becomes

$$n(\omega) = \left(\frac{\overline{S(\omega)}S(\omega)}{\overline{S(\omega)}S(\omega) + \epsilon\overline{N(\omega)}N(\omega)}\right)d(\omega), \tag{3.39}$$

where $\overline{S}$ indicates the complex conjugate of S, and all values are scalars for a constant ω. For values of ω where signal is not expected, the value of $\overline{S(\omega)}S(\omega)$ rises and the weighting of the data into the noise increases. For values of ω where signal is expected, $\overline{S(\omega)}S(\omega)$ falls and the weighting of the data into the noise decreases. Equation (3.39) can be recognized as an optimal or Wiener filter (Press et al., 1986).

In the frequency-wavenumber, or ω-k, domain, the same idea applies, except that each value of $n(\omega, k)$ is evaluated for points in the ω-k plane instead of at points of constant frequency. The ω in equation (3.39) becomes (ω, k), and the expressions are separated into samples of constant wavenumber (k) and constant frequency (ω). This can be extended into more dimensions by specifying k_1, k_2, etc., for all the spatial directions considered. The inversion to separate signal and noise may then be thought of as decomposing the data into frequency and dip components, then distributing these components between the signal and noise as determined by the frequency and dip components of $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$. The advantage of having more dimensions is that the noise and signal may be better distinguished as they are spread over ω, k_1, k_2, and so on. Even if some overlap in the characterization of noise and signal remains, the overlap should tend to decrease as the number of dimensions increase.

Although viewing the separation of signal and noise in the frequency domains may be enlightening, for the work done in this thesis the inversions will generally be done in the

time and space domain which provides the advantages of simplicity of coding, control of the filter shape, and easy windowing of data to account for non-stationarity.

3.2.3 Incorporating gain into inversion

In the inversions considered so far, the signal $\mathbf{s}$ and noise $\mathbf{n}$ have been characterized only by the filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$, limiting the description of the data by the filters to the spectrum. For some applications, more information must be specified. In particular, the amplitudes of the seismic data should be considered. Recordings of seismic data show rapid weakening of the signal with time caused by a combination of wavefront spreading and the attenuation of the earth. If this weakening of the signal is not compensated for when an inversion is attempted, most of the inversion's effort will be expended on the strong shallow portion of the records. The weaker deeper portion of the records, which appears small in the least-squares sense, will be almost ignored by a least-squares solver. To equalize the treatments of the shallow and deep portions of the seismic records, this amplitude difference must be accounted for.

Another reason to account for the amplitudes is to take advantage of the extra information contained in the differences of the expected amplitudes of the noise and signal. Much noise originates from the surface and will either be of constant amplitude or weaken at a slower rate than does the signal. For example, in chapter 10 the noise is expected to be of constant amplitude, whereas the signal is expected to weaken as t^2, where t is sample time.

One method of accounting for the weakening of the signal would be to gain the input by t^2, but this would strengthen the noise at depth. A better method would be to account for the amplitude differences in the inversion itself. These amplitude differences may be taken advantage of by using them as part of the characterization of the signal and noise. As an example, suppose the noise amplitude falls off as t and the signal amplitude falls off as t^2. Systems (3.33) and (3.34) may be modified to become

$$0 \approx \underset{\sim}{\mathbf{S}} t^2 \mathbf{s} \tag{3.40}$$

$$0 \approx \underset{\sim}{\mathbf{N}} t \mathbf{n}, \tag{3.41}$$

then by substituting $\mathbf{d} - \mathbf{n}$ for $\mathbf{s}$, system (3.36) becomes

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}} t^2 \mathbf{d} \\ 0 \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} t^2 \\ \epsilon \underset{\sim}{\mathbf{N}} t \end{pmatrix} \mathbf{n}. \tag{3.42}$$

The factors t and t^2 might be considered as weights that control the distribution of the expected signal and noise, as well as being factors that equalize the contributions of the signal and noise to the output. The factors t and t^2 in system (3.42) are in fact matrices that have the values of t and t^2 along the diagonal that correspond to the time values of the samples of $\mathbf{n}$ and $\mathbf{s}$. For simplicity, I will represent these matrices with t and t^2 here and later in this thesis.

Notice that the assumption of stationarity for $\mathbf{s}$ and $\mathbf{n}$ has been violated somewhat by the time scaling. This is not necessarily a problem since the filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ involve only the spectrum of the signal and noise. This spectrum could be assumed to be constant. The functions t and t^2 that balance the contributions of $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ in (3.42) would presumably be applied to the data from which $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ are calculated so the scaled $\mathbf{s}$ and n would be stationary. In system (3.42) it is assumed that the filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ are small enough to ignore the variation of $\mathbf{s}$ and $\mathbf{n}$ within the filter caused by the t and t^2 scaling. If this is not true, there will be a difference between applying the scaling before the filters and applying the scaling after the filters.

Chapter 4

Multi-dimensional filter design

4.1 Filter shapes and dimensionality

One-dimensional filters have been used on seismic data practically as long as seismic data have been collected. As a step beyond simple bandpass filters, adaptive filtering techniques have been applied successfully to seismic data for years (Webster, 1978). In particular, predictive deconvolution has been especially useful in areas such as the Gulf of Mexico to remove short period multiples (Dobrin and Savit, 1988; Robinson and Treitel, 1980; Kanasewich, 1981). This thesis addresses some aspects of the multi-dimensional extensions to the previous one-dimensional techniques.

4.1.1 Filter dimensionality

Since the phenomena measured by seismic techniques are multi-dimensional, the manipulation of the measurements of these phenomena will generally require multi-dimensional filters. While in the case of deconvolution in time, the reverberation effect is approximately contained in one dimension, this one-dimensional approximation breaks down for long-period reverberations. In the case of f-x prediction, two-dimensional filtering is done by decomposing the problem into a large set of one-dimensional complex filters. The emphasis in this thesis is on the more general case of both calculating and applying multi-dimensional filters directly with multi-dimensional convolutions.

One characteristic of multi-dimensional filters is that the number of samples contributing to an output point increases rapidly with the dimensionality of the data. This is an advantage since much more information is available to predict an output point, or, from

a different point of view, given a constant number of samples from which to predict, more dimensions allow the predictions to be done from smaller distances. A disadvantage of using many dimensions is that the cost of computation also rises quickly with the dimensionality of the data. Fortunately, the complexity of the programs needed for applying and calculating multi-dimensional filters rises slowly with the number of dimensions, especially when the filters are calculated with Claerbout's conjugate-gradient methods (Claerbout, 1992a; Claerbout, 1995). The relative simplicity of this approach is important, since calculating even two-dimensional filters using the traditional techniques used to calculate one-dimensional filters appears to be a formidable task, and the problem becomes more complex as the dimensionality increases. Almost all of the filters used in this thesis have been calculated with the conjugate-gradient methods.

4.1.2 Filter shapes

Claerbout(1992a) presented a proof attributed to John Burg that shows a one-sided, or Burg, filter, when calculated to minimize the energy of a signal, will have a spectrum that is the inverse of the signal. Thus, when the filter is applied to the signal, the result is white. Claerbout extends this proof to a one-sided two-dimensional filter in his later work (Claerbout, 1995). These properties are useful in predicting the behavior of a filter, so it is with a certain reluctance that shapes other than the one-sided, or Burg, filters are used.

Nevertheless, other shapes have particular advantages. Claerbout's steep-dip deconvolution (Claerbout, 1993) uses filters that allow the prediction and removal of steeply dipping events such as ground roll, but preserves reflections. The filter used in this case

is a two-dimensional filter similar to the following:

$$\begin{array}{ccccccccc}
a & a & a & a & a & a & a & a & a \\
a & a & a & a & a & a & a & a & a \\
0 & a & a & a & a & a & a & a & 0 \\
0 & a & a & a & a & a & a & a & 0 \\
0 & a & a & a & a & a & a & a & 0 \\
0 & 0 & a & a & a & a & a & 0 & 0 \\
0 & 0 & a & a & a & a & a & 0 & 0 \\
0 & 0 & a & a & a & a & a & 0 & 0 \\
0 & 0 & 0 & a & a & a & 0 & 0 & 0 \\
0 & 0 & 0 & a & a & a & 0 & 0 & 0 \\
0 & 0 & 0 & a & a & a & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{array} \tag{4.1}$$

where the 1 is the output position and each a denotes a (different) adjustable filter coefficient that is chosen to minimize the power output. The corresponding one-dimensional filter would appear as:

$$\begin{array}{c}
a \\
a \\
a \\
a \\
a \\
a \\
a \\
a \\
a \\
0 \\
0 \\
0 \\
1
\end{array} \tag{4.2}$$

which is simply a standard deconvolution filter. Claerbout's steep-dip decon is then a two-dimensional extension to standard deconvolution which allows predictions from samples in traces near the output trace. The cost of this process is high, since a new two-dimensional filter is calculated for each trace, but the benefits may be worth the extra computation.

Another example of a shaped filter is one that is often used in this thesis. This is that of a purely lateral prediction filter. Burg's two-dimensional filter

$$\begin{array}{ccccc} 0 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\ 0 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\ 1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \end{array} \tag{4.3}$$

may be modified to make a purely lateral prediction filter. To make the filter purely lateral requires that predictions not be done from within the trace which hold the output sample. No predictions are done in the vertical direction. The purely lateral version of the filter shown in (4.3) is:

$$\begin{array}{ccccc} 0 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\ 0 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\ 1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ 0 & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ 0 & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \end{array} \quad . \tag{4.4}$$

Note that only the first column has changed. The output point, under the 1 coefficient, is the only non-zero coefficient in that column. This eliminates any predictions done within a trace.

Eliminating predictions from within a trace is important because the filter coefficients corresponding to prediction within a trace tend to overwhelm the predictions done from trace to trace. Also, since the assumption that signal is consistent from trace to trace has been made, only the trace-to-trace predictions are wanted. When doing these trace-to-trace, or purely lateral predictions, these intra-trace predictions are undesired, and so should be eliminated.

A three-dimensional version of Burg's filter appears in Figure 4.1. The corresponding purely-lateral filter is shown in Figure 4.2. Once again, the coefficients within the trace are eliminated.

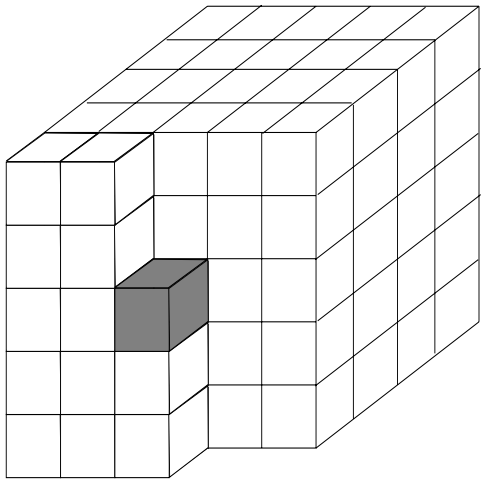

FIG. 4.1. The Burg 3-D filter. (After Claerbout,1995). The vertical direction is time, and the other directions are in space. filters-burg3D [NR]

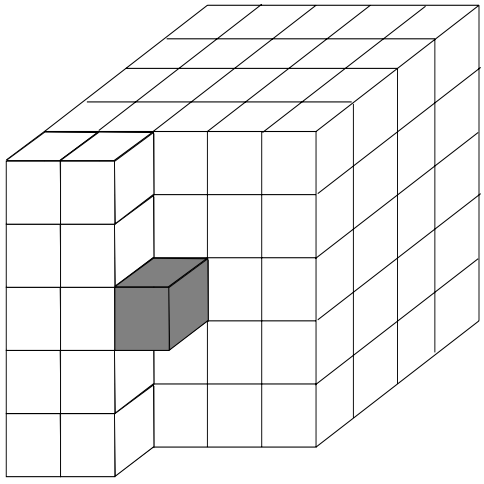

FIG. 4.2. The purely-lateral 3-D filter. The vertical direction is time, and the other directions are in space. filters-lateral3d [NR]

Filtering with purely-lateral filters also brings up a fundamental difference between how the time axis and how the spatial axes are treated in seismic processing. Along the time axis, the unpredictable information is important, so predictable information, such as source wavelets and reverberations from multiple reflections are eliminated. Traditionally, this has been done by prediction-error filtering, or deconvolution. Along the spatial axes, the important information consists of predictable events, and unpredictable information is generally assumed to be noise. This difference comes from both the sedimentary character of most of the geology considered in seismic exploration and from the geometry of surface-seismic recording, where both the sources and detectors are at the surface. Any energy recorded that appears above a certain angle is evanescent (Claerbout, 1985) and does not contain information about the desired reflection events.

Changes to the filter shapes allows events to be predicted to be separated better. As an example, in chapter 9, noise which is expected to be confined to a single trace is predicted by one-dimensional filters. The signal, on the other hand, is predicted by a purely-lateral two- or three-dimensional filter. The success of the inversion depends on how well these differently shaped filters predict the different phenomena.

4.2 The calculation of a filter

4.2.1 Least-squares methods

While measures of the residuals other than the L-2 norm may be used, the least-squares method is used here. Norms between one and two may be obtained with the iteratively re-weighted least-squares (IRLS) method (Nichols, 1994b; Darche, 1989; Green, 1984). Norms other than L-2, especially L-1, may be used in cases where the presence of high-amplitude events overwhelm the least-squares residuals.

In this thesis, two different approaches to high-amplitude errors are taken. In chapter 7, the problem of high-amplitude noise corrupting the filter calculation is solved by iteratively solving for a signal and a signal annihilation filter. Another approach is taken in chapter 2, where samples that produce large residuals are removed before the inversion, then, in chapter 8, the signal and noise are separated simultaneously with the prediction of the missing data caused by the sample removal. It is hoped that these techniques will solve most practical problems.

The calculation of a filter is done by solving $\mathbf{0} \approx \underset{\sim}{\mathbf{D}}\mathbf{f}$, where $\underset{\sim}{\mathbf{D}}$ is a matrix made up of the

given data and $\mathbf{f}$ is the filter to be solved for. Claerbout(1992a) shows how to solve for this filter $\mathbf{f}$ by expressing the convolution of the filter $\mathbf{f}$ and the data $\underset{\sim}{\mathbf{D}}$ as matrix operators and their adjoints. If the operator and its adjoint are available, a conjugate-gradient routine may be used to calculate a least-squares minimization of a system (Luenburger, 1984). This approach to computing the filter simplifies the problem considerably. Two- and three-dimensional filters may be calculated as simply as a one-dimensional filter, provided the filter operation and its adjoint are available.

4.2.2 Methods of using a filter

As stated in the introduction, the filters calculated here will be used in two techniques: in simple filtering and in inversions.

For signal and noise separation by filtering, the noise is expected to be whatever remains after a signal-annihilation filter is applied to the data. This can be expressed as $\mathbf{n}_{\text{pef}} = \underset{\sim}{\mathbf{S}}\mathbf{d}$, where $\underset{\sim}{\mathbf{S}}$ is the signal-annihilation filter, $\mathbf{d}$ is the data, and $\mathbf{n}_{\text{pef}}$ is the prediction-error filter estimate of the noise. While the desired action of the filter is $\mathbf{0} \approx \underset{\sim}{\mathbf{S}}\mathbf{s}$, where $\mathbf{s}$ is the signal, the signal $\mathbf{s}$ is not available for calculating $\underset{\sim}{\mathbf{S}}$. In the case where the noise is considered to be unpredictable, the filter $\underset{\sim}{\mathbf{S}}$ can be calculated by minimizing $\underset{\sim}{\mathbf{S}}\mathbf{d}$. This makes $\underset{\sim}{\mathbf{S}}\mathbf{d}$ the prediction error, and the result for $\mathbf{n}_{\text{pef}}$ is the prediction-error filtering estimate of the noise. The t-x and f-x prediction filtering discussed in chapters 5 and 6 calculates the prediction-error filtering estimate of the noise $\mathbf{n}_{\text{pef}}$ with the filtering $\underset{\sim}{\mathbf{S}}\mathbf{d}$ done in two and three dimensions.

As pointed out by Soubaras(1994), the noise as defined by prediction-error filtering is inconsistent with the definition that the data is the sum of the signal and noise, $\mathbf{d} = \mathbf{s} + \mathbf{n}$, even though $\mathbf{s}$ is calculated as $\mathbf{s} = \mathbf{d} - \mathbf{n}_{\text{pef}}$. If the signal $\mathbf{s}$ is perfectly predicted by filter $\underset{\sim}{\mathbf{S}}$, the signal will be completely annihilated by the filter so that $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$. When $\underset{\sim}{\mathbf{S}}$ is applied to $\mathbf{d} = \mathbf{s} + \mathbf{n}$, the result is $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{n}$, as opposed to $\underset{\sim}{\mathbf{S}}\mathbf{d} = \mathbf{n}_{\text{pef}}$, the definition of noise by prediction-error filtering. Thus, to avoid a conflict of these definitions, prediction-error filtering requires a filter that removes the signal $\mathbf{s}$ without disturbing the noise $\mathbf{n}$, which is expressed as $\underset{\sim}{\mathbf{S}}\mathbf{n} \approx \mathbf{n}$. If the prediction-error filtering result $\underset{\sim}{\mathbf{S}}\mathbf{d}$ or $\underset{\sim}{\mathbf{S}}\mathbf{n}$ is not close to the actual noise $\mathbf{n}$, the accuracy of the signal calculated from $\mathbf{d} - \mathbf{n}_{\text{pef}}$ will be compromised. In short, a prediction-error filter must not distort the noise. This requirement will always be violated to some extent.

If the filters are used in an inversion, the assumption that $\underset{\sim}{\mathbf{S}}\mathbf{n} \approx \mathbf{n}$ is not required. The

inversion only requires that $\mathbf{0} \approx \underset{\sim}{\mathbf{S}}\mathbf{s}$. The form of $\underset{\sim}{\mathbf{S}}\mathbf{n}$ is less important. $\underset{\sim}{\mathbf{S}}\mathbf{n}$ may be, for example, reversed in polarity or time-shifted when compared to $\mathbf{n}$, and the inversion will still function well. This might be understood as making the phase of the filter unimportant, since the spectral power is the main concern.

One advantage of using a filter in an inversion is that more freedom is allowed for correcting the results to account for missing data, as described in the previous chapter and examined in more detail in chapter 8. While the application prediction-error filters can be modified to treat some missing data problems by predicting in only one direction, inversion gives a more natural method of allowing for missing data, as well as predicting and restoring the missing data.

Chapter 5

Noise removal by filtering

This chapter discusses two approaches to predicting linear events: a frequency-space, or f-x, prediction technique, and a time-space, or t-x, prediction technique. The f-x prediction technique was introduced by Canales (1984) and further developed by Gulunay (1986), based on Treitel's complex series prediction work (Treitel, 1974). This technique divides the two-dimensional filtering problem into many one-dimensional filtering problems in space, one for each frequency. These f-x prediction techniques were significant improvements over the other noise attenuation methods available at that time. The name Gulunay used for this process was FXDECON, which stood for frequency-space domain predictive deconvolution. The reference to deconvolution is something of a misnomer, since deconvolution refers to the removal of predictable information, whereas in this chapter, the data of interest are the predictable parts of the input. After a transformation of the data from the time-space domain into the frequency-space domain, the process of predicting a linear event can be divided into many smaller problems of predicting periodic events within a frequency. The t-x prediction process presented in this chapter is done with a single prediction filter calculated in the time-space domain using a conjugate-gradient method. The conjugate-gradient method and programs for filter calculations similar to the ones used here are discussed in Claerbout (1992a).

While the two methods generally produce similar results, t-x prediction has several advantages over the older f-x prediction. These advantages allow t-x prediction to pass less random noise than the f-x prediction method. Most of the extra random noise is passed because the f-x prediction technique, while dividing the prediction problem into separate problems for each frequency, produces a filter as long as the data series in time when the

collection of filters is transformed into a single filter in the t-x domain. Because the f-x prediction filter is very long in the time direction, its many free filter coefficients allow some random noise to be passed and spurious events to be generated. Since the length in time of the t-x prediction filter can be controlled, t-x prediction avoids the disadvantages of f-x prediction.

In three dimensions, better results are expected for both techniques since more data goes into every prediction and since some of the linearity assumptions can be relaxed in three dimensions (Chase, 1992; Abma, 1993; Gulunay et al., 1993). . I found that extending these prediction techniques from two dimensions into three dimensions produces better results than two passes of the 2-dimensional processes in the inline and crossline directions.

This chapter examines these two prediction techniques and compares the results in both two and three dimensions. I also show some of the advantages that 3-dimensional prediction has over the 2-dimensional applications of these techniques.

5.1 Two-dimensional lateral prediction

Much seismic data, especially those acquired on land, are contaminated with random noise that impedes interpretation and interferes with further processing and analysis. Random noise is recognized by its dissimilarity from trace to trace. Signal, on the other hand, is recognized by its lateral continuity. Much of this continuity results from the sedimentary character of the data being considered.

The methods I consider here predict only linear events. While the human eye recognizes the continuity of nonlinear events, the mathematical tools available work best on linear problems. Even though many continuous seismic events are not linear, windowing the image into smaller areas makes most of these events at least approximately linear. Hornbostel (1991) introduced a t-x prediction technique that allowed rapidly changing data without requiring windowing, but in this chapter, I used the same windowing technique for both the f-x and t-x predictions.

5.1.1 Prediction of seismic signals in the t-x domain

The 2-dimensional t-x technique predicts linear events with a 2-dimensional time-space domain filter. This filter is calculated to minimize the energy inside a design window using

a conjugate-gradient routine [Claerbout (1992a)]. The filters used in the examples in this article have the form:

$$\begin{matrix} 0 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\ 0 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\ 1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ 0 & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ 0 & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \end{matrix} \quad . \tag{5.1}$$

The vertical axis is the time axis, and the horizontal axis is the space axis. The output position is under the 1 coefficient on the left side of the filter. This filter has no free coefficients in the column corresponding to the output trace, which forces whatever predictions are made to be lateral. Application of this filter removes predictable energy from the design window, leaving the unpredictable part, which is considered here to be noise. Subtracting the unpredictable data from the original data produces the predictable data.

As an example of a calculated filter, a flat event in a noiseless window produces a filter with all zeros except on the row containing the 1, where each of the a_{01}, a_{02}, a_{03}, and a_{04} coefficients in the filter above becomes -0.25. This filter minimizes the energy of the output by exactly predicting the flat event. Anything left after filtering is considered noise and removed from the input. Although making any one of the a_{01}, a_{02}, a_{03}, or a_{04} coefficients -1 produces a filter that exactly predicts an event, I set up the problem so that the coefficients tend to be equal for the best noise attenuation.

The choice of the filter size depends on the size of the design window, the maximum dip in the data, the number of dips within the window, and the desired strength of the prediction effect. For t-x prediction, the choice of the filter's length in space is similar to the choice of the filter length in f-x prediction. Enough traces need to be included to create a good estimate of the signal. I tend to use 5 to 7 traces on 2-dimensional data, while for 3-dimensional data, this number can be somewhat smaller in each direction. The filter length in time for the t-x filter does not seem to be an especially sensitive parameter unless very steep events exist. In the examples shown here, I used 3 to 5 samples. Events that have a moveout greater than the filter length in time are easily predicted, provided the bandwidth of the data is not too high. To keep the application of the calculated filter symmetrical, the filter is applied in both forward and reverse directions in space, with the results averaged.

The process of applying t-x prediction, as well as the f-x prediction discussed next, is

applied to windows small enough for events of interest to appear linear. After filtering, these windows are merged to produce the output image.

5.1.2 Prediction of seismic signals in the f-x domain

An f-x prediction like Gulunay's (1986) predicts linear events in the frequency-space domain. A linear event given by the expression $r(x,t) = \delta(a+bx-t)$, where x is the lateral position and t is time, when Fourier transformed in time becomes $r(x,\omega) = e^{i\omega(a+bx)}$ or $r(x,\omega) = e^{i\omega a}(\cos(\omega bx) + i\sin(\omega bx))$ (Bracewell, 1978; Briggs and Henson, 1995; Arfken, 1985). For a simple linear event, this function is periodic in x. This periodicity can be seen along any constant frequency line in the f-x domain display in Figure 5.1.

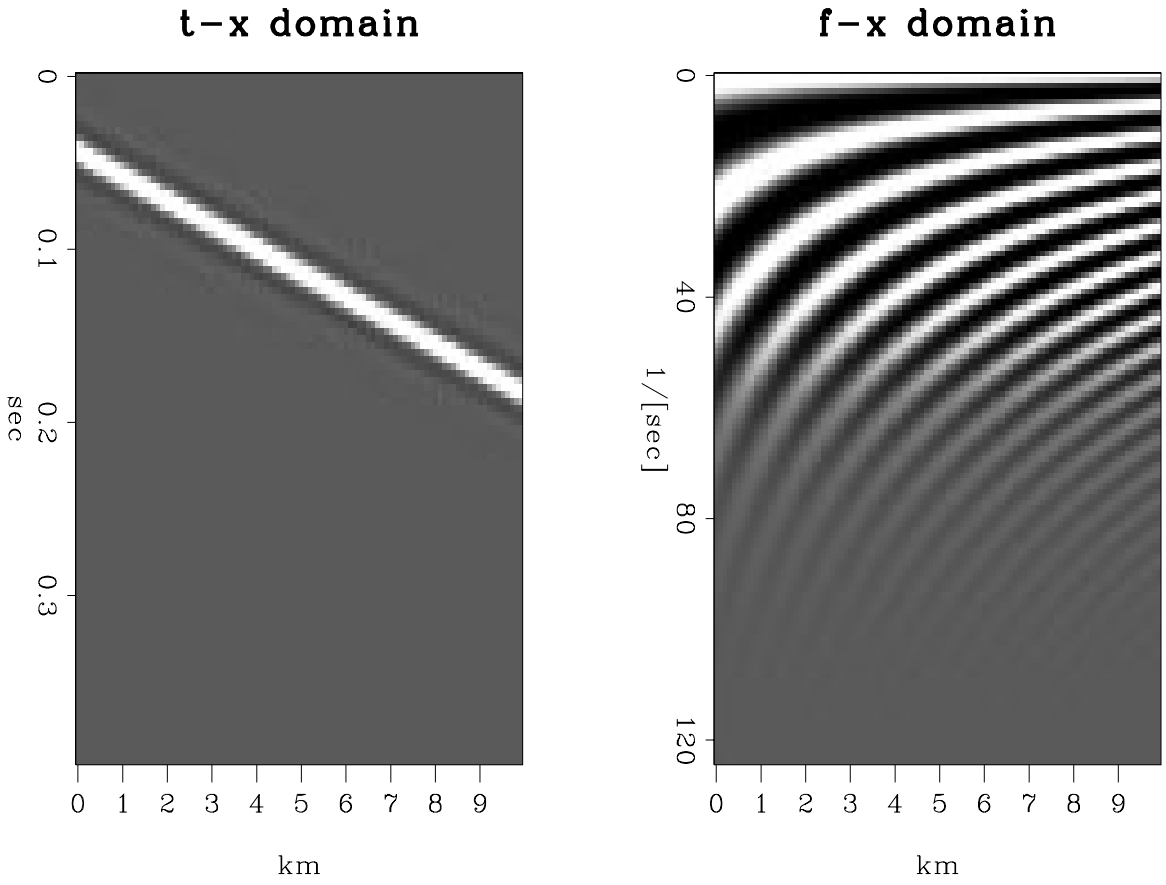

FIG. 5.1. A single dip shown in the t-x domain and in the real part of the f-x domain. In the f-x domain, the signal is periodic along any horizontal line. TXFX-sdip [R]

To predict a linear event in $\mathbf{x}$, where $\mathbf{x}$ is a sampled version of $r(x,\omega)$ for a single

frequency, Gulunay (1986) proposed calculating a least-squares prediction filter $\mathbf{f}$ from the system $\mathbf{d} = \underset{\sim}{\mathbf{X}}\mathbf{f}$, or, as expanded,

$$\begin{pmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \\ \cdot \\ \cdot \\ \cdot \\ x_{n+1} \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_1 & 0 & 0 & 0 \\ x_2 & x_1 & 0 & 0 \\ x_3 & x_2 & x_1 & 0 \\ x_4 & x_3 & x_2 & x_1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ x_n & x_{n-1} & x_{n-2} & x_{n-3} \\ 0 & x_n & x_{n-1} & x_{n-2} \\ 0 & 0 & x_n & x_{n-1} \\ 0 & 0 & 0 & x_n \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}. \tag{5.2}$$

The input to the prediction problem is the data from a single frequency over the width of the window in space. This input is a set of complex numbers ($x_1 \quad x_2 \quad x_3 \quad \ldots \quad x_n$). The desired output $\mathbf{d}$ is ($x_2 \quad x_3 \quad \cdots \quad x_n \quad x_{n+1}$), a one-sample step-ahead prediction of the input. While $\mathbf{d}$ can be built with a longer shift of the input data (Hornbostel, 1991), a shift of one sample is generally used. Notice that the desired output $\mathbf{d}$ starts with x_2 and ends with x_{n+1}. Gulunay (1986) set up the problem using this extra element from the input to guarantee that the filter would produce a result with the same amplitude as the input regardless of the length of the filter and of the data. The rows of the matrix $\underset{\sim}{\mathbf{X}}$ are shifted versions of the input that produce a convolution with the desired filter $\mathbf{f}$. This filter $\mathbf{f}$ can be calculated using the normal equations $\mathbf{f} = (\underset{\sim}{\mathbf{X}}^\dagger \underset{\sim}{\mathbf{X}})^{-1} \underset{\sim}{\mathbf{X}}^\dagger \mathbf{d}$, where $\dagger$ indicates the conjugate transpose, or adjoint. This is a standard solution to least-squares problems, except that the sample values are complex numbers, so the adjoint operation $\dagger$ cannot ignore taking the complex conjugate of the matrix elements on which it operates.

The f-x prediction is applied to small windows to ensure that events are locally linear, just as in the t-x prediction case, and the data within each window are then Fourier transformed. For the spatial series created at each frequency by the Fourier transform, a prediction filter is calculated as described in the preceding paragraph. Each calculated filter is first applied forward and then reversed in space, with the results averaged to maintain a symmetrical application, as in the t-x prediction case. The inverse Fourier transform is then applied to the result in each window, and the windows are merged to

form the output image.

The calculation of the filter for each frequency is independent of the calculations of the filters for other frequencies. While the filters calculated at each frequency are a least-squares solution, this multitude of least-squares solutions does not necessarily produce a collective filtering action that is the best result. In the next section, I discuss the effect of this partitioning and the resulting differences between the actions of t-x prediction and f-x prediction.

5.1.3 The relationship of f-x prediction to t-x prediction

When applied to data with relatively mild random noise problems, most of the cases where t-x prediction was compared to f-x prediction showed practically identical results. In tests on synthetics with simple flat or dipping events in the absence of noise, both techniques passed events without distortion. These same simple cases with an added background of relatively low-amplitude random noise also produced comparable results. The results using real data with both techniques were also similar. Differences appeared only when the signal-to-noise ratio is low.

These similarities and differences are explained by comparing the form of the t-x prediction filter and the form of the effective t-x domain filter created by f-x prediction. The individual filters calculated by the f-x prediction have the form

$$1 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \ , \tag{5.3}$$

where a_1, a_2, a_3, and a_4 are complex numbers. Collecting these filters and presenting them

in the frequency-space domain produces the following composite filter:

$$\begin{matrix}
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & a_{-4,1} & a_{-4,2} & a_{-4,3} & a_{-4,4} \\
1 & a_{-3,1} & a_{-3,2} & a_{-3,3} & a_{-3,4} \\
1 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\
1 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\
1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\
1 & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\
1 & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\
1 & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\
1 & a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{matrix} \quad . \tag{5.4}$$

Transforming this filter from the frequency-space domain into the time-space domain gives the effective t-x domain filter,

$$\begin{matrix}
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & b_{-4,1} & b_{-4,2} & b_{-4,3} & b_{-4,4} \\
0 & b_{-3,1} & b_{-3,2} & b_{-3,3} & b_{-3,4} \\
0 & b_{-2,1} & b_{-2,2} & b_{-2,3} & b_{-2,4} \\
0 & b_{-1,1} & b_{-1,2} & b_{-1,3} & b_{-1,4} \\
1 & b_{0,1} & b_{0,2} & b_{0,3} & b_{0,4} \\
0 & b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\
0 & b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\
0 & b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\
0 & b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{matrix} \quad , \tag{5.5}$$

where the number of rows is the number of time samples in the window being considered. The Fourier transform has converted the first column of 1s into a single 1 at the output position, creating a time-space filter with the same form as the t-x prediction filter shown in (5.1), but with the number of rows greatly increased.

For data with a high signal-to-noise ratio, the coefficients of the filter producing most of the prediction are expected to lie close to the center of the filter in time, since events far

from the output point in time are unlikely to affect data at the output position. For data with a low signal-to-noise ratio, correlations between random events widely separated in time create undesirable lineups with significant amplitudes in the f-x prediction results. Since the f-x prediction produces a very long effective t-x filter in time, its prediction is contaminated with these random correlations, where the t-x prediction, with its smaller filter, is not. Examples showing that the f-x prediction passes more noise than the t-x prediction are presented later in this chapter for both 2-dimensional and 3-dimensional cases.

The time-length problem of the f-x prediction filter can be compared to the problem of calculating a one-dimensional prediction filter from a short time series. If a long one-dimensional prediction filter is calculated and applied to a short time series, practically everything in the time series will be predicted because of the many degrees of freedom allowed by the many filter coefficients. Having a 2-dimensional filter with a significant spatial width compensates for some of this over-prediction, but a t-x prediction filter with a short length in time is a more complete solution to the problem.

Another effect of having a long filter length in time for f-x prediction is the generation of false events. If parallel events are embedded in a background of random noise, f-x prediction generates events parallel to the original events. While both t-x and f-x techniques tend to line up noise with strong events, f-x prediction actually generates events because of its long filter length in time. These spurious events occur since parallel events in a random noise background produce an f-x filter where the prediction of one event is influenced by the presence of the other events parallel to it in spite of being widely separated in time. The t-x prediction is unaffected by the influence of widely separated events because of its short filter length in time.

An example of how f-x prediction generates spurious events can be seen in Figure 5.2. In this case, two flat events are immersed in noise on the left side of the displays, and the right side of the display is left free from noise to show the response of the filter. In both the f-x and the t-x predictions just one design window is used, covering all the data. With f-x prediction, events widely spaced in time can be used to predict an output point. For the f-x prediction in Figure 5.2, the noise on the left side of the input allows the predictions to be made using input from both events, so the equivalent t-x filter has significant coefficients far in time from the output point. When this filter is applied to the clean data on the right side, spurious events generated by these widely separated coefficients are seen. In the f-x

prediction result, strong events are generated above and below the two original events, and weaker events can be seen lined up with the original events on the right side. The t-x prediction results do not show these erroneous events. While the f-x prediction may be modified to eliminate this problem by constraining the filter coefficients to be smooth in frequency, the t-x prediction did not generate spurious events since its length in time is more naturally controlled.

The generation of spurious events will be considered in more detail in the next chapter. Although in the examples in Figure 5.2 show that spurious events are not generated by t-x prediction in this case, cases where events are closely spaced allow spurious events to be generated by both f-x and t-x prediction.

A typical f-x prediction program has an advantage over a similar t-x prediction program because the filter length in time does not need to be specified and therefore cannot accidentally be made too short. The cost of not requiring this parameter is the risk of generating false events and of passing more noise than a t-x prediction.

5.1.4 Comparison of two-dimensional f-x and t-x predictions

I compared the results of varying the time-extents for t-x prediction filters in the presence of high-amplitude noise against the results of f-x prediction. In a simple noise-only case, when I used the same number of coefficients in the lateral direction for both processes, f-x prediction passed about twice the noise energy as a short time-length t-x prediction. As the time-length of the t-x prediction filter increased, the t-x prediction passed more random noise. When filters with a time-length comparable to the data time-length were used, I found almost no difference between the filters or between the results of t-x prediction and f-x prediction. Thus, because of its ability to limit the filter length in time, t-x prediction has a definite advantage over f-x prediction in removing random noise.

This difference in passing random noise is shown in Figure 5.3, where the results of the t-x prediction and f-x prediction are applied to a 2-dimensional stacked section. Here I used a time length of five coefficients for the t-x prediction result and a spatial length of five coefficients for both prediction methods. The window sizes in both cases were 30 traces by 300 time samples, or 0.6 seconds in this instance. This section contains a moderate amount of noise and was one of the few non-synthetic cases showing recognizable differences between t-x prediction and f-x prediction. While the results are similar, f-x prediction passes somewhat more noise than t-x prediction. The noise is most apparent in

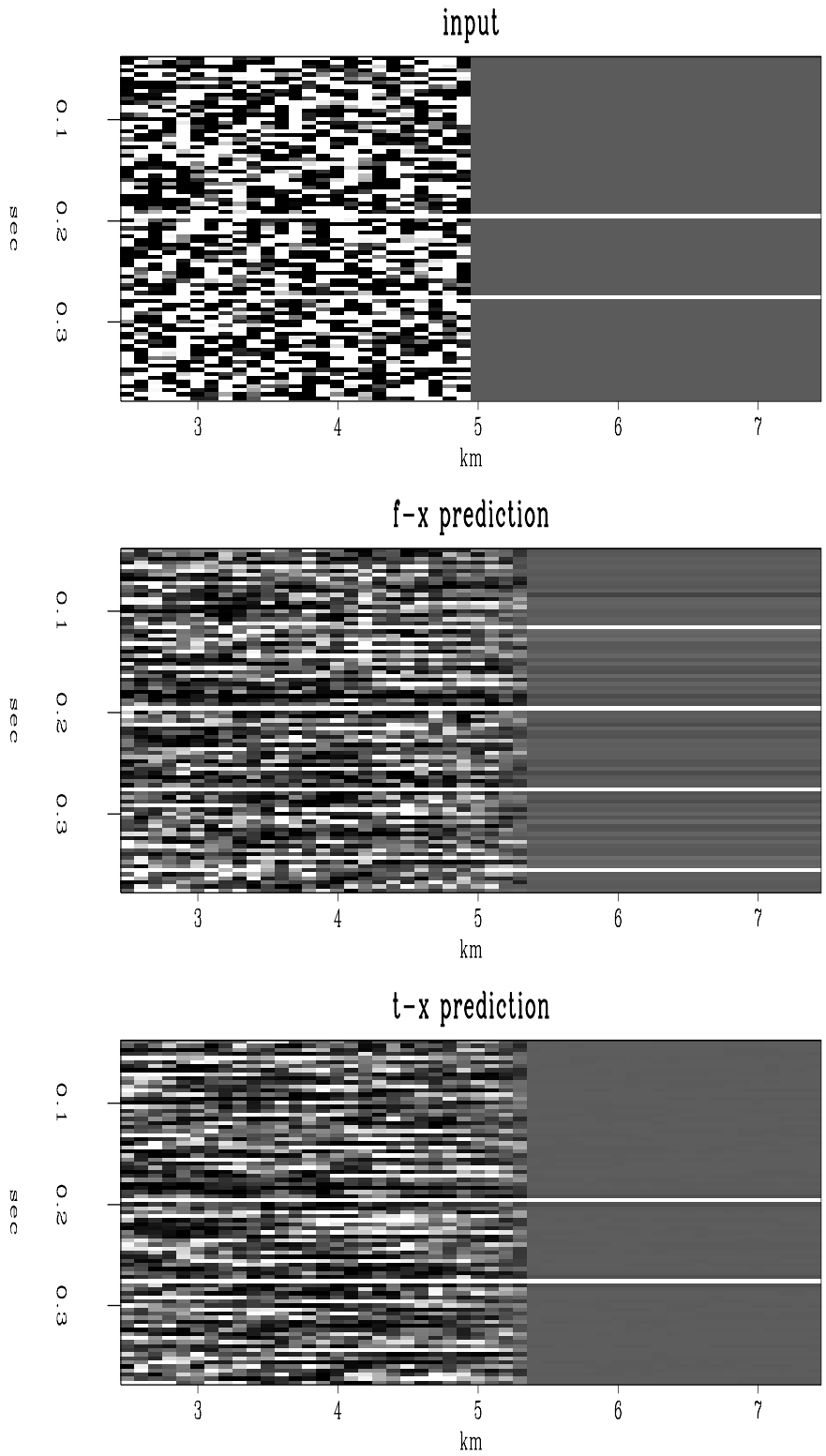

FIG. 5.2. An example of the creation of false events with f-x prediction and the corresponding result from t-x prediction. Notice that the f-x prediction has created a few strong events and numerous weak events, while the t-x prediction is clean. TXFX-two [R]

the originally zeroed area above about 0.3 seconds and in the deepest part of the section below 1.2 seconds.

5.1.5 The biasing of f-x prediction toward the output point

Though less of a problem than the long time-length filter, the tendency for the lateral prediction coefficients to be concentrated near the output trace position with Gulunay's f-x prediction (1986) should cause slightly more noise to be passed than with t-x prediction. The source of this biasing can be seen in the system describing the f-x prediction filtering, $\mathbf{d} = \underset{\sim}{\mathbf{X}}\mathbf{f}$, or

$$\begin{pmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \\ \cdot \\ \cdot \\ \cdot \\ x_{n+1} \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_1 & 0 & 0 & 0 \\ x_2 & x_1 & 0 & 0 \\ x_3 & x_2 & x_1 & 0 \\ x_4 & x_3 & x_2 & x_1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ x_n & x_{n-1} & x_{n-2} & x_{n-3} \\ 0 & x_n & x_{n-1} & x_{n-2} \\ 0 & 0 & x_n & x_{n-1} \\ 0 & 0 & 0 & x_n \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}, \tag{5.6}$$

where the vector $\mathbf{d}$ is the desired output, $\underset{\sim}{\mathbf{X}}$ is the input data shifted with respect to the filter, and $\mathbf{f}$ is the filter to be calculated. The zeros in the last few coefficients of $\mathbf{d}$ tend to reduce the f_2, f_3, and f_4 coefficients of the filter. The zeros at the top of the $\underset{\sim}{\mathbf{X}}$ matrix have a similar effect. The result of this tendency to weight the f-x prediction filter coefficients toward the output point of the filter may be appealing in terms of producing an output trace made up of the nearest traces, but the noise in the nearest traces is also passed with less attenuation. This increased weighting of the nearest traces produces a filter that is slightly less effective in rejecting noise.

The increased weighting of the nearest trace can be eliminated by setting up the

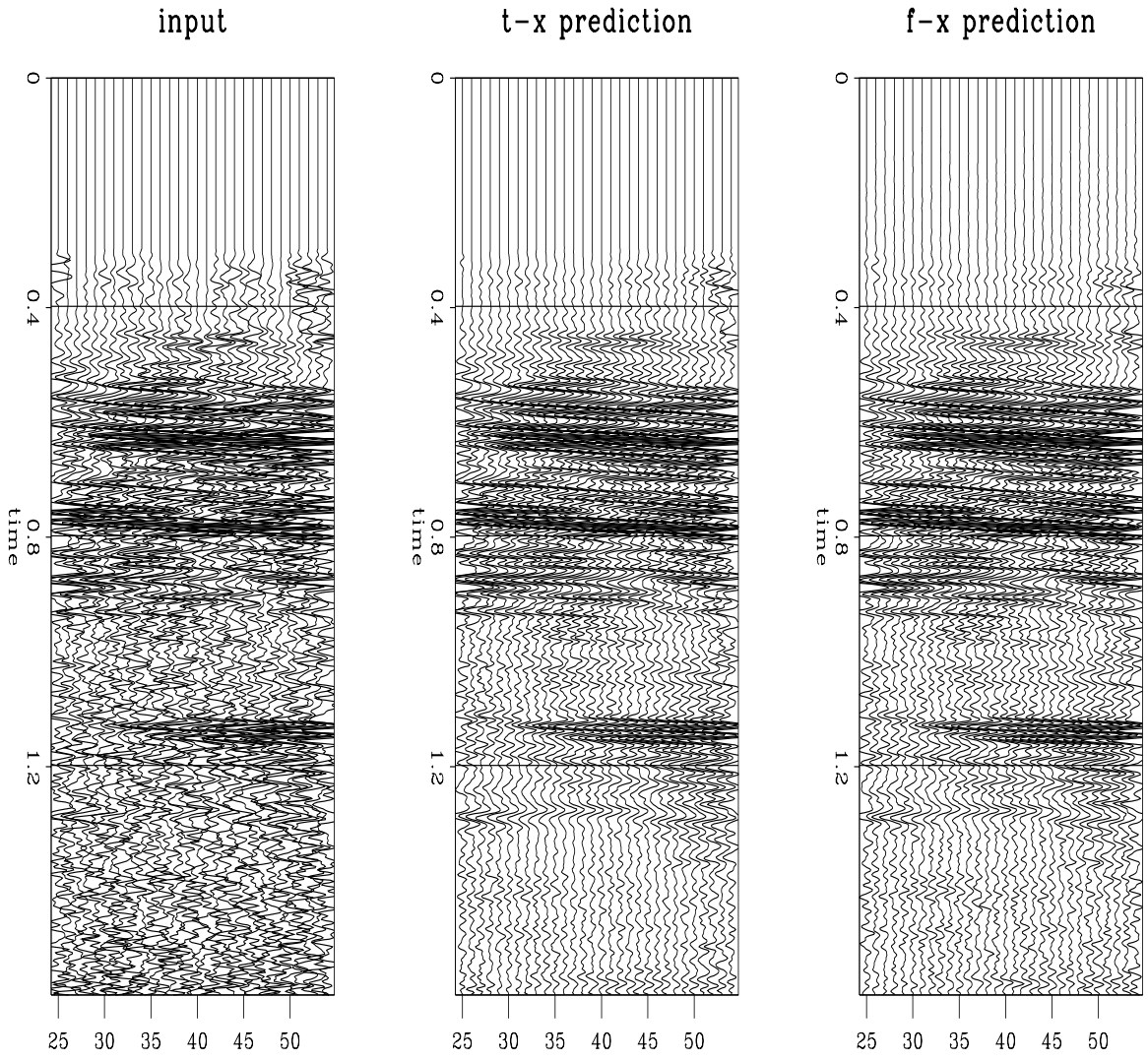

FIG. 5.3. 2-dimensional t-x prediction and f-x prediction on a stacked line. The left section is the input; the middle section the t-x prediction result; the right section the f-x prediction result. The results of the two techniques are similar, but somewhat less noise is passed with t-x prediction than with f-x prediction. TXFX-DDw [R]

problem differently. Removing the top and bottom rows of equation (5.6) produces

$$\begin{pmatrix} x_5 \\ x_6 \\ x_7 \\ . \\ . \\ . \\ x_{n-1} \\ x_n \\ x_{n+1} \end{pmatrix} = \begin{pmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ x_6 & x_5 & x_4 & x_3 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_{n-2} & x_{n-3} & x_{n-4} & x_{n-5} \\ x_{n-1} & x_{n-2} & x_{n-3} & x_{n-4} \\ x_n & x_{n-1} & x_{n-2} & x_{n-3} \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}, \tag{5.7}$$

so the filter is calculated only where nonzero data are available. It is interesting to note that $(\underset{\sim}{\mathbf{X}}^{\dagger}\underset{\sim}{\mathbf{X}})$ does not necessarily have a unique inverse. For a flat event, all the coefficients in the $\underset{\sim}{\mathbf{X}}$ matrix have equal values, and all the elements of $\underset{\sim}{\mathbf{X}}^{\dagger}\underset{\sim}{\mathbf{X}}$ also have equal values, making $\underset{\sim}{\mathbf{X}}^{\dagger}\underset{\sim}{\mathbf{X}}$ singular.

Expanding equation (5.7) to add a damping condition to equally weight the filter coefficients gives

$$\begin{pmatrix} x_5 \\ x_6 \\ x_7 \\ . \\ . \\ . \\ x_{n-1} \\ x_n \\ x_{n+1} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ x_6 & x_5 & x_4 & x_3 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_{n-2} & x_{n-3} & x_{n-4} & x_{n-5} \\ x_{n-1} & x_{n-2} & x_{n-3} & x_{n-4} \\ x_n & x_{n-1} & x_{n-2} & x_{n-3} \\ \epsilon & 0 & 0 & 0 \\ 0 & \epsilon & 0 & 0 \\ 0 & 0 & \epsilon & 0 \\ 0 & 0 & 0 & \epsilon \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}, \tag{5.8}$$

where ϵ is a small number. Solving this set of equations avoids the tendency that I have seen with Gulunay's method for the largest coefficients to cluster near the output position.

Although these modifications should improve the noise attenuation properties of the complex prediction filter, the biasing effect on the noise attenuation is very small compared to the improvement from using the shorter time-length filter of a typical t-x prediction. A feature of Gulunay's method is, that for noiseless data, prediction is unneeded and concentrating the strongest prediction coefficients near the output has no effect on the results. As the strength of the noise increases, the distribution of the amplitudes of the filter coefficients becomes more even, since adding noise to the input provides the same effect as adding a factor to the main diagonal of $\underset{\sim}{\mathbf{X}}^{\dagger}\underset{\sim}{\mathbf{X}}$.

It might be thought that this noise-dependent coefficient weighting gives f-x prediction a possible advantage over t-x prediction. However, if an irregularity such as a fault exists in the input data, both t-x and f-x predictions will generate identical filters in the noiseless case, since the difference between the input and the predicted data will be smallest for a filter with strong filter coefficients adjacent to the output trace and producing the least smearing of the irregularity. For perfectly regular data with noise added, the f-x prediction filter coefficient distribution will approach that of the t-x prediction filter as the strength of the noise is increased. Thus, f-x prediction always has an effectiveness that is less than or equal to the t-x prediction's effectiveness.

5.1.6 Computer time requirements

The number of filter coefficients controls the computer time required by both of these techniques. Since the applications being considered here are post-stack and generally fast, at least for 2-dimensional cases, the computer time required is generally insignificant. It is interesting to note that while the effective t-x prediction time-domain filter has more coefficients than the individual filters used in f-x prediction, because f-x prediction produces a different filter for each frequency, the total number of filter coefficients is much larger than the number for t-x prediction. Nevertheless, the computer time needed to apply these two processes in the 2-dimensional case is comparable, since each filter calculated by f-x prediction is smaller than the single filter calculated by t-x prediction. Since the time to calculate a filter is proportional to n^3 with the routine I used, where n is the number of filter coefficients, the calculation times used by the two approaches become nearly equal. For the 3-dimensional case, the application of t-x prediction tends to be more costly than f-x prediction, but not tremendously so, since the filter sizes used by t-x prediction can be smaller than those for a comparable f-x prediction. However, as

the size of the t-x prediction filter increases, the processing time increases rapidly. On the other hand, should cost be an issue, Claerbout (1992a) pointed out that the number of iterations in the conjugate-gradient routine used by my t-x prediction can be significantly reduced from its theoretical limit with good results.

5.2 Three-dimensional lateral prediction

Three-dimensional lateral prediction has two important advantages over two-dimensional prediction. The first is that, in a 3-dimensional volume, any output sample is close to more samples in the input than in the 2-dimensional case. Having more nearby samples allows better random noise attenuation. Second, the assumption that events are linear within a 2-dimensional window or 3-dimensional sub-volume can be relaxed. As pointed out by Chase (1992) and Gulunay *et al.* (1993), events that are nonlinear in one direction but linear in another are predicted exactly with a 3-dimensional filter.

Prediction over a 3-dimensional volume can be done with two passes of a 2-dimensional process or with a true 3-dimensional technique. Since two passes of either the t-x or f-x 2-dimensional techniques do not allow processors to take full advantage of the opportunities 3-dimensional data presents, I have extended the 2-dimensional techniques to true 3-dimensional techniques using 3-dimensional filters.

5.2.1 The three-dimensional extension of t-x prediction

Extending the t-x prediction to three dimensions is a simple matter of defining the 3-dimensional convolution and its adjoint, that is, its conjugate transpose. Instead of dividing the input into 2-dimensional windows, the input volume is broken into 3-dimensional sub-volumes. Each of these sub-volumes is used in a separate least-squares problem for calculating a 3-dimensional filter, which is a straightforward extension of the calculation of a 2-dimensional t-x filter. Figure 5.4 shows the configuration of the 3-dimensional filter I used. This configuration is a modification of the more general 3-dimensional filter shown in Claerbout (1992a), page 198. While the 3-dimensional filter might be made symmetrical in space by constraining the filter coefficients across the central element to be equal (Gulunay et al., 1993), I simply applied the filter in multiple directions and averaged the results.

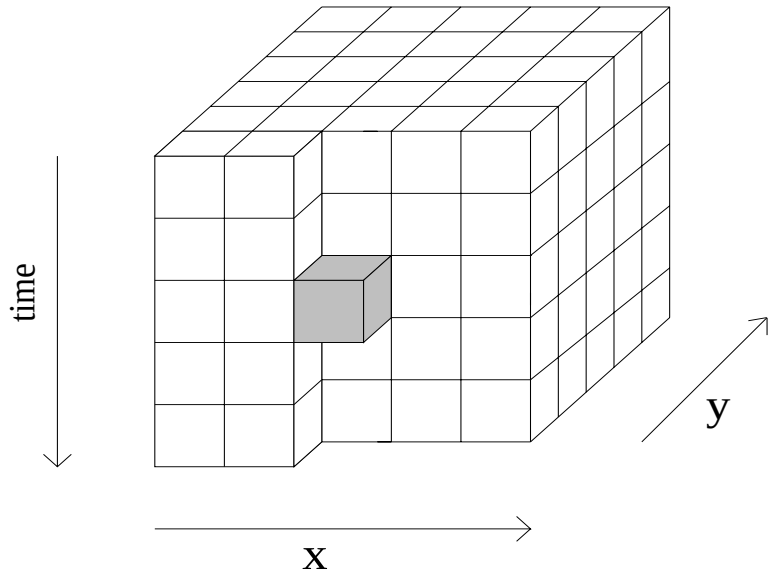

FIG. 5.4. A three-dimensional lateral prediction filter. The darkened cube is the output position. TXFX-DDDfilter [NR]

5.2.2 The three-dimensional extension of f-x prediction

The extension of f-x prediction into three dimensions is more difficult than that of t-x prediction. For each frequency, instead of a prediction along a vector, the prediction of a set of complex numbers within a plane is required. For the examples of 3-dimensional f-x prediction shown here, I computed a complex-valued 2-dimensional filter at each frequency with a conjugate-gradient routine. While other techniques for computing this filter exist, they should produce similar results. The advantage of this approach is that the huge matrix $\underset{\sim}{\mathbf{X}}$ used to describe the 3-dimensional convolution of the filter with the data does not need to be stored, and the inverse of $\underset{\sim}{\mathbf{X}}^{\dagger}\underset{\sim}{\mathbf{X}}$ does not need to be computed, which simplifies the problem significantly (Claerbout, 1992a).

The shape of the 2-dimensional filter used to predict numbers in a 2-dimensional frequency slice has the form

$$\begin{matrix} c_{-2,0} & c_{-2,1} & c_{-2,2} & c_{-2,3} & c_{-2,4} \\ c_{-1,0} & c_{-1,1} & c_{-1,2} & c_{-1,3} & c_{-1,4} \\ 1 & c_{0,1} & c_{0,2} & c_{0,3} & c_{0,4} \\ 0 & c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ 0 & c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \end{matrix} \quad , \tag{5.9}$$

where all the coefficients are complex-valued. The justification for this shape is more fully discussed on page 198 of Claerbout (1992a), but this form may be compared to a horizontal slice through the center of the filter shown in Figure 5.4. If the collection of the filters for all frequencies is Fourier transformed in time, a filter similar to the one shown in Figure 5.4, but extended in time, is formed.

5.2.3 Examples of three-dimensional lateral prediction

An example of the 3-dimensional prediction's ability to predict nonlinear events is shown in Figure 5.5, where the input consists of several dipping layers cut by a fault in the crossline direction, so that events are nonlinear in the inline direction. In the cubes in Figure 5.5 and the figures that follow, the vertical direction is the time axis, the horizontal direction is the inline spatial axis, and the direction running into the page is the crossline spatial axis. The lines on the cubes indicate the position of the slices shown on the faces of the cube. The one-pass 3-dimensional prediction did not smear the fault, because the calculated 3-dimensional filter created a prediction in the crossline direction that preserved the discontinuity in the inline direction. With the two-pass prediction, the inline pass smeared the reflections across the fault. For the noiseless case of Figure 5.5, the f-x prediction is not shown because it gave the same results as the t-x prediction.

The results of applying f-x prediction and t-x prediction to a 3-dimensional land survey provided by ARCO are shown in Figures 5.6 to 5.8 to demonstrate the differences between the two processes. This data set is interesting because it has a significant noise level with fairly flat, predictable events.

For both two-pass applications, the filter size was five elements in the spatial direction. For both 3-dimensional one-pass applications, I employed a filter with five elements in both spatial directions. The t-x prediction used a five-element filter length in the time direction for both one- and two-pass applications. The window sizes were 60 traces in the inline direction, 60 traces in the crossline direction, and 200 samples, or 0.4 seconds in time.

Both the two-pass and the one-pass t-x prediction results in Figures 5.7 and 5.8 show less noise than the corresponding f-x results; otherwise the results are similar. While the one-pass t-x prediction and f-x prediction results are much the same, the t-x prediction output shows somewhat less noise.

The advantage of using 3-dimensional lateral prediction is especially clear in Figures 5.10 and 5.8. Both the one-pass results show significantly less smearing of the

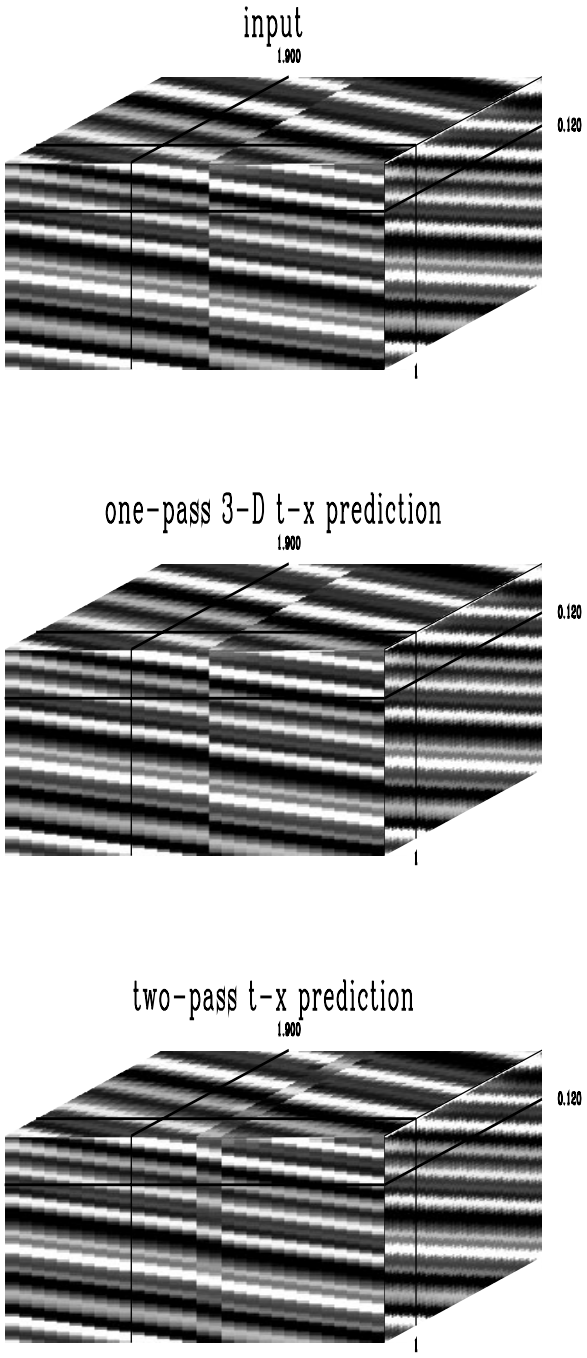

FIG. 5.5. A comparison of smearing with one- and two-pass t-x prediction. The top cube is the input, the middle cube is the result of a one-pass 3-dimensional t-x prediction, and the bottom cube is the result of two passes of one-dimensional t-x prediction in the inline and crossline directions. The one-pass result shows no smearing of the fault; the two-pass result shows a smeared fault image. TXFX-yplanesld [CR]

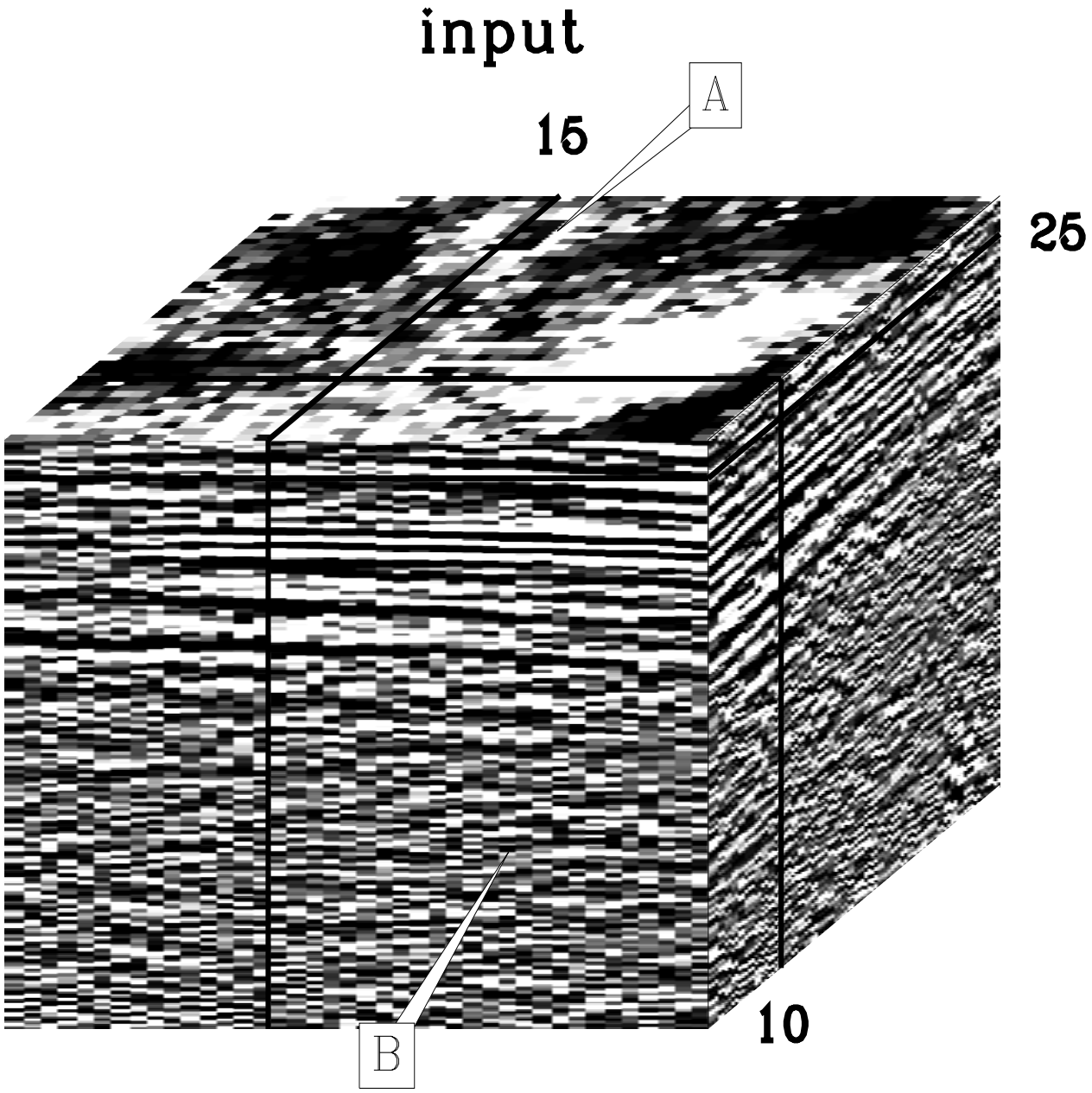

FIG. 5.6. The input to the t-x and f-x predictions. A significant amount of noise is seen here. In this figure and the ones that follow, the vertical axis is time and the horizontal axes are space. TXFX-Arcoorig [CR]

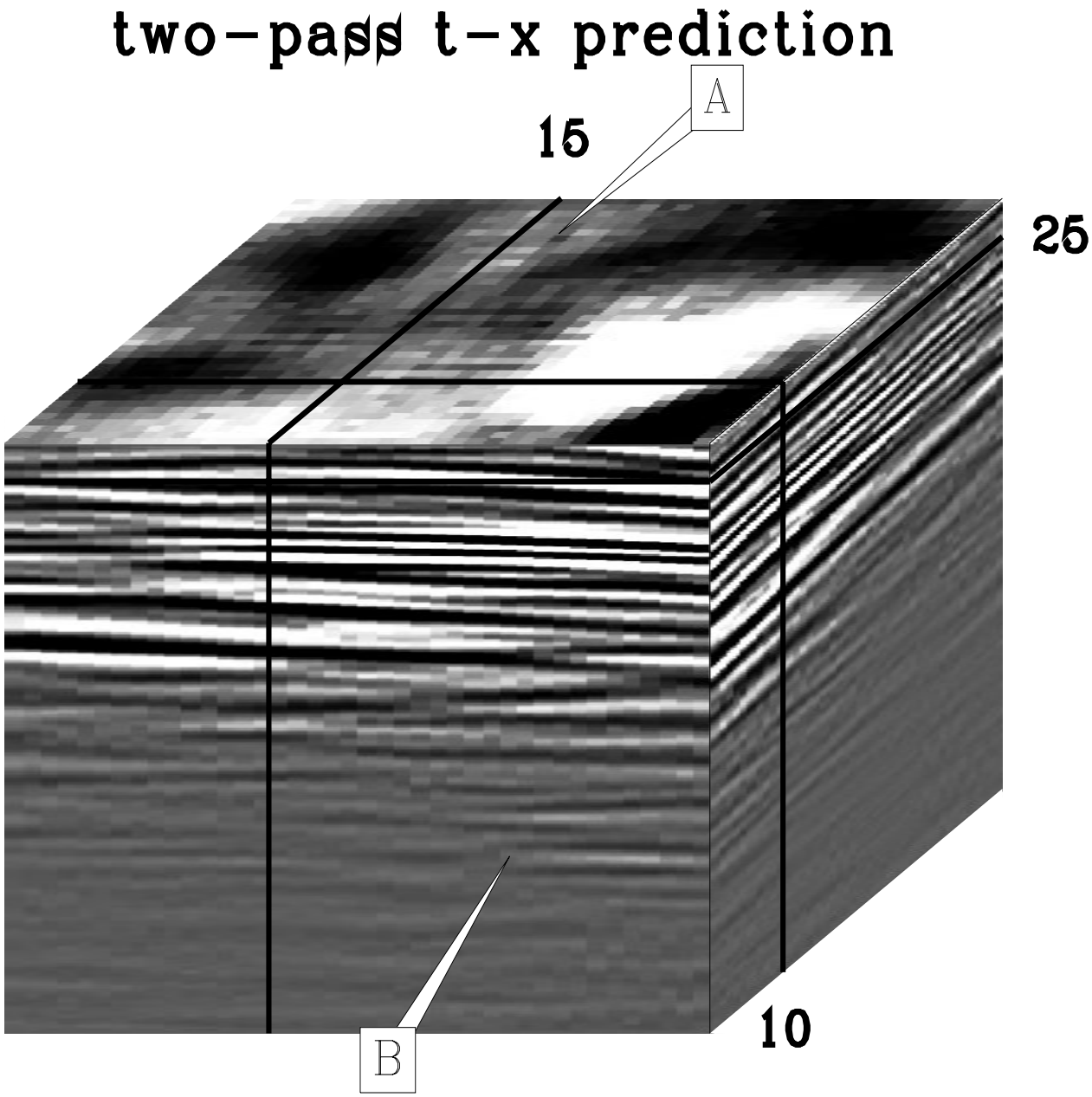

FIG. 5.7. The result of two passes of 2-dimensional t-x prediction processing. While this result is an improvement over the input, some of the details are seen to be lost when compared to the 3-dimensional t-x prediction. More noise has been attenuated than with the two-pass f-x prediction. TXFX-Arcotx2d [CR]

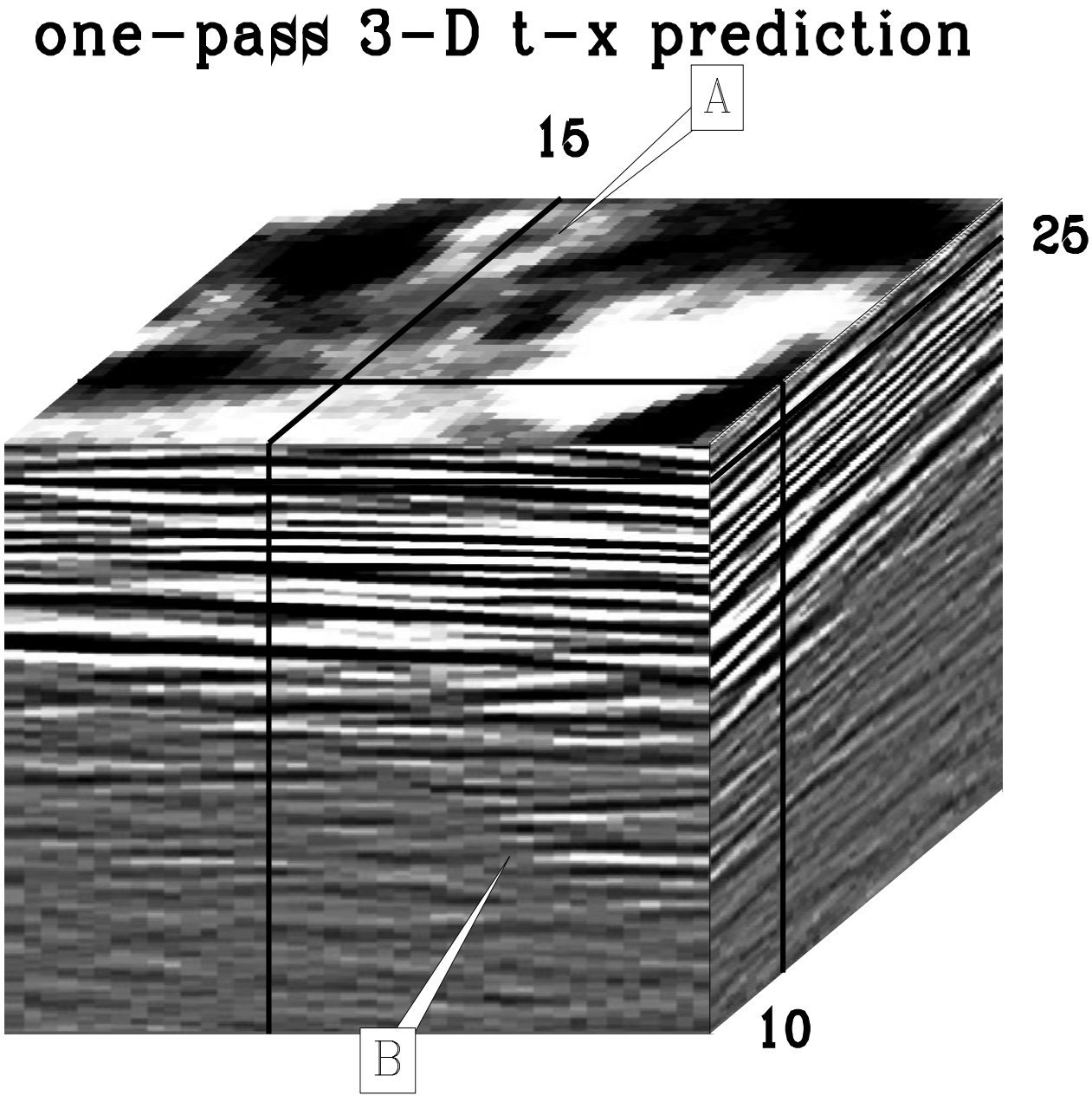

FIG. 5.8. The result of 3-dimensional t-x prediction processing. This result preserves the details lost in the two-pass t-x prediction. TXFX-Arcotx3d [CR]

structure. On the top faces of the cubes in Figures 5.10 and 5.8, the one-pass results appear clean and reasonable, whereas the two-pass results show smearing along the inline and crossline directions. An example of the smearing of the detail can be seen at point A of these figures, where a small doughnut-shaped feature is badly smeared in both the two-pass results. The front face of the cubes in Figures 5.9 to 5.8 are significantly different for the one- and two-pass results; with the one-pass results showing much more detail. The features at point B in the figures once again demonstrate the loss of detail. Although the differences between the 3-dimensional t-x and f-x results in Figures 5.10 and 5.8 are less than those between the 2-dimensional t-x and f-x results in Figure 5.3, the results of the 3-dimensional t-x prediction appear slightly cleaner than those of the 3-dimensional f-x prediction.

5.3 Conclusions

Since f-x prediction has been shown to be equivalent to a t-x prediction with a long time length, it may not be surprising that t-x prediction generally produces results similar to those of f-x prediction. Although both these techniques work equally well in low noise cases, t-x prediction provides better random noise reduction than the older f-x prediction technique in the presence of moderate- to high-amplitude noise. The t-x prediction also avoids the generation of false events in the presence of parallel events when using f-x prediction, at least for parallel events with spacings wider than the filter length in time. The advantages of t-x prediction are the result of its ability to control the length of the prediction filters in time. Because t-x prediction has a shorter effective filter length in time than f-x prediction, t-x prediction passes significantly less random noise than f-x prediction. While Gulunay's f-x prediction biases the prediction toward the traces nearest to the output point, allowing more noise to be passed, this bias appears unimportant when compared to the problem with the length of the effective filter in time.

3-dimensional prediction allows improved noise attenuation because more samples are used to make predictions. 3-dimensional prediction also relaxes the requirement that events be linear. Comparisons of one- and two-pass predictions on a land data set show that the one-pass results retain significantly more detail than the two-pass results. In both one- and two-pass predictions, the f-x prediction passes more random noise than the t-x prediction.

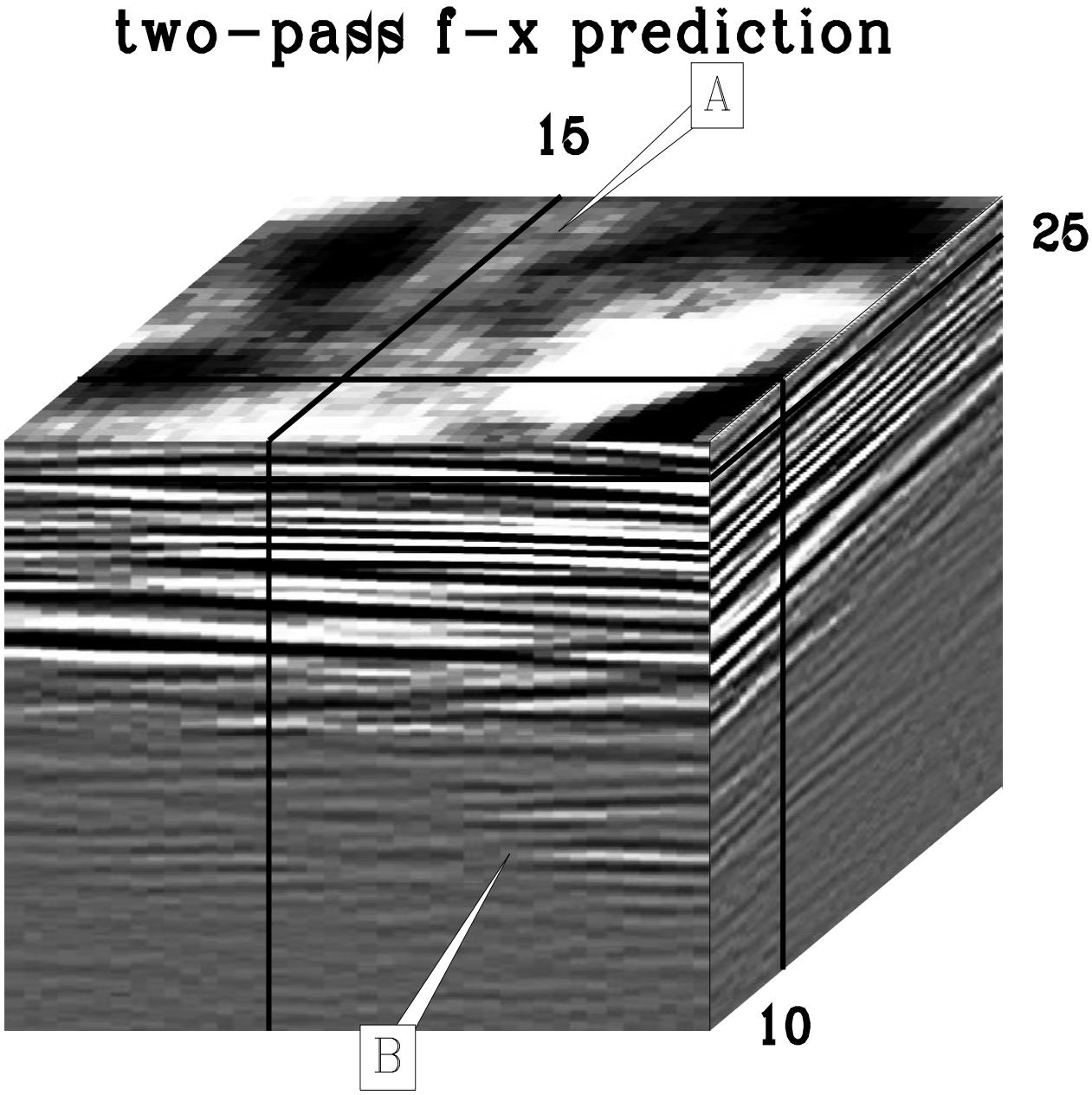

FIG. 5.9. The result of two passes of 2-dimensional f-x prediction processing. While this result is an improvement over the input, some of the details are seen to be lost when compared to the 3-dimensional f-x prediction. TXFX-Arcofx2d [CR]

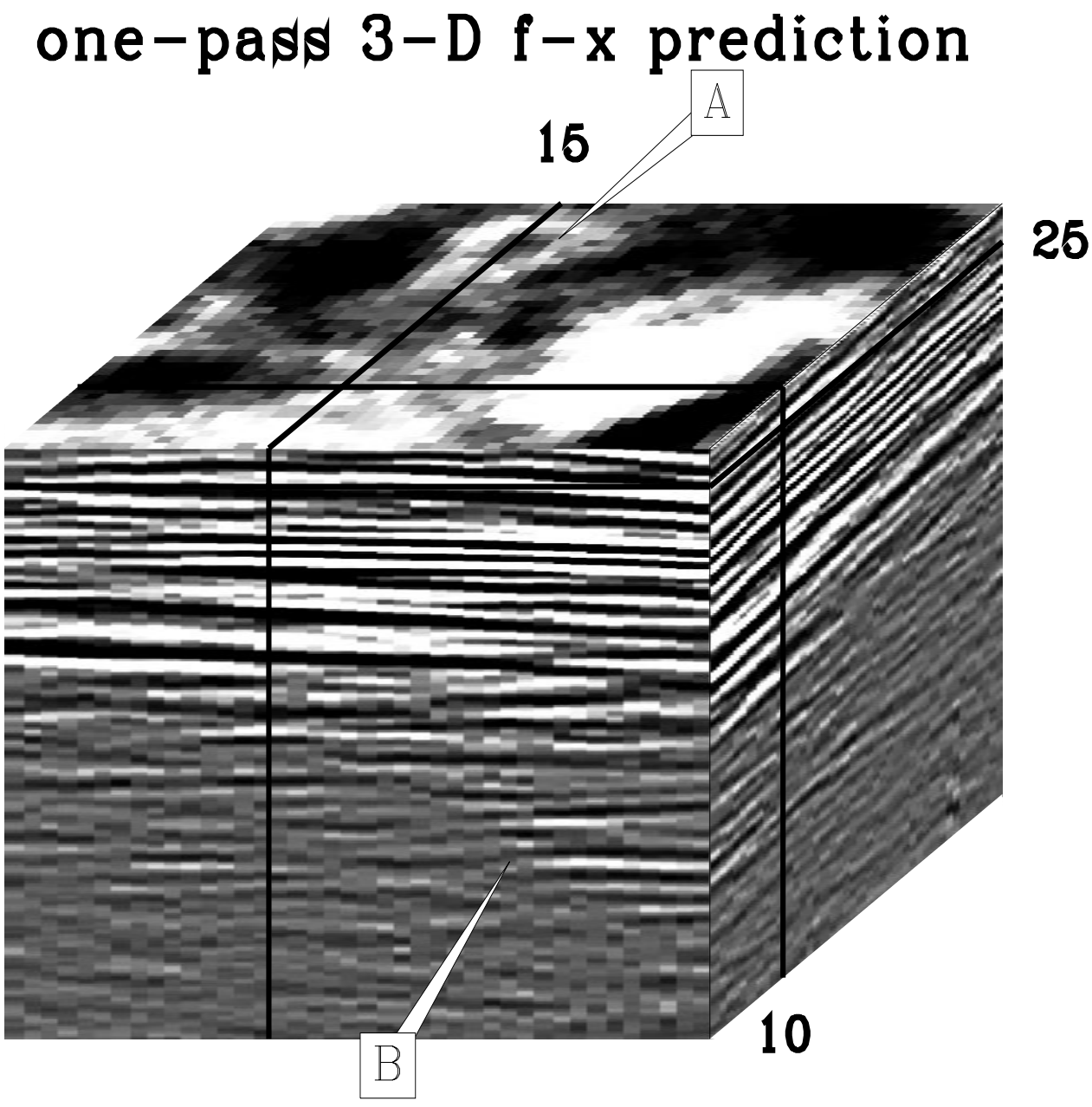

FIG. 5.10. The result of 3-dimensional f-x prediction processing. This result preserves the details lost in the two-pass f-x prediction. TXFX-Arcofx3d [CR]

In the next chapter, the generation of spurious events is examined in more detail. In chapter 7, I will examine two more shortcomings of f-x and t-x prediction, that of the amplitude loss in the signal and that of the filter response to the noise being left in the signal. To avoid these difficulties, a method of extending t-x prediction is presented in chapter 7.

Chapter 6

Spurious event generation with f-x and t-x prediction filtering

In the previous chapter, the f-x prediction technique and the t-x prediction technique were compared and contrasted. It was shown that an f-x prediction is equivalent to a t-x prediction with a very long filter length in time. This filter, with its long length in time, generates spurious events from parallel events embedded in noise on the input. Even with a short filter length in time, the wavelet of events may be corrupted, since a wavelet on an event appears as a series of parallel events. This chapter compares the action of these two prediction techniques, and shows examples of spurious event generation for both widely spaced events and for wavelets.

6.1 F-x prediction expressed as t-x prediction

As shown in Abma and Claerbout (1993) and in chapter 5 of this thesis, the similarities and differences between f-x prediction and t-x prediction are explained by comparing the form of the t-x prediction filter and the form of the effective t-x domain filter created by f-x prediction. This comparison, which is also shown in chapter 5, is quickly reviewed here.

For each frequency, f-x prediction generates a prediction filter over the samples in space. The individual filters calculated by the f-x prediction have the form

$$1 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \ , \tag{6.1}$$

where a_1, a_2, a_3, and a_4 are complex numbers. Collecting these filters for all frequencies and presenting them in the frequency-space domain produces the following composite filter:

$$\begin{matrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{-3,1} & a_{-3,2} & a_{-3,3} & a_{-3,4} \\ 1 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\ 1 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\ 1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ 1 & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ 1 & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ 1 & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{matrix} \quad . \tag{6.2}$$

Transforming this filter from the frequency-space domain into the time-space domain gives the effective t-x domain filter,

$$\begin{matrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & b_{-3,1} & b_{-3,2} & b_{-3,3} & b_{-3,4} \\ 0 & b_{-2,1} & b_{-2,2} & b_{-2,3} & b_{-2,4} \\ 0 & b_{-1,1} & b_{-1,2} & b_{-1,3} & b_{-1,4} \\ 1 & b_{0,1} & b_{0,2} & b_{0,3} & b_{0,4} \\ 0 & b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\ 0 & b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\ 0 & b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{matrix} \quad , \tag{6.3}$$

where the number of rows is the number of time samples in the data being considered, i.e. the design window. The Fourier transform has converted the first column of 1s into a single 1 at the output position, creating a time-space filter.

A typical t-x prediction filter appears as:

$$\begin{matrix} 0 & b_{-1,1} & b_{-1,2} & b_{-1,3} & b_{-1,4} \\ 1 & b_{0,1} & b_{0,2} & b_{0,3} & b_{0,4} \\ 0 & b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \end{matrix} \quad . \tag{6.4}$$

This filter has the same form as the f-x prediction filter shown in (6.3) but with the

number of rows greatly decreased. The action of an f-x prediction filter can be seen to be the action of a t-x prediction filter with a very long time-length.

6.2 Spurious event generation with f-x prediction

This section reviews the discussion in chapter 5, where it was shown that one effect of the long filter length in time for f-x prediction is the possible generation of false events. If strong parallel events are embedded in a background of random noise, f-x prediction generates weak but significant events parallel to the original events. While both t-x and f-x techniques tend to line up noise with strong events, f-x prediction actually generates new events with its long filter length in time. These spurious events occur because parallel events in a random noise background produce an f-x filter where the prediction of one strong event is influenced by the presence of other parallel events. An event corrupted by noise can have its prediction improved by using information from a nearby parallel event if the filter length in time is long enough to take advantage of the extra information. The t-x prediction is much less likely to be affected by the influence of widely separated events because of its short filter length in time.

The strength of the spurious events depends on the ratio of the signal to the background noise and the number of parallel events that contribute to the output sample's prediction. With no noise, no spurious events are generated. If there is no signal, or a very weak signal with respect to the noise, no prediction is done and no spurious events are generated. If many events parallel to the signal exist, each event contributes little to the prediction, and the spurious events generated by each of these parallel events will be too small to notice in most practical settings.

An example of the spurious events generated by f-x prediction can be seen in Figure 6.1. In this case, two flat events are immersed in noise on the left side of the displays, and the right side of the display is left noise free to show the response of the filter. For both f-x and t-x predictions, only one design window is used. With f-x prediction, events widely spaced in time can be used to predict any output point. For the f-x prediction in Figure 6.1, the noise on the left side of the input allows the predictions to be made using input from both events, so the equivalent t-x filter has significant coefficients far in time from the output point. When this filter is applied to clean data on the right side, spurious events generated by these widely separated coefficients are seen. In the f-x prediction result,

strong events are generated above and below the two original events, and weak events line up with the original events on the right side. The t-x prediction results do not show these erroneous events. While the f-x prediction may be modified to eliminate this problem by constraining the filter coefficients to be smooth in frequency, the t-x prediction does not generate spurious events since its time length is more naturally controlled.

The false events generated by the f-x prediction generally appear to be weak. To give a better idea of the amplitudes of the false events, the results of the f-x prediction seen in the Figure 6.1 are displayed as an elevation plot in Figure 6.2. In real data with an even distribution of noise, these spurious events are unlikely to be mistaken for real events, since the remaining noise should hide the errors.

Finally, to demonstrate that the f-x prediction and the t-x prediction show the same effects when a long filter length in time is used and that the generated events are not Fourier domain wraparound, Figure 6.3 shows a t-x prediction filter with a time-length of 50 samples. Notice that the results are similar.

A typical f-x prediction program has an advantage over a similar t-x prediction program because the filter length in time is fixed and does not need to be specified. Therefore, this length cannot be accidentally made too short. The cost of not requiring this parameter is creating the risk of generating false events and by passing more noise (Abma and Claerbout, 1993).

6.3 Wavelet distortion

The previous examples show spurious events generated at widely spaced times because parallel events affect each other's predictions. These widely spaced events may be removed by shortening the filter length in time. Closely spaced parallel events will produce spurious events with little separation in time. These appear as distortions to the wavelet.

A single event that is extended in time, such as a reflection with a wavelet convolved with it, suffers distortions, as seen in Figures 6.4 to 6.6. Figure 6.4 shows an example of lateral prediction with a single event. The original event extending over three samples in time is shown in Figure 6.5. When a prediction filter was applied to the data, a trace from the noise-free side of the data was extracted and is shown in Figure 6.6. While Figure 6.6 shows the t-x prediction result, f-x prediction produces similar effects.

Notice that the event in Figure 6.6 appears similar to the event in Figure 6.5, but is

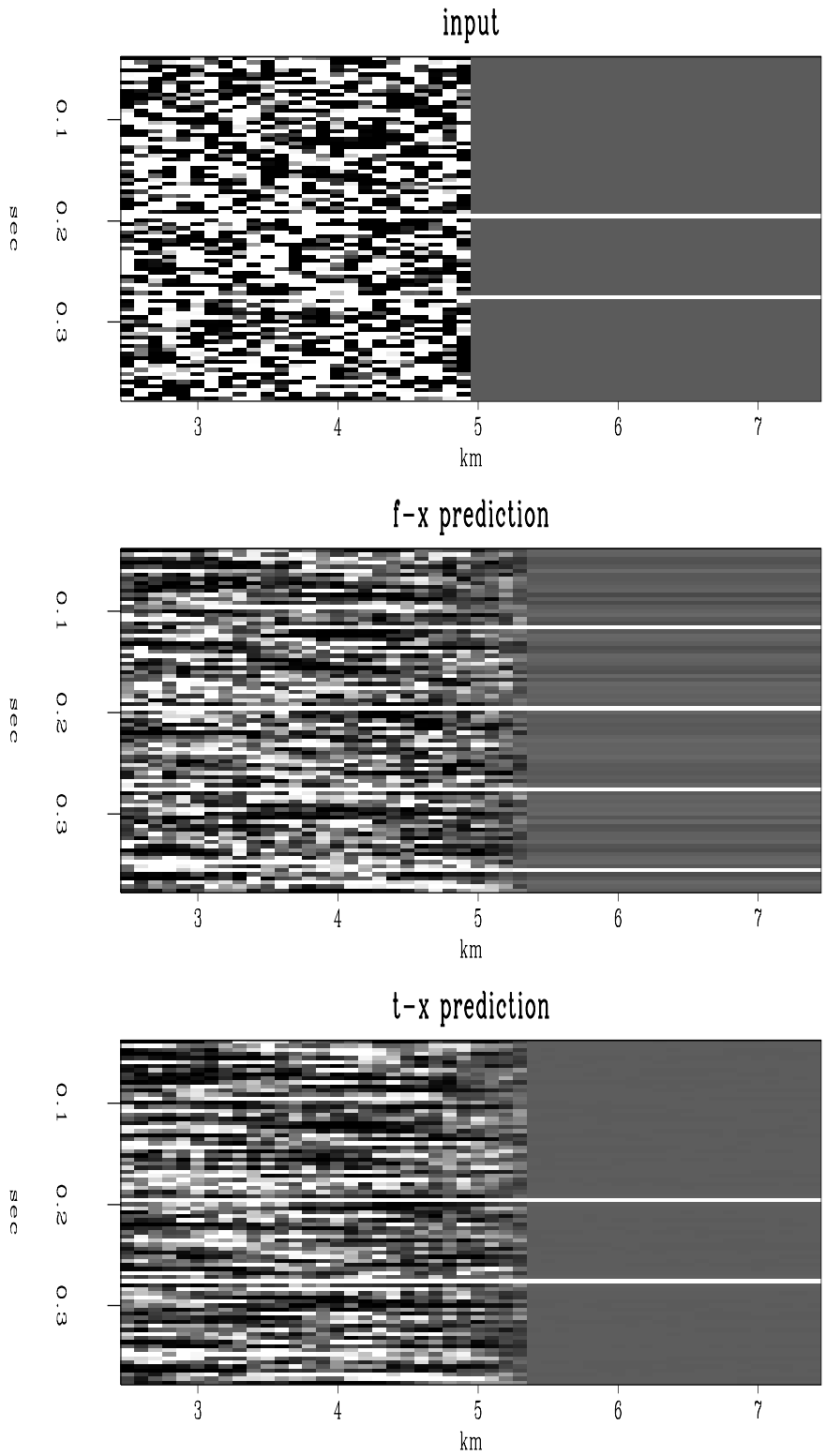

FIG. 6.1. An example of the creation of false events with f-x prediction and the corresponding result from t-x prediction. spurious-two [R]

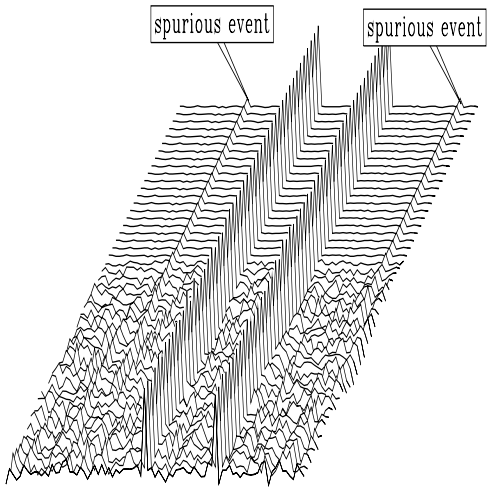

FIG. 6.2. The relative amplitudes of the false events to the original events for the f-x prediction of Figure 1. spurious-twoh [R]

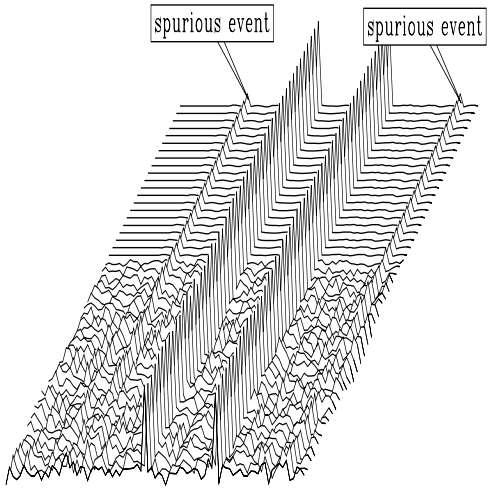

FIG. 6.3. The relative amplitudes of the false events to the original events for a t-x prediction with an extremely long filter length in time. spurious-twomdlong [R]

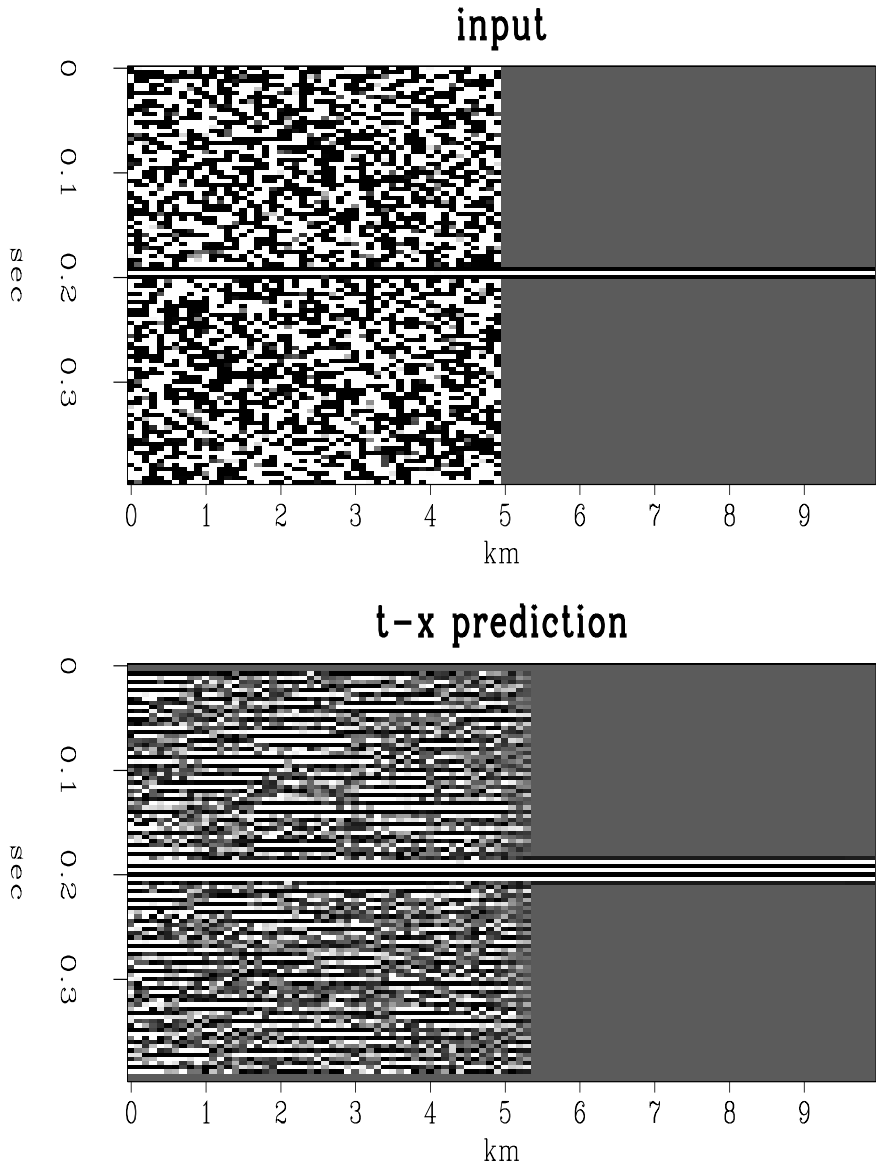

FIG. 6.4. An example of lateral prediction with a single event. spurious-one [R]

extended in time. As the level of the noise is decreased, the spreading of the wavelet in time will decrease. There will be a trade-off between noise attenuation and resolution in time that will depend on the level of the noise. Normally this trade-off is well worth the small sacrifice of resolution, but the user should be aware that a trade-off is taking place.

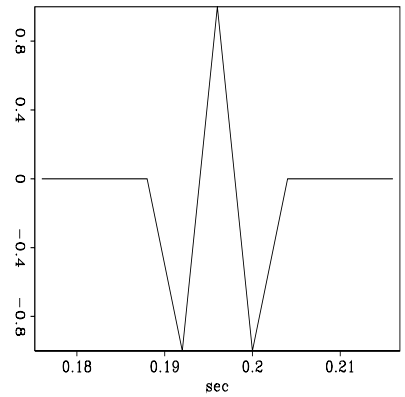

FIG. 6.5. The original trace before prediction filtering. spurious-smultspike [R]

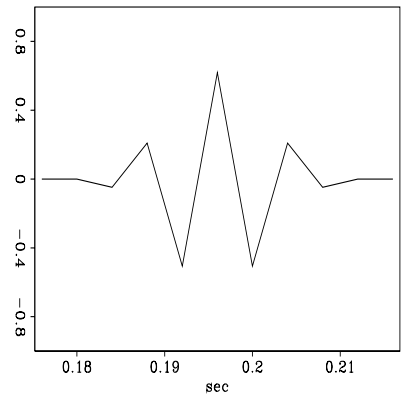

FIG. 6.6. The trace after prediction filtering. spurious-smultmda [R]

6.4 Discussion

If a large number of closely spaced parallel events buried in noise fall in a window, the calculated filter will consist of the central main spike plus small contributions from all parallel events. With a large number of parallel events, the contribution of any one event parallel to the event being predicted will be small. For example, data in the Gulf of Mexico generally has many closely spaced reflections. If a long prediction filter in time is used

with this data, events far from the output will contribute to the output point, but since each contribution is small, the spurious events generated by the filter are unlikely to be a problem. In an area with only a few reflections that are widely spaced, parallel events may generate significant spurious events as seen in Figures 6.2 and 6.3.

These demonstrations should not be construed as suggesting that lateral predictions produce unreliable answers. The spurious events are low amplitude and will generally be hidden by the remaining noise. Furthermore, if many parallel events exist within a design window, the spurious events generated are weakened and distributed over time so that the effect is reduced. Figure 6.7 shows an f-x prediction over a dataset with many parallel events. While many of the expected spurious events are very weak, a spurious event close to the original events on the left is almost as strong as those seen in Figure 6.2. I suspect that these phenomena will cause problems only in cases where a few isolated reflections appear in a background of noise, and where weak reflections are being sought. In this case, a t-x prediction rather than an f-x prediction should be used to avoid generating events separated from the original events by significant times. Spurious events that appear as a change of wavelet are less likely to be interpreted as new events, but may interfere with the interpretation of subtle stratigraphic changes.

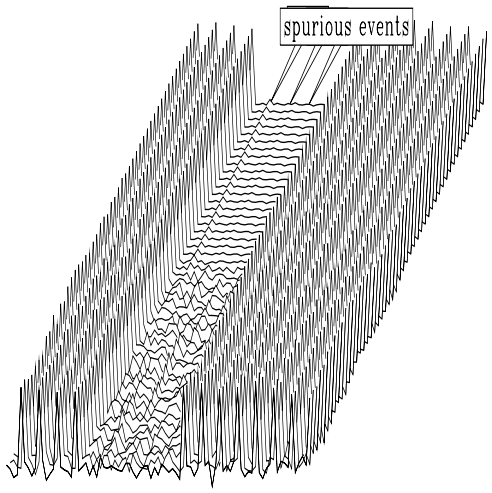

FIG. 6.7. For f-x prediction applied to a dataset with many parallel events, spurious events generally have lower amplitudes than for datasets with fewer events. spurious-manyh [R]

6.5 Conclusions

While f-x prediction generally produces results similar to those of t-x prediction, f-x prediction may create spurious events when parallel events occur in the presence of noise. These spurious events are caused by the long filter length in time of the effective time-domain filter of the f-x filter. The t-x prediction avoids the generation of false events in the presence of strong parallel events by limiting the length of the filter in time. With both f-x and t-x predictions, wavelets may be distorted, since events with wavelets may be considered as closely spaced sets of parallel events. In typical seismic data, this problem is unlikely to be a serious problem, but if a few isolated reflections appear in a field of noise, weak events may be generated that will confuse the interpretation. In the next chapter, a solution to this shortcoming of f-x and t-x prediction will be presented.

Chapter 7

Random noise removal enhanced by inversion

In the previous chapter, the generation of spurious events by t-x and f-x prediction was shown to occur when a signal occured in a background of noise. Soubaras (1994) also pointed out that the amplitudes of signals may be reduced by these prediction-filtering techniques. Both these effects are examples of how t-x and f-x prediction-filtering methods may break down in the presence of noise. These breakdowns are partially caused by the corruption of the prediction filter by noise. It may also be seen that the response of the filter to the noise can also contribute to these breakdowns when it overwhelms weak reflections.

These problems can be overcome by posing the noise removal as an inversion problem. This inversion removes the filter response from the calculated noise; plus, the inversion allows the filter to be recalculated without the noise corruption. The recalculated filter allows improved signal prediction.

In this chapter, I will show how the noise removal may be posed as an inversion problem and how the noise estimate from prediction filtering is used to increase the accuracy and speed of the solver. The combination of the inversion and the recalculation of the filter will be shown to preserve the amplitude of reflectors and to reduce spurious events generated by the prediction filtering (Abma, 1994). The process is demonstrated on synthetic and real data.

7.1 Shortcomings of prediction filtering

High-amplitude noise produces flaws in prediction-filtering techniques such as t-x and f-x prediction filtering. One flaw is the reduction of reflection amplitudes. Another is the generation of spurious events, as seen in the previous chapter and in Abma (1994). Both these errors are due to the corruption of the signal-prediction filter by the noise in the data from which the filter is calculated.

Another, less obvious, flaw in prediction filtering is that, even with a filter that perfectly predicts the signal, the output of this filtering does not perfectly separate the signal and noise. To demonstrate this, I define $\mathbf{d}$ as the available data, $\mathbf{s}$ as the signal, and $\mathbf{n}$ as the noise. The relationship between the data, the signal, and the noise is defined to be $\mathbf{d} = \mathbf{s} + \mathbf{n}$. Although the prediction of the signal could be stated otherwise, the prediction is done here with a signal annihilation filter $\underset{\sim}{\mathbf{S}}$. The filter $\underset{\sim}{\mathbf{S}}$ is a purely lateral 2- or 3-dimensional filter as discussed in chapter 5. If the signal $\mathbf{s}$ is prefectly predictable, the filter $\underset{\sim}{\mathbf{S}}$ completely removes the signal so that $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$. In fact, only an approximate signal annihilation filter is generally available so that $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$, but to simplify the following discussion, $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$ will be assumed for now. When the data $\mathbf{d}$ is filtered by the exact signal annihilation filter, the result is $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{s} + \underset{\sim}{\mathbf{S}}\mathbf{n}$, which becomes $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{n}$, since $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$. Since prediction filtering defines the noise as $\underset{\sim}{\mathbf{S}}\mathbf{d}$, a filtered version of the noise $\underset{\sim}{\mathbf{S}}\mathbf{n}$ is obtained from the prediction filtering instead of the actual noise $\mathbf{n}$.

Prediction filtering makes the assumption that the noise $\mathbf{n}$ is unaffected by the signal annihilation filter $\underset{\sim}{\mathbf{S}}$. The difference between $\underset{\sim}{\mathbf{S}}\mathbf{n}$ and $\mathbf{n}$ may also be seen as an inconsistency between definitions of the noise in the expressions $\mathbf{n} = \mathbf{d} - \mathbf{s}$ and $\mathbf{n} = \underset{\sim}{\mathbf{S}}\mathbf{d}$ (Soubaras, 1994). For weak noise and large filters, the assumption that the noise $\mathbf{n}$ is unaffected by the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ is reasonable. For strong noise and short filters, the response of the noise to the filter is important. Although prediction filters may be made as large as desired, Chapters 5 and 6 show that large filters allow more noise to pass into the signal and that filters that are large along the time axis tend to create spurious events. For very large noises, the filter response is alway significant.

An example of the filter response to noise is shown in Figure 7.1. In the original data seen in this figure, the signal is a flat event and the noise is an isolated spike. Since the prediction filter is applied in two directions, the response of the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ can be seen on both sides of the spike's position in the prediction-filter result.

The prediction-filtering result also shows a small amplitude loss in the flat event. The corruption of the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ by the spike causes this amplitude loss.

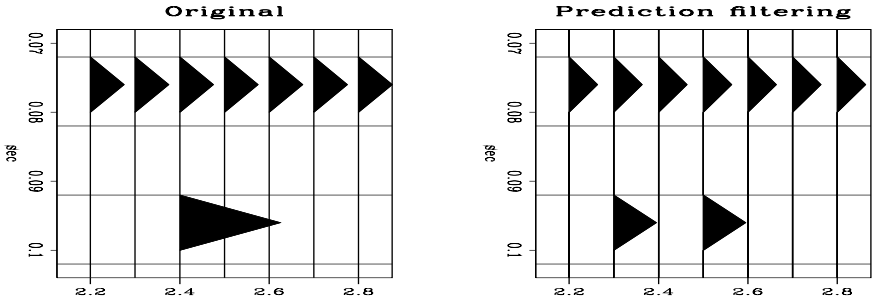

FIG. 7.1. The action of a prediction filter on a flat layer and a spike. rinver-respn [R]

Getting a more accurate calculation of the noise requires solving the expression $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{n}$ when $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$. If the exact signal annihilation filter is not available and $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$, the noise must be solved for from the regression $\underset{\sim}{\mathbf{S}}\mathbf{n} \approx \underset{\sim}{\mathbf{S}}\mathbf{d}$. Similar expressions have been used for noise removal by Claerbout and Abma (1994) and Abma and Claerbout (1994). In the next section I will present a solution to $\underset{\sim}{\mathbf{S}}\mathbf{n} \approx \underset{\sim}{\mathbf{S}}\mathbf{d}$.

7.2 Noise estimation by inversion

As described in chapter 3, a structure for creating an inverse may be

$$\mathbf{0} \approx \underset{\sim}{\mathbf{W}}(\underset{\sim}{\mathbf{L}}\mathbf{m} - \mathbf{d}) \tag{7.1}$$

$$\mathbf{0} \approx \epsilon \underset{\sim}{\mathbf{A}}\mathbf{m}, \tag{7.2}$$

where $\underset{\sim}{\mathbf{W}}$, $\underset{\sim}{\mathbf{L}}$, and $\underset{\sim}{\mathbf{A}}$ are linear operators, and $\mathbf{m}$ and $\mathbf{d}$ correspond to a model and to the data. The value of ϵ is used to weight the relative importance of (7.1) and (7.2). Replacing $\underset{\sim}{\mathbf{W}}$ with the signal annihilation filter $\underset{\sim}{\mathbf{S}}$, $\underset{\sim}{\mathbf{L}}$ with $\underset{\sim}{\mathbf{I}}$, the identity matrix, and ignoring $\mathbf{0} \approx \epsilon \underset{\sim}{\mathbf{A}}\mathbf{m}$ for the moment gives an expression $\underset{\sim}{\mathbf{S}}\mathbf{d} \approx \underset{\sim}{\mathbf{S}}\mathbf{n}$ for calculating the noise from the data given the signal annihilation filter $\underset{\sim}{\mathbf{S}}$. The expression $\underset{\sim}{\mathbf{S}}\mathbf{d} \approx \underset{\sim}{\mathbf{S}}\mathbf{n}$ is not useful in itself for calculating the noise $\mathbf{n}$, since the filter $\underset{\sim}{\mathbf{S}}$ is not perfect and is unlikely to completely annihilate the signal to the point where the inversion for $\mathbf{n}$ could not restore it. Without additional constraints, the obvious solution to $\underset{\sim}{\mathbf{S}}\mathbf{d} \approx \underset{\sim}{\mathbf{S}}\mathbf{n}$ is $\mathbf{d} = \mathbf{n}$. In practice, I have found that, although the filter $\underset{\sim}{\mathbf{S}}$ could attenuate the signal significantly, a simple inversion of $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{n}$ for $\mathbf{n}$ restores much of the signal into the calculated noise $\mathbf{n}$. What is needed is a constraint to replace $\mathbf{0} \approx \epsilon \underset{\sim}{\mathbf{A}}\mathbf{m}$ in system (7.2).

The constraint used here to keep signal out of the calculated noise is that the noise is approximately the noise estimated from prediction filtering $\underset{\sim}{\mathbf{S}}\mathbf{d}$. This is a reasonable approximation, since $\underset{\sim}{\mathbf{S}}\mathbf{d}$ should be somewhat close to the actual noise. The difference between the actual noise $\mathbf{n}$ and the approximated noise $S\mathbf{d}$ should be fairly small and involves only the response of the noise to the filter $\underset{\sim}{\mathbf{S}}$. The approximation is weighted as $\epsilon\mathbf{n} \approx \epsilon\underset{\sim}{\mathbf{S}}\mathbf{d}$. The value for ϵ may be changed to account for the signal-to-noise ratio of the data.

The system of regressions to be solved is now

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \epsilon\underset{\sim}{\mathbf{S}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \mathbf{n}. \tag{7.3}$$

The results of solving this system are referred to as inversion prediction in the following discussion to distinguish it from prediction filtering.

Since this system estimates $\mathbf{n}$ from the approximation $\underset{\sim}{\mathbf{S}}\mathbf{d}$, it is reasonable to initialize $\mathbf{n}$ to $\underset{\sim}{\mathbf{S}}\mathbf{d}$ before entering the iterative solver. Another reason for initializing $\mathbf{n}$ to $\underset{\sim}{\mathbf{S}}\mathbf{d}$ is that the filter $\underset{\sim}{\mathbf{S}}$ is generally small and will pass only a limited range of spatial and temporal frequencies. In the case of a spike in the data, inversion for the noise with a small filter does not allow the complete restoration of the spike. Because the noise is expected to be almost white and in some cases dominated by spikes, initializing $\mathbf{n}$ to $\underset{\sim}{\mathbf{S}}\mathbf{d}$ improves the calculation of $\mathbf{n}$ and reduces the number of iterations needed.

Equation (7.3) expressed as a minimization of the residual $\mathbf{r}$ is

$$\mathbf{r} = \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \mathbf{n} - \begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \epsilon\underset{\sim}{\mathbf{S}}\mathbf{d} \end{pmatrix}. \tag{7.4}$$

Initializing $\mathbf{n}$ to $\underset{\sim}{\mathbf{S}}\mathbf{d}$ involves adding

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \tag{7.5}$$

to the right-hand side of equation (7.4) to produce, with some simplification,

$$\mathbf{r} = \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \mathbf{n} + \begin{pmatrix} \underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{S}}\mathbf{d} - \underset{\sim}{\mathbf{S}}\mathbf{d} \\ 0 \end{pmatrix}. \tag{7.6}$$

Since the iterative solver just updates $\mathbf{n}$ without regard to the initial value (Claerbout, 1995), the value of $\mathbf{n}$ in this equation may be considered as the change of the calculated

noise from the first estimate of the noise $\underset{\sim}{\mathbf{S}}\mathbf{d}$. This may be expressed as

$$\mathbf{r} = \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \Delta\mathbf{n} + \begin{pmatrix} \underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{S}}\mathbf{d} - \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \mathbf{0} \end{pmatrix}. \tag{7.7}$$

This is the effective system of regressions that is implimented in this chapter.

The results of inversion prediction are sensitive to the value of ϵ. At the moment, the optimum value of ϵ is uncertain. It would seem that ϵ should decrease as the signal-to-noise ratio decreases, since the difference between the actual noise $\mathbf{n}$ and the estimated noise $\underset{\sim}{\mathbf{S}}\mathbf{d}$ is larger. However, in the presence of strong noise, the larger ϵ is, the more stable the inversion should be. If ϵ is relatively large, around 1.0, the amplitudes of the reflections are preserved and spurious events are somewhat suppressed. As ϵ gets very large, the result approaches the prediction filter result. When ϵ gets small, the amplitudes of the reflectors are attenuated, since the signal filter $\underset{\sim}{\mathbf{S}}$ does not perfectly annihilate the signal before the inversion. For small ϵ, the spurious events tend to return also. The best value of ϵ appears to be different for samples with Gaussian noise than for samples with uniformly distributed noise. For most work, it appears that good values of ϵ vary from 0.1 to 3.0. Small values of ϵ remove background noise, but seem to introduce organized noise into the calculated signal. For the real data examined, the background noise increases as ϵ increases, and the continuity of the data increases as ϵ decreases. Further work is needed to determine how the strength and type of noise affects the value of ϵ.

An example of the difference between prediction filtering and inversion prediction is seen in Figure 7.2. The filter $\underset{\sim}{\mathbf{S}}$ is calculated from the data to predict the flat event. When $\underset{\sim}{\mathbf{S}}$ is applied to the spike, the filter response can be seen in the prediction-filter result. The inversion prediction result has effectively eliminated the filter response.

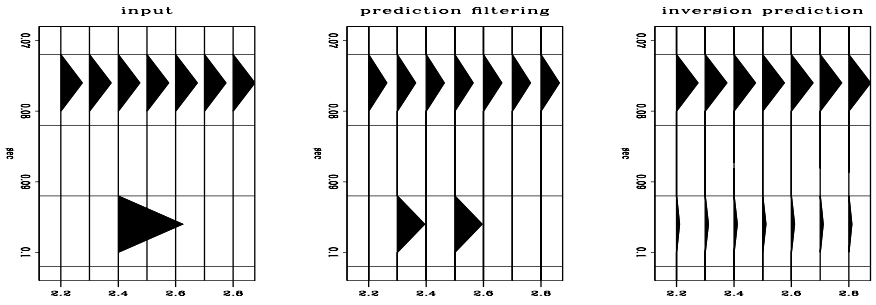

FIG. 7.2. A comparison of the action of a t-x prediction filter and an inversion prediction on a spike. rinver-onespikea [R]

7.3 Improving the signal-prediction filter

In the previous discussion, it was assumed that the signal filter $\underset{\sim}{\mathbf{S}}$ completely annihilates the signal, that is $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$. In reality, imperfect filters are derived from noisy data. For prediction filtering, the filters are derived from the least-squares solutions to the expression $\underset{\sim}{\mathbf{S}}\mathbf{d} = \mathbf{0}$. Since the data $\mathbf{d}$ contains noise, rather than getting an $\underset{\sim}{\mathbf{S}}$ where $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$, we must contend with an imperfect $\underset{\sim}{\mathbf{S}}$ such that $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$. This section shows how a better $\underset{\sim}{\mathbf{S}}$ may be calculated by reducing the influence of the noise.

The presence of noise in the estimation of the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ affects the calculation of the estimated signal in two ways. First, spurious events may be generated. These events may be widely separated in f-x prediction or may be seen as distortions of an event's wavelet. The cause of these distortions is discussed in chapter 6. Second, the amplitudes of the reflectors in the calculated signal are reduced due to the imperfect prediction. As the strength of the noise increases, the more corrupted the filter becomes and the more the reflectors are attenuated.

To improve the calculation of the filter $\underset{\sim}{\mathbf{S}}$, $\underset{\sim}{\mathbf{S}}$ should be derived from the signal $\mathbf{s}$ instead of the data $\mathbf{d}$. Since the actual signal is unavailable, I use the inversion prediction result from equation (7.3) to get an estimate of the signal. Although the signal estimate is not perfect because $\underset{\sim}{\mathbf{S}}$ is imperfect, this signal estimate can be used to create a new $\underset{\sim}{\mathbf{S}}$ that is less affected by the noise. The process of calculating the signal, then getting a new signal annihilation filter, may be iterated as often as desired.

At this point, you might wonder why we should bother with the inversion when a cleaned-up signal may be obtained from prediction filtering. The inversion is more expensive than prediction filtering and might be avoided until a more perfect filter $\underset{\sim}{\mathbf{S}}$ is available. Unfortunately, the signal annihilation filter calculated from the signal derived from prediction filtering will be exactly the same as the original filter calculated from the data. The residual $\mathbf{r}$ in the filter calculation expression $\mathbf{r} = \underset{\sim}{\mathbf{S}}\mathbf{d}$ becomes zero when the data $\mathbf{d}$ is replace by the signal estimated from prediction filtering. This is because all the noise calculated in prediction filtering is orthogonal to $\underset{\sim}{\mathbf{S}}$, but everything in the estimated signal fits $\underset{\sim}{\mathbf{S}}$ perfectly.

Once an improved signal filter $\underset{\sim}{\mathbf{S}}$ is calculated from the estimated signal, this new filter may be used either to produce an improved prediction-filtering result, or it may be used to derive another inversion prediction result. If the response of the filter to the noise is

assumed to be small, the improved prediction-filtering result might be the final result, but generally, if the noise is large enough to corrupt the filter, the response of the filter to the noise should be removed with inversion prediction.

Figures 7.3 and 7.4 in the next section show that iterating the calculation of the signal annihilation filter has the desired effect of preserving the amplitudes of the calculated signal and reducing the wavelet distortion in cases of small signal-to-noise ratios. Both effects are the result of removing some of the noise from the data used in the filter calculations. The amplitude improvement is a straightforward result of having a filter that predicts the signal well, rather than having a filter that predicts the signal poorly. The reduction of the generated spurious events results from the filter not being forced by the noise to use events parallel to the predicted events to improve the predictions (Abma, 1994).

In the examples shown in the next section, three iterations of estimating the signal annihilation filter $\underset{\sim}{\mathbf{S}}$ were used. I have found that one or two iterations do not allow the amplitudes of the reflections to be restored properly and more iterations seem to weaken the reflections. More work needs to be done to find how the number of iterations affects weak events that do not line up with the strongest events in a section. It is possible that iterating tends to eliminate weak events not lined up with the strongest reflections, since a preliminary filter might attenuate a weak event which then would not be recovered in the following passes.

7.4 Examples

7.4.1 Synthetic data examples

The first synthetic example is one previously used in chapter 6 to show how t-x prediction filtering can generate spurious events that appear as wavelet distortions. Figure 7.3 shows how inversion prediction for the noise using equation (7.6) compares to prediction filtering. Although the inversion prediction result shows more organized noise in the background than the prediction-filtering result, the amplitude of the signal is better preserved in the inversion prediction result. Close-ups of the wavelets are seen in Figure 7.4. Notice that the input event has been distorted by the t-x prediction-filter result. While the inversion prediction result still shows some distortion of the wavelet, the distortion is small and the amplitude of the wavelet is better preserved than it is in the prediction-filtering result.

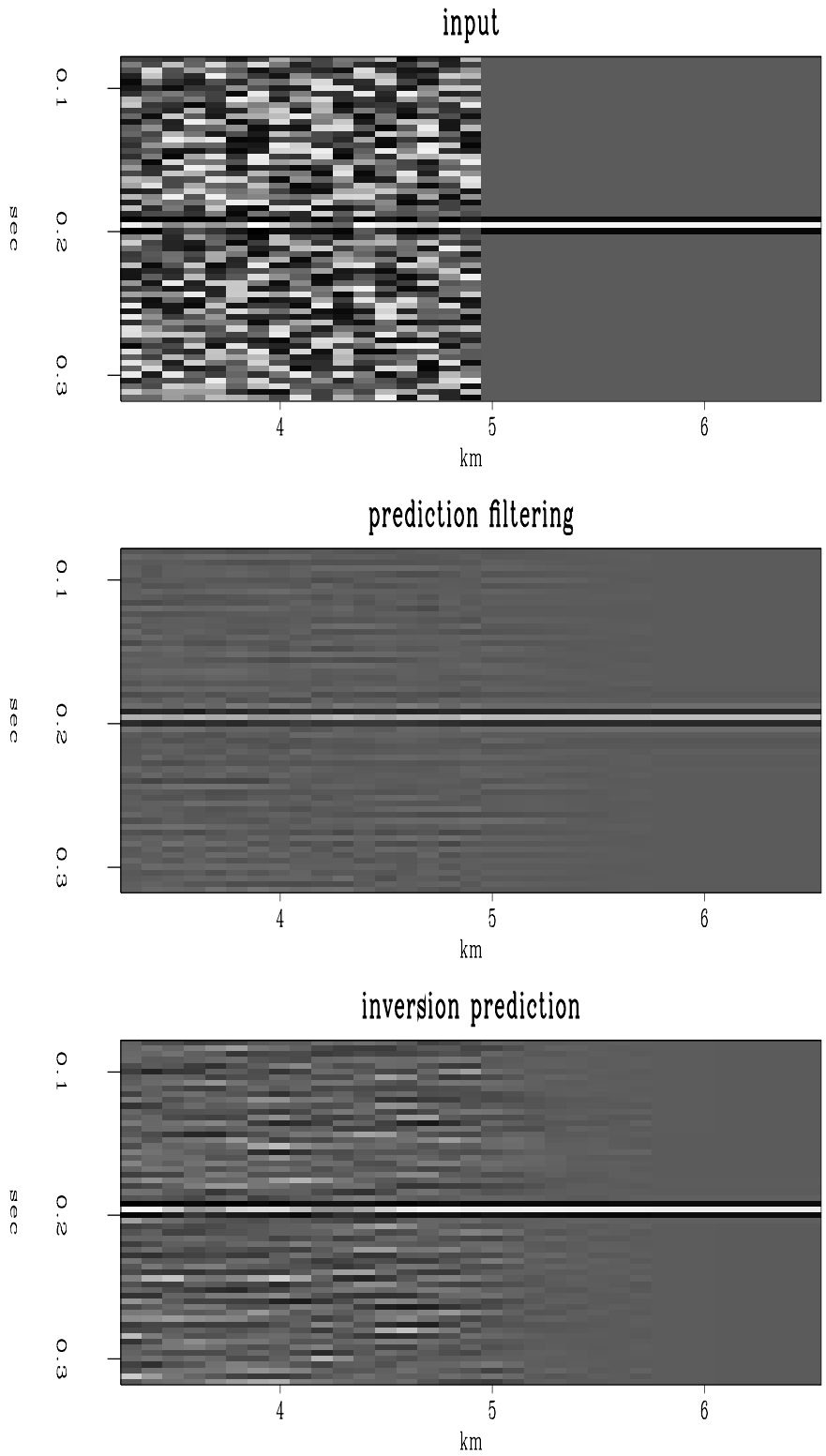

FIG. 7.3. A reflection buried in a field of random noise. The top plot is the original, the middle plot is the original with t-x prediction, and the bottom plot is the signal using the inversion results. rinver-synth3 [R]

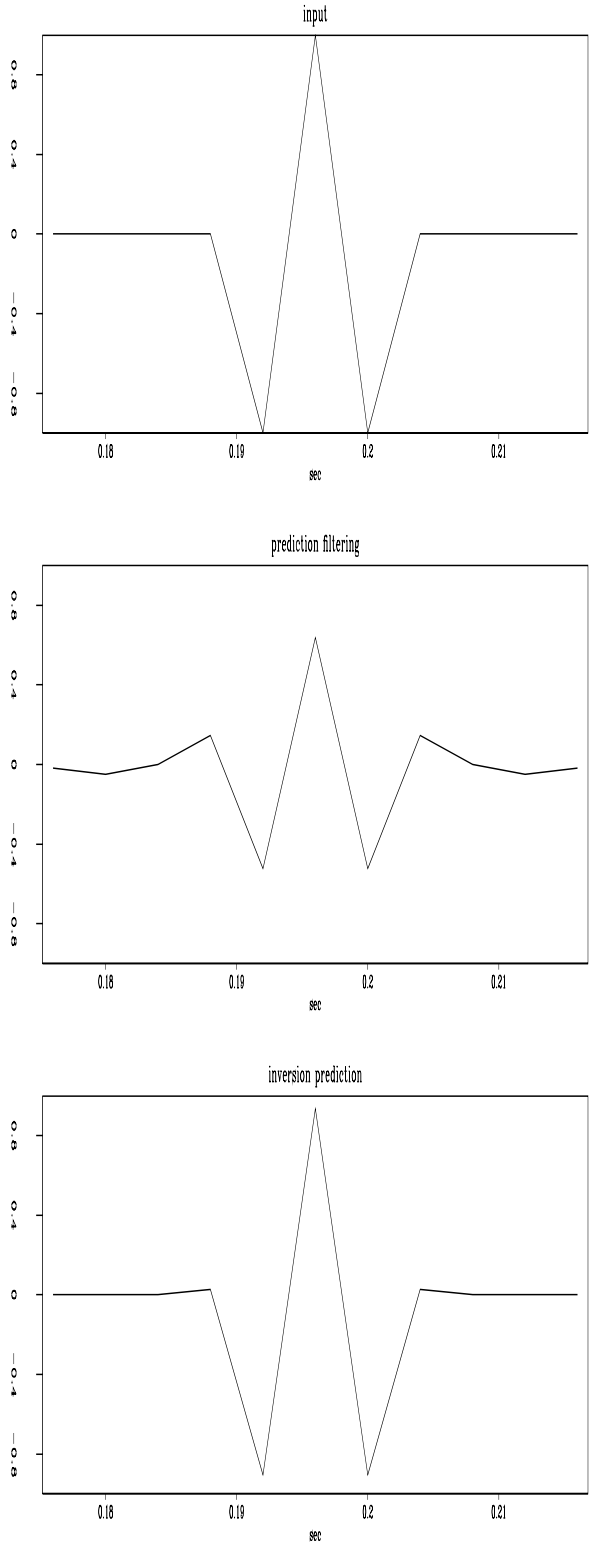

FIG. 7.4. A single trace taken from the right side of the data. The original reflection on the top shows a three-point wavelet. The middle plot is the t-x prediction result. The bottom plot is the inversion prediction result. rinver-graph3 [R]

7.4.2 Real data examples

Real data processed with the prediction prediction show results similar to the synthetic examples. It is difficult to confirm which result is better without independent information. Wavelet distortion is difficult to recognize in complex real data. Even so, the reflection amplitudes appear to be improved on the inversion prediction results when compared to the prediction-filtering results.

The first section in Figure 7.5 shows the input, a 2-dimensional line from a 3-dimensional survey. The second section in Figure 7.5 shows the result of applying a prediction filter to the data in the first panel. The results are significantly better than the input. The third section in Figure 7.5 show the results of the inversion prediction. The amplitudes on the inversion results are better preserved than the prediction-filter results. It is difficult to judge whether the events between 1.2 and 1.6 seconds are organized noise or weak reflections attenuated by the t-x prediction filter, but they are likely to be organized noise similar to that seen in the synthetic examples. Figure 7.6 shows a close-up of the data in Figure 7.5. The results of the inversion prediction are more appealing than the t-x prediction-filtering results.

7.5 Conclusions

In the presence of strong noise, prediction filtering attenuates reflections and produces spurious events. Inversion prediction preserves the reflection amplitudes and reduces the amplitudes of the spurious events. Although I was hoping for an improvement over prediction filtering, the signal-to-noise ratio of the output of inversion prediction generally appears to be about equal to that of prediction filtering. Inversion prediction removes the response of the filter to the noise, but this effect is difficult to see in real seismic data. The main advantage of the inversion prediction technique may be to clean up the signal annihilation filter in the presence of strong noise. For real seismic data, preserving the signal amplitude and reducing the amplitudes of spurious events may be more important than eliminating the filter response. However, if the noise consists of very large spikes, eliminating the filter response becomes important. Removing the filter response with the inversion may have more effect on the calculation of an improved filter than it does on the interpretation of the section.

In the next chapter, I extend this technique to account for missing data. Allowing

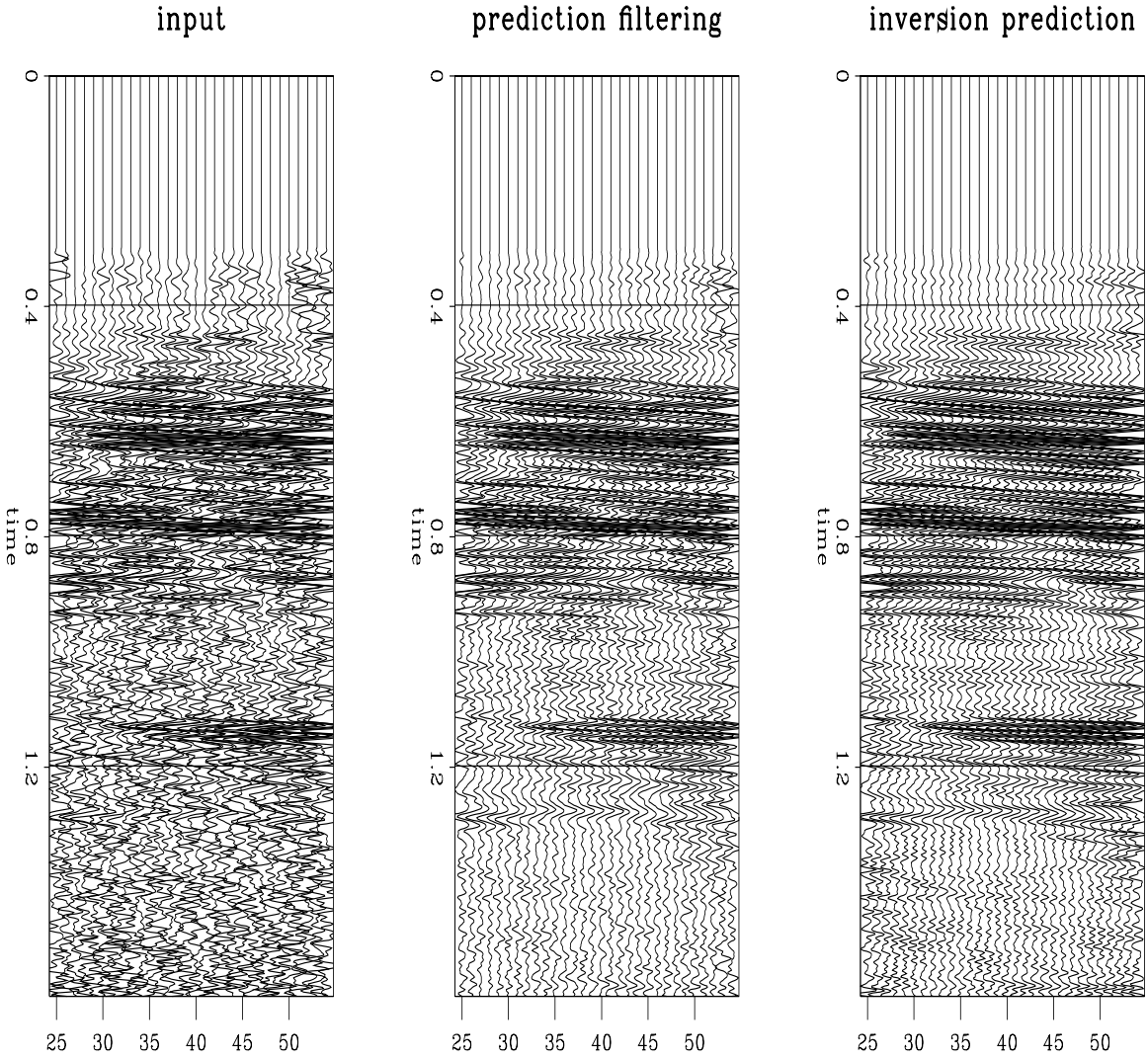

FIG. 7.5. The original data, t-x prediction, and inversion prediction rinver-real3 [R]

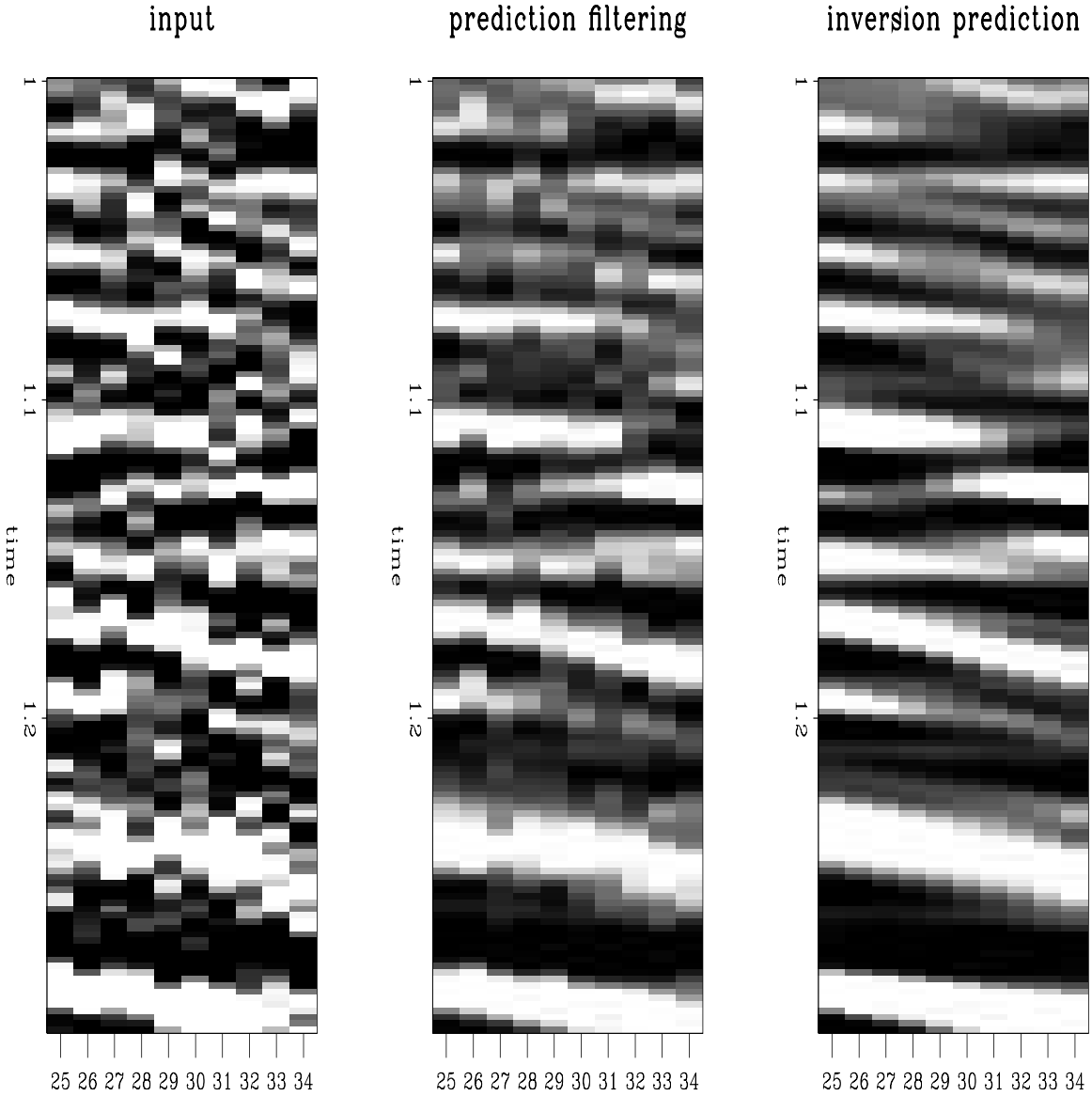

FIG. 7.6. A closeup of the original data, the t-x prediction, and the inversion prediction. rinver-clsup3 [R]

for missing data allows the removal of high-amplitude noise that would otherwise corrupt a least-squares inversion. Tolerating missing data also provides for prestack data, which, because of irregular acquisition, generally have not had these prediction techniques successfully applied.

Chapter 8

Signal and noise separation with missing data estimation

In the previous chapter, I showed a method for separating signal and noise with an inversion technique. In that chapter, it was assumed that the data was completely available and the only problem was the separation of signal and noise on samples organized on a regular grid. For much prestack data, although the data is generally still organized on a regular grid, at least some of the data is missing. In addition to the data not recorded, some samples are so contaminated with noise that they must be ignored, especially when using a least-squares inversion technique. A method for removing these samples was shown in chapter 2. This removal of bad data creates more samples that are effectively missing.

This chapter addresses the issue of predicting missing data while separating signal and noise by inversion.

8.1 Missing data prediction with signal and noise separation

Once the bad samples associated with high-amplitude noise are removed, it is necessary to restore these samples, as well as the samples that were not recorded. The process used in this chapter is a modification of the technique suggested by Claerbout (1995), but it requires only a single filter to describe the signal, as in the previous chapter. In chapter 10, a method using filters that describe both signal and noise will be used, and the method

of restoring missing data that follows the approach in Claerbout (1995) will be used.

8.1.1 Definitions

In this section, the terms needed to describe the inversion are defined. First, it is assumed that a signal annihilation filter $\underset{\sim}{\mathbf{S}}$ is available. When applied to the signal $\mathbf{s}$, the signal is eliminated to a good approximation: $\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$. The filter $\underset{\sim}{\mathbf{S}}$ is a purely lateral prediction filter as described in chapter 4 and is calculated in the same way as the $\underset{\sim}{\mathbf{S}}$ in chapter 7. The data $\mathbf{d}$ is assumed to be the sum of signal $\mathbf{s}$ and noise $\mathbf{n}$, or $\mathbf{d} = \mathbf{s} + \mathbf{n}$. The data $\mathbf{d}$ is also separated into the data that is known $\mathbf{k}$ and the data that is missing $\mathbf{m}$, so that $\mathbf{d} = \mathbf{k} + \mathbf{m}$. The missing data is the data not recorded or the data that has been eliminated by the high-amplitude noise muting routine presented in the previous section. Two masks are defined for use in the inversion. $\underset{\sim}{\mathbf{K}}$ is the mask, that when applied to the data $\mathbf{d}$, generates the known data values: $\mathbf{k} = \underset{\sim}{\mathbf{K}}\mathbf{d}$. $\underset{\sim}{\mathbf{M}}$ is the mask, that when applied to the data $\mathbf{d}$, generates the missing data values: $\mathbf{m} = \underset{\sim}{\mathbf{M}}\mathbf{d}$. The identity matrix $\underset{\sim}{\mathbf{I}}$ results when $\underset{\sim}{\mathbf{K}}$ and $\underset{\sim}{\mathbf{M}}$ are added: $\underset{\sim}{\mathbf{I}} = \underset{\sim}{\mathbf{K}} + \underset{\sim}{\mathbf{M}}$.

To summarize :

$\mathbf{d}$ = data

$\mathbf{s}$ = signal

$\mathbf{n}$ = noise

$\mathbf{k}$ = known data

$\mathbf{m}$ = missing data

$\underset{\sim}{\mathbf{K}}$ = known data mask

$\underset{\sim}{\mathbf{M}}$ = missing data mask

$\underset{\sim}{\mathbf{S}}$ = signal annihilation filter.

The relationships between these factors are as follows:

$\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$

$\mathbf{d} = \mathbf{s} + \mathbf{n}$

$\mathbf{d} = \mathbf{k} + \mathbf{m}$

$\underset{\sim}{\mathbf{I}} = \underset{\sim}{\mathbf{K}} + \underset{\sim}{\mathbf{M}}$ or $\mathbf{d} = \underset{\sim}{\mathbf{K}}\mathbf{d} + \underset{\sim}{\mathbf{M}}\mathbf{d}$.

8.1.2 Inversion for missing data with signal and noise

The basic form of the inversion is the same as that used in chapter 7, where the inversion for the noise $\mathbf{n}$ from the regression $\underset{\sim}{\mathbf{S}}\mathbf{d} \approx \underset{\sim}{\mathbf{S}}\mathbf{n}$ is stabilized by assuming that the noise is

approximately the noise estimated from the prediction-error filtering result $\underset{\sim}{\mathbf{S}}\mathbf{d}$. These are combined to produce the system

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \epsilon\underset{\sim}{\mathbf{S}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \epsilon \end{pmatrix} \mathbf{n}. \tag{8.1}$$

If $\mathbf{k} + \mathbf{m}$ is substituted for $\mathbf{d}$ in the above system of regressions, and the unknown missing data $\mathbf{m}$ is moved to the right side, the system becomes

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{K}}\mathbf{d} \\ \epsilon\underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{K}}\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} & -\underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{M}} \\ \epsilon & -\epsilon\underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{M}} \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ \mathbf{m} \end{pmatrix}. \tag{8.2}$$

This system might be further modified to account for the areas in the data where the prediction-error filtering result is not expected to produce a good estimate of the result. For isolated missing data samples, the change in the result is likely to be small, since a single missing sample is can be considered to be noise and will be well predicted. For groups of missing traces, the prediction-error filtering result will not be a good estimate, and the previous system should be modified to ignore the estimate of the missing data in these areas. The emphasis here is recovering small numbers of missing data samples, not interpolation of large gaps in the data. Even so, in the examples to follow, the signal seems to be reasonably extrapolated several traces into an area of missing traces.

8.1.3 Initializing the inversion

As in the previous chapter, the value of $\mathbf{n}$ is initialized to $\underset{\sim}{\mathbf{S}}\mathbf{d}$, which is just the prediction-error filter estimate of the noise. For poststack data, this is generally a good estimate, since it is just the common prediction-error estimate of the noise used in f-x or t-x prediction filtering. For prestack data, this estimate will be less accurate, since many more traces are likely to be missing in prestack data than in poststack data, but it is better than a starting estimate of zero for $\mathbf{n}$. Using $\underset{\sim}{\mathbf{S}}\mathbf{d}$ for the initialization is necessary to improve the results and reduce the cost of the inversion by reducing the number of iterations needed by the solver. Although the code is complicated slightly by this initialization, the time taken by the inversion is reduced by an order of magnitude. Even with this increased speed, the result is superior to the result with the noise initialized to zero. Details of the initialization and its implementation with the conjugate-gradient solver used here is discussed in Abma (1995). More details about the conjugate-gradient method itself may be found in Claerbout (1995), Strang (1986), and Luenburger (1984).

8.2 Results

The shot gather shown in Figure 8.1 was first processed with the high-amplitude noise removal process shown in chapter 2 to produce the results in Figure 8.2. The value of w, the multiplier of the median, was 5 in this case. After the bad samples were removed, the inversion of system (8.2) was used to predict the noise and the missing data. The missing data was then added to the known data to produce a full set of data. This full set of data then had the noise subtracted from it to produce the signal section shown in Figure 8.3. The results in Figure 8.3 appear reasonable. The noise is well attenuated and the signal appears strong. Although the signal is not well predicted into the missing data area near zero offset, it has been extended several traces. Figure 8.4 shows the difference between the original data shown in Figure 8.1 and the calculated signal shown in Figure 8.3. While some signal appears in the difference section, it is weak compared to the noise. Some of this apparent signal may be due to differences in the coupling of each receiver's geophones to the Earth. Irregularities in the reflection amplitudes will be poorly predicted by the signal annihilation filter and will appear in the noise section.

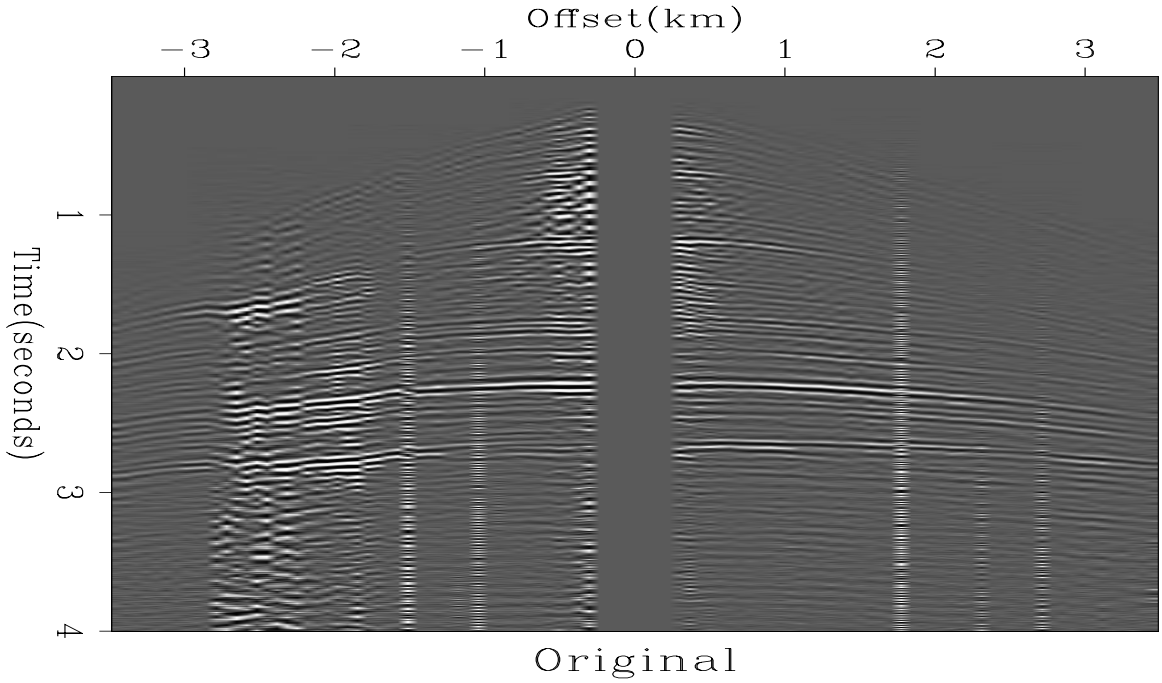

FIG. 8.1. A shot gather showing some bad traces and other noise. missing-original [R]

Figure 8.6 shows the data from Figure 8.5 with the noise removal applied. Next,

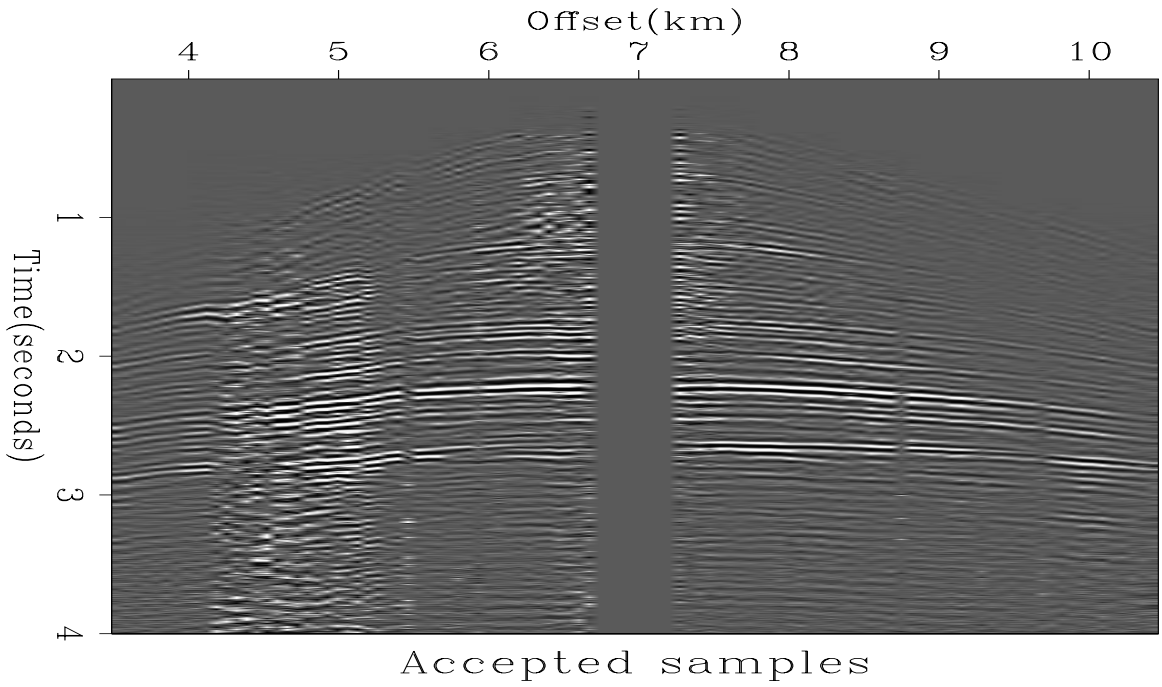

FIG. 8.2. The accepted samples from the data in the previous figure. missing-mpatchs [R]

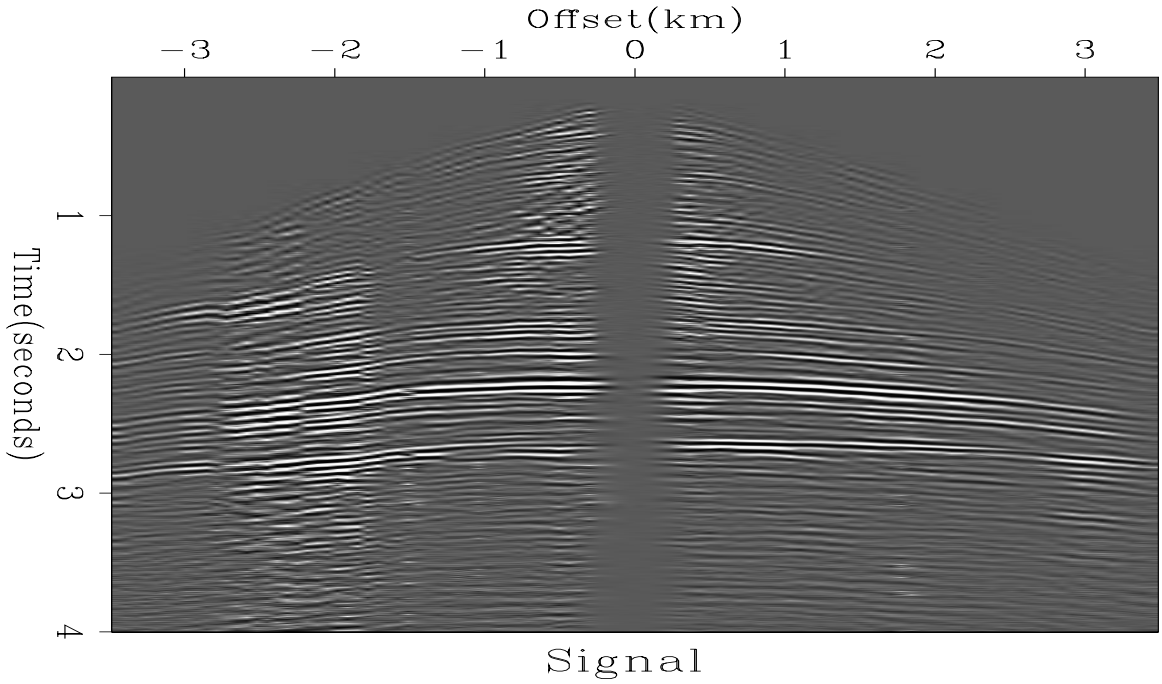

FIG. 8.3. The signal extracted from the shot gather. missing-abmpatch [R]

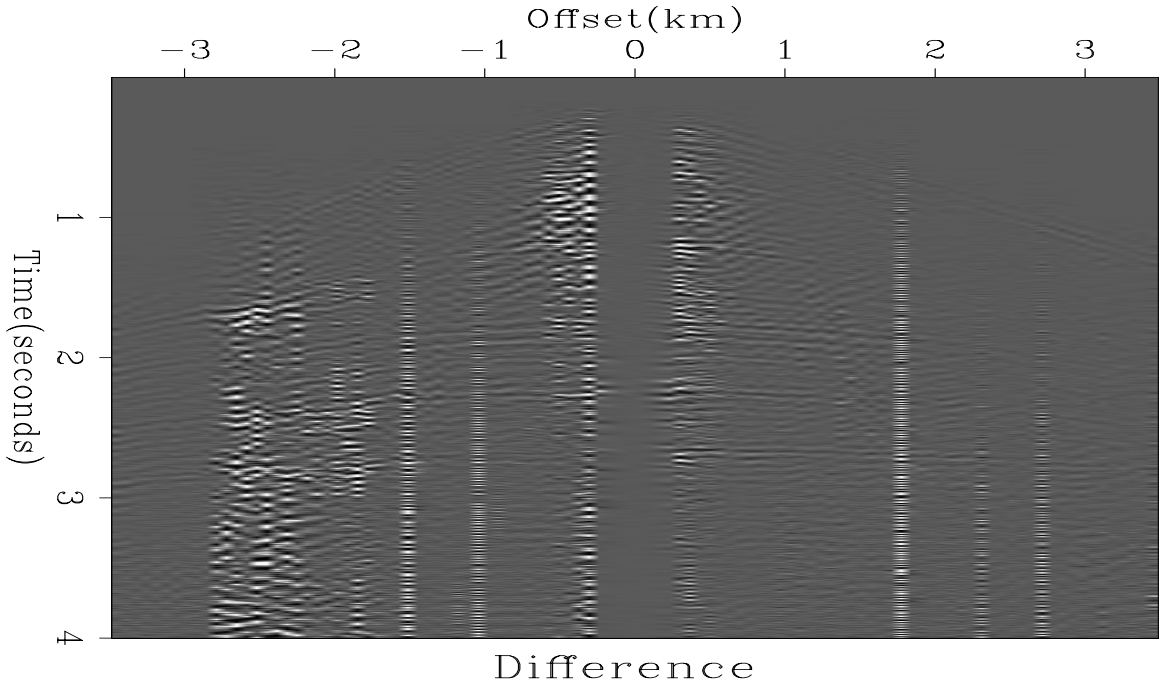

FIG. 8.4. The difference between the original data and the signal extracted. missing-diff [R]

Figure 8.7 shows the data from Figure 8.6 where the inversion was used to predict the missing data while separating signal and noise, as in the previous Figures. Notice that most of the coherent noise discussed previously was not restored into the signal in Figure 8.7. Figure 8.8 shows the difference between the original data and the noise. Little signal has leaked into the noise section and the results appear satisfactory.

8.3 Conclusions

To separate signal and noise in the presence of high-amplitude noise, the worst of the high-amplitude noise should be removed to avoid crippling the least-squares inversion. In chapter 2, a method for eliminating high-amplitude noise is demonstrated. In this chapter, an inversion that predicts missing data while it separates signal and noise is presented. This inversion is a modification of the inversion used in the previous chapter. Although missing data far from the known data is not well predicted, most of the missing data have been restored with reasonable success. The signal and the noise have also been separated successfully.

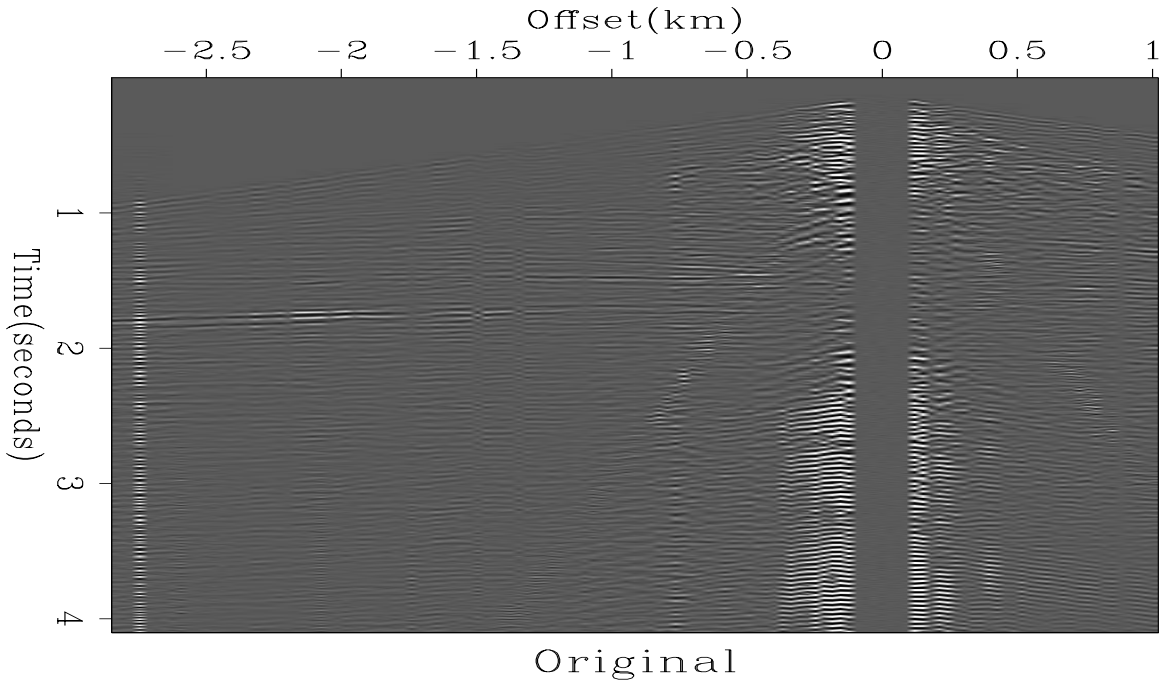

FIG. 8.5. A shot gather showing some bad traces and other noise. missing-original2 [R]

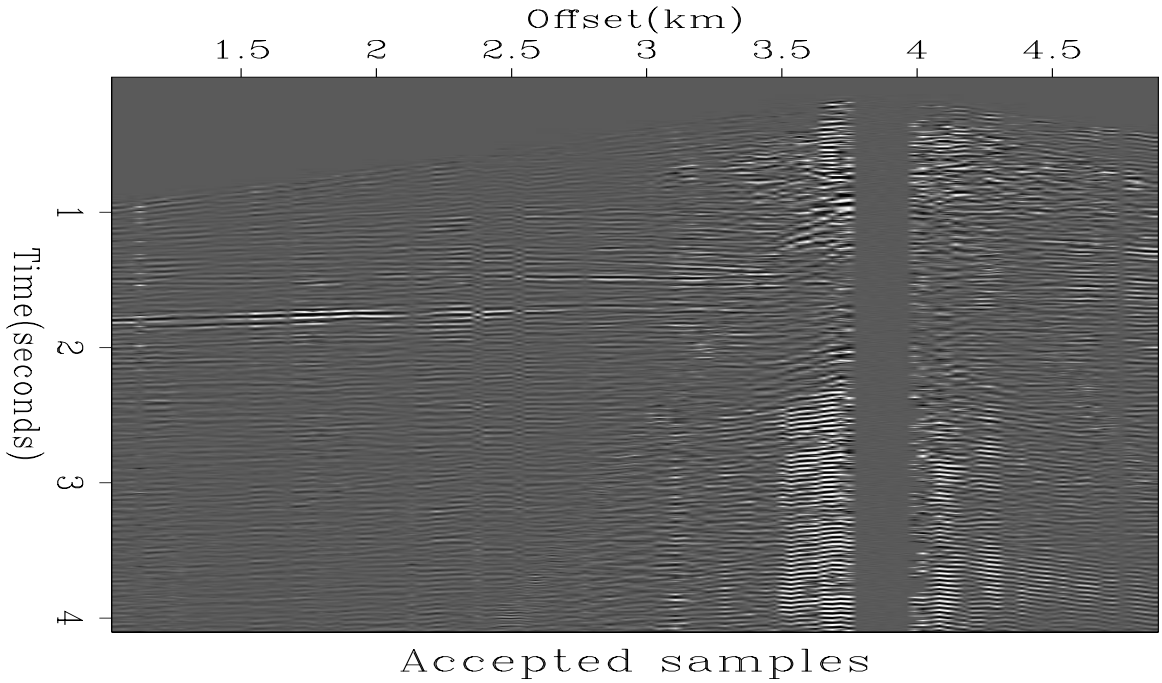

FIG. 8.6. The accepted samples from the data in the previous figure. missing-mpatch2s [R]

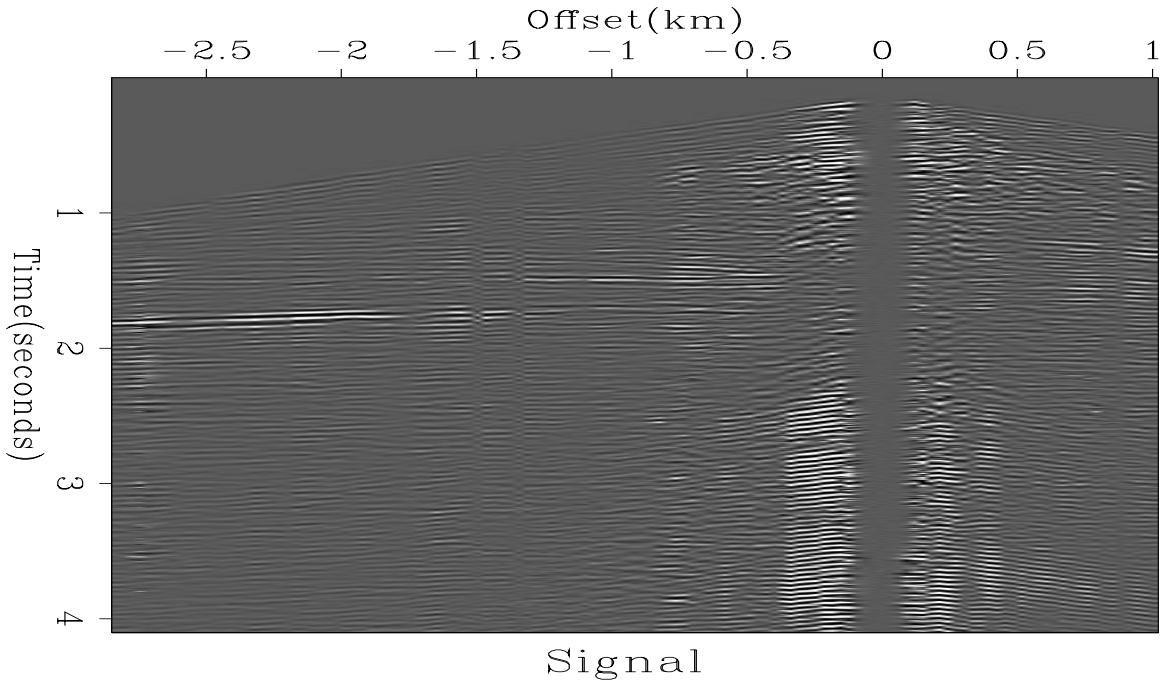

FIG. 8.7. The signal extracted from the shot gather. missing-abmpatch2 [R]

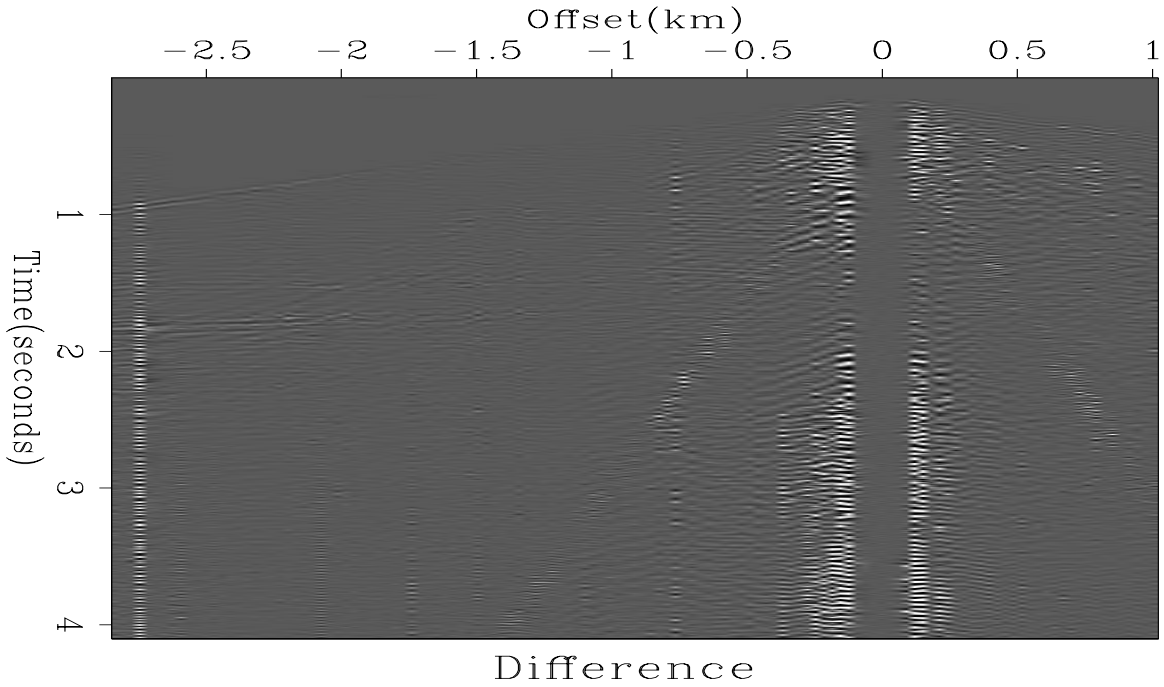

FIG. 8.8. The difference between the original data and the signal extracted. missing-diff2 [R]

When removing random noise, the process of predicting missing data and the process of separating signal and noise are likely to give similar answers when performed either separately or simultaneously. There is a practical advantage, since the cost of doing a single inversion is likely to be less than two inversions. Coherent noise will require the simultaneous calculation of the missing data with the calculation of the signal or noise. In the next chapters, I will extend these inversions to use not only a signal filter, but to use filters describing the noise also.

Chapter 9

Noise removal by characterizing both noise and signal: theory

In the previous chapter, the separation of signal and noise was done using only the signal annihilation filter. This process works well in the case of noise that cannot be predicted with a purely lateral filter. In this chapter, I address the more complicated issues involved in separating coherent noise from signal. This involves both signal and noise annihilation filters, as well as more complicated weighting of the various parts of the system to be solved.

9.1 Least-squares separation of signal and noise

9.1.1 Assumptions and definitions

In the following discussion, three assumptions are made to separate signal and noise from data. First, the data is defined to be a simple sum of the signal and noise; that is, $\mathbf{d} = \mathbf{s} + \mathbf{n}$, $\mathbf{d}$ being the observed data, $\mathbf{s}$ the signal, and $\mathbf{n}$ the noise. Next, there exists a filter $\tilde{\mathbf{S}}$ that predicts the signal, $\tilde{\mathbf{S}}\mathbf{s} \approx \mathbf{0}$. Finally, there exists a filter $\tilde{\mathbf{N}}$ that predicts the noise, $\tilde{\mathbf{N}}\mathbf{n} \approx \mathbf{0}$. The methods of getting $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{N}}$ will be covered later.

The assumed noise filter $\tilde{\mathbf{N}}$ requires a change in the definition of the noise from the previous chapters, where unpredictable noise was separated from a predictable signal. Although it will be shown later that unpredictable noise may be removed with the techniques to be discussed here, more emphasis is given now to coherent noise.

Three conditions are expected to be met by the final solution for the signal and noise:

$$\mathbf{d} = \mathbf{s} + \mathbf{n} \tag{9.1}$$

$$\underset{\sim}{\mathbf{S}}\mathbf{s} \approx \mathbf{0} \tag{9.2}$$

$$\underset{\sim}{\mathbf{N}}\mathbf{n} \approx \mathbf{0}. \tag{9.3}$$

Equation (9.1) is just a definition of how signal and noise combine make the data. Equations (9.2) and (9.3) characterize the expected properties of the signal and noise. These might be considered more as levelers than as equations, since the result of either $\underset{\sim}{\mathbf{S}}\mathbf{s}$ or $\underset{\sim}{\mathbf{N}}\mathbf{n}$ is very unlikely to be zero. The final solution for the signal $\mathbf{s}$ and the noise $\mathbf{n}$ is expected to minimize, in the least-squares sense, both $\underset{\sim}{\mathbf{S}}\mathbf{s}$ and $\underset{\sim}{\mathbf{N}}\mathbf{n}$.

Using equations (9.1) to (9.3), two systems of regressions may be generated, one to calculate the noise and one to calculate the signal. To calculate the signal, $\mathbf{n}$ replaces $\mathbf{d}-\mathbf{s}$ in equation (9.3), which is then combined with equation (9.2) to give a single system of regressions

$$0 \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s} - \begin{pmatrix} \mathbf{0} \\ \underset{\sim}{\mathbf{N}}\mathbf{d} \end{pmatrix}. \tag{9.4}$$

A similar manipulation produces a calculation of the noise

$$\mathbf{0} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{n} - \begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \mathbf{0} \end{pmatrix} \tag{9.5}$$

Once the signal is calculated, the noise is simply $\mathbf{n} = \mathbf{d} - \mathbf{s}$. If the noise is calculated, the signal is $\mathbf{s} = \mathbf{d} - \mathbf{n}$.

9.1.2 Initial estimates of the calculated noise and signal

For the discussion that follows, the calculation of the signal from equation (9.4) is labeled $\mathbf{s}_1$. The calculation of the noise from equation (9.4) is then $\mathbf{n}_1 = \mathbf{d} - \mathbf{s}_1$. The calculation of the noise from equation (9.5) is labeled $\mathbf{n}_2$. The calculation of the signal from equation (9.5) is then $\mathbf{s}_2 = \mathbf{d} - \mathbf{n}_2$. Is it true that $\mathbf{s}_1 = \mathbf{s}_2$ and is $\mathbf{n}_1 = \mathbf{n}_2$? Ideally, yes, but actually, the situation is more complicated. The two signals and the two noises calculated need not be equal, as will be shown in the examples below.

To imagine how $\mathbf{s}_1 \neq \mathbf{s}_2$, consider an event that is eliminated by both filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$. This event is in the null space of both $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ (Strang, 1988; Menke, 1989; Nichols,

1994a). In both equation (9.4) and equation (9.5) that event will be eliminated from the system, and no information about this event will be available for the solver. Therefore, no part of that event will occur in the calculated solutions $\mathbf{s}_2$ and $\mathbf{n}_1$. The event will then be completely contained in $\mathbf{n}_2$ and $\mathbf{s}_1$.

The initial estimates of $\mathbf{s}_2$ and $\mathbf{n}_1$ might be set to values varying from zero to the data $\mathbf{d}$ when using iterative methods for solving equations (9.4) and (9.5). If the initial estimates of $\mathbf{s}_1$ and $\mathbf{n}_1$ are zero, the problem will appear as equations (9.4) and (9.5). If the initial value of the signal $\mathbf{s}$ in equation (9.4) is the data $\mathbf{d}$, the constant

$$\begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{d} \tag{9.6}$$

should be subtracted from the residual that is minimized in solving equation (9.4)

$$\mathbf{r} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s} - \begin{pmatrix} \mathbf{0} \\ \underset{\sim}{\mathbf{N}}\mathbf{d} \end{pmatrix}, \tag{9.7}$$

$\mathbf{r}$ being the residual. If the constant is added to the right-hand side instead of subtracted from the residual, the expression

$$\mathbf{r} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s} - \begin{pmatrix} \mathbf{0} \\ \underset{\sim}{\mathbf{N}}\mathbf{d} \end{pmatrix} + \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{d} \tag{9.8}$$

results. Simplifying this gives

$$\mathbf{r} \approx \begin{pmatrix} \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s} + \begin{pmatrix} \underset{\sim}{\mathbf{S}}\mathbf{d} \\ \mathbf{0} \end{pmatrix}. \tag{9.9}$$

When this equation is compared to equation (9.5), it might be supposed that the $\mathbf{s}$ in equation (9.9) is $-\mathbf{n}$ from equation (9.5), which was previously labeled as $-\mathbf{n}_2$. When the initial value $\mathbf{d}$ of $\mathbf{s}$ is added, the result becomes $\mathbf{d}-\mathbf{n}_2$. Instead of the previously calculated value of $\mathbf{s}_1$ from equation (9.4), using the initial estimate of d for $\mathbf{s}$ in equation (9.4) gives the value $\mathbf{s}_2 = \mathbf{d} - \mathbf{n}_2$, which is the same answer as equation (9.5). A similar relationship is true for equation (9.4) and equation (9.5) if the estimated noise is set to $\mathbf{d}$.

The difference between solving with zero as the initial solution and solving with the data as the initial solution is simply where to put the null space. If the initial solution contains no null space data, the final solution will not contain any of the null space data. If the initial solution contains data that falls in the null space, the final solution will

leave this null space unchanged (Nichols, 1994a). The difference between equations (9.4) and (9.5) is the placement of events that fall in the null space.

To summarize the previous discussion, the solutions for the signal and noise derived from equations (9.1), (9.2), and (9.3) are the same whether the noise or signal is calculated, provided the initial estimates of the signal and noise are the same and the estimates for the signal and noise sum to the data $\mathbf{d}$. For example, equation (9.4) solved with an initial estimate of the signal of zero assumes the noise has an initial value of the data. Solving equation (9.5) with the same initial values of the signal being zero and the noise being the data gives the same results for the calculated noise and data. For a more symmetrical result, the noise and signal might both be initialized with half the data. This choice of initial values gives us a useful tool in specifying how data in the null space of both $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{N}}$ are distributed.

9.2 Synthetic examples of signal and noise estimations

9.2.1 Solutions with initial estimates of zero

Here I offer some simple examples of the previous ideas. Figure 9.1 shows two sections. The first section, which represents the signal, has a signal filter $\tilde{\mathbf{S}}$ calculated from it. The second section, which represents the noise, has a noise filter $\tilde{\mathbf{N}}$ calculated from it. Notice that the signal and noise sections contain a common event of intermediate dip. The data consists of all three events and is shown in Figure 9.2. Figure 9.3 shows the result of calculating the signal using the system of regressions seen in equation (9.4). Notice the event common to both the signal and noise does not appear in the calculated signal. Since the noise is just the signal subtracted from the noise, the common event appears in the noise section.

When noise is calculated with the system of regressions seen in equation (9.5), the event common to both the definition of noise and signal does not appear in the calculated noise, but is seen in the signal, which is now the noise subtracted from the data.

9.2.2 Solutions with non-zero initial estimates

When solving for the signal with equation (9.4) using an initial estimate of the signal being the data, the common event now appears in the calculated signal, as is seen in Figure 9.5.

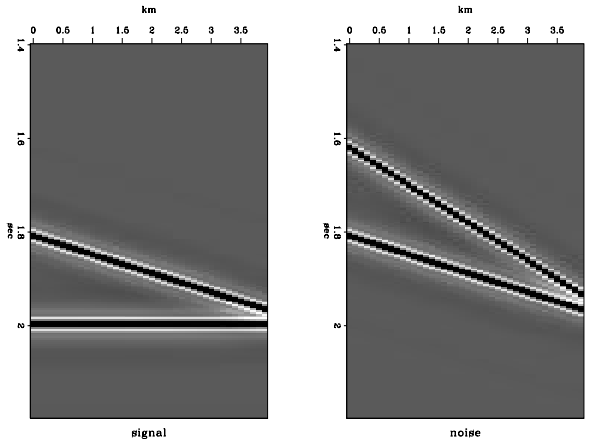

FIG. 9.1. The events on the left are defined as signal, the events on the right are defined as noise. noiserem-signalnoise [R]

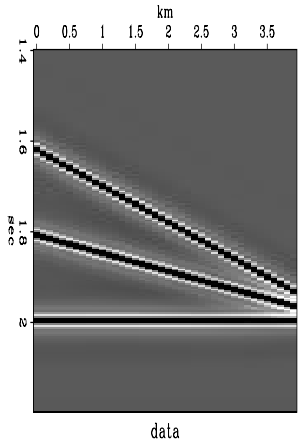

FIG. 9.2. The data, made up of both signal and noise. noiserem-data [R]

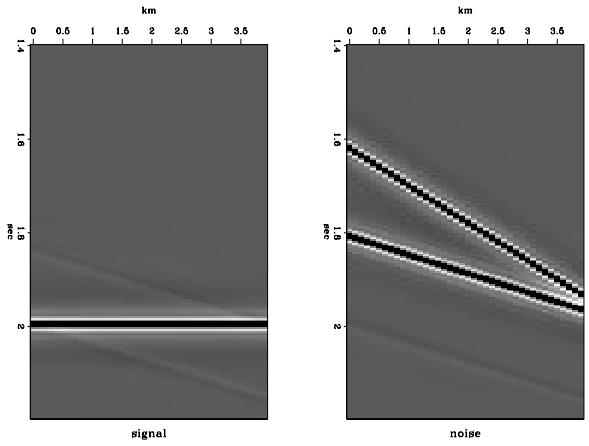

FIG. 9.3. The calculated signal and noise using an initial solution of zero for the signal. noiserem-separ7sa [R]

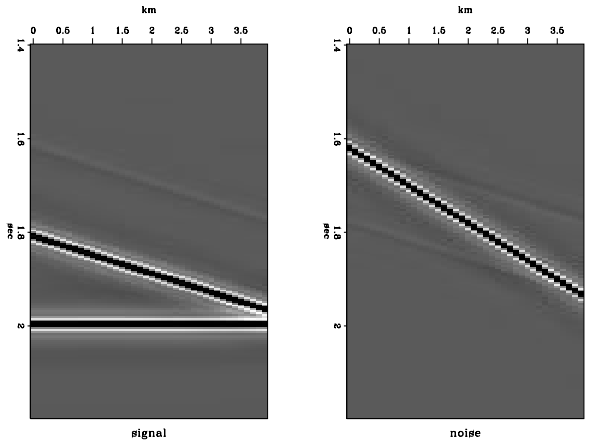

FIG. 9.4. The calculated signal and noise using an initial solution of zero for the noise. noiserem-separ7na [R]

When solving for the noise with equation (9.5) using an initial estimate of the noise being the data, the common event now appears in the calculated noise, as seen in Figure 9.6.

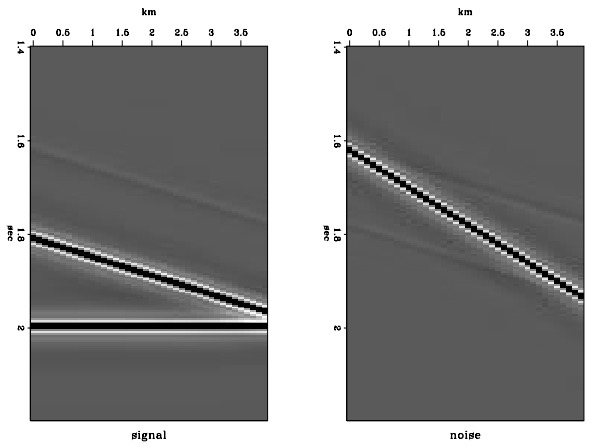

FIG. 9.5. The calculated signal and noise using an initial solution of the data for the signal. noiserem-separ7san [R]

The initial estimates for the signal and noise are not limited to zero and the data. If there is no reason to believe that the data in the null space of the operators $\underset{\sim}{\mathbf{S}}$ or $\underset{\sim}{\mathbf{N}}$ should belong to either the noise or the data, a more symmetrical approach would be to use one-half the data as the initial estimates of both the signal and noise.

In real data, the separation between the signal operator $\underset{\sim}{\mathbf{S}}$ and the noise operator $\underset{\sim}{\mathbf{N}}$ is likely to be less clear than it is in these examples. True null spaces, where an event is completely zeroed, are less likely in the presence of noise. The separation of events that are suppressed by both filters, but not in the null space, is considered next.

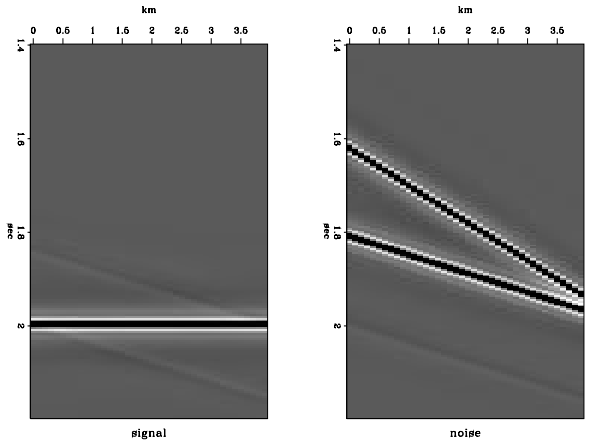

FIG. 9.6. The calculated signal and noise using an initial solution of the data for the noise. noiserem-separ7nas [R]

9.3 Distribution of events not in the operator null space

In the discussion above, events were assumed to be either in the null space of an operator or not. In general, an operator will attenuate an event to some extent. The null space of an operator is completely attenuated, while other events are partially attenuated, depending on the situation in which the filter is designed. Operators derived from real seismic data are unlikely to perfectly predict either signal or noise. Events that are simply attenuated by the signal filter $\underset{\sim}{\mathbf{S}}$ and the noise filter $\underset{\sim}{\mathbf{N}}$ will be distributed between the calculated signal $\mathbf{s}$ and the calculated noise $\mathbf{n}$ depending on the relative attenuation of $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$.

9.3.1 Event attenuation by weighting

Imagine an event that is attenuated, but not removed, by filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$.

$$\begin{aligned} \boldsymbol{\epsilon}_1 &= \underset{\sim}{\mathbf{S}}\mathbf{x} \\ \boldsymbol{\epsilon}_2 &= \underset{\sim}{\mathbf{N}}\mathbf{x}, \end{aligned} \tag{9.10}$$

where $\mathbf{x}$ contains the event, $\underset{\sim}{\mathbf{S}}$ is the signal filter, $\underset{\sim}{\mathbf{N}}$ is the noise filter, $\boldsymbol{\epsilon}_1$ is the response of the event to filter $\underset{\sim}{\mathbf{S}}$, and $\boldsymbol{\epsilon}_2$ is the response of the event to filter $\underset{\sim}{\mathbf{N}}$.

$$\begin{aligned} \varepsilon_1 &= \Sigma \boldsymbol{\epsilon}_1^2 \\ \varepsilon_2 &= \Sigma \boldsymbol{\epsilon}_2^2 \end{aligned} \tag{9.11}$$

is a measure of the power of the filtered events. When getting a least-squares solution for equations such as (9.4) and (9.5), the events included in $\mathbf{x}$ will be distributed between signal and noise as $1/\varepsilon_1$ for the signal and $1/\varepsilon_2$ for the noise.

This distribution may be changed by modifying the system of equations. Consider, for example, this system:

$$\mathbf{0} \approx \begin{pmatrix} \lambda \underset{\sim}{\mathbf{S}} \\ \underset{\sim}{\mathbf{N}} \end{pmatrix} \mathbf{s} - \begin{pmatrix} \mathbf{0} \\ \underset{\sim}{\mathbf{N}}\mathbf{d} \end{pmatrix}. \tag{9.12}$$

If λ is less than 1, $1/\varepsilon_1$ will increase, allowing relatively more of the event into the signal. If λ is greater than 1, $1/\varepsilon_2$ will increase, allowing relatively more of the event into the noise. If λ is very large, only events that are almost perfectly removed by $\underset{\sim}{\mathbf{S}}$ will be allowed into the signal.

Even for events that are perfectly predicted and removed by filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$, the distribution of events may be controlled by the weighting in equation (9.9), which is the prediction equation with the initial estimate of the signal as the data. In this case, the λ controls the final distribution of events in the null space of $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$. Once again, if λ is less than 1, $1/\varepsilon_1$ will increase, forcing relatively more of the event into the signal. If λ is greater than 1, $1/\varepsilon_2$ will increase, forcing relatively more of the event into the noise.

Once again, the weighting in system (9.12) may be thought of in terms of using $\underset{\sim}{\mathbf{S}}\mathbf{s}$ and $\underset{\sim}{\mathbf{N}}\mathbf{n}$ as levelers. If $\underset{\sim}{\mathbf{S}}\mathbf{s}$ is weighted higher than $\underset{\sim}{\mathbf{N}}\mathbf{n}$, the least-squares solutions of $\mathbf{s}$ and $\mathbf{n}$ will be modified since the values of ε_1 and ε_2 are modified. In the unlikely event that either $\underset{\sim}{\mathbf{S}}\mathbf{s}$ or $\underset{\sim}{\mathbf{N}}\mathbf{n}$ actually becomes zero, the weighting becomes unimportant, since one of the conditions is fit perfectly and no better solution could be found. In practical situations, both $\underset{\sim}{\mathbf{S}}\mathbf{s}$ and $\underset{\sim}{\mathbf{N}}\mathbf{n}$ will have some residual and can only be minimized.

9.3.2 Examples of event distribution by weighting

Several cases are shown here that illustrate the effects of the weighting in the previous discussion. The first case shows the effects of varying λ in equation (9.12). Figure 9.7 shows the events from which the signal filter $\underset{\sim}{\mathbf{S}}$ and the noise filter $\underset{\sim}{\mathbf{N}}$ are calculated. The data in Figure 9.7 are therefore taken as definitions of signal and noise, the signal being the flat event and the noise being the dipping event. Figure 9.8 shows the data to be separated into signal and noise. In addition to the signal and noise seen in Figure 9.7, an event with a dip of intermediate slope has been added in Figure 9.8. This event is only slightly attenuated by the filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$.

By solving equation (9.12) with $\lambda = 1$, the results seen in Figure 9.9 are obtained. The event with the intermediate slope has been about evenly distributed between the signal and the noise.

Next, equation (9.12) is solved with $\lambda = 10$. Increasing λ increases the weight given to the top part of equation (9.12), $\mathbf{0} \approx \tilde{\mathbf{S}}\mathbf{s}$, so events that do not fit $\tilde{\mathbf{S}}$ extremely well get eliminated from $\mathbf{s}$. As expected, Figure 9.10 shows the event with intermediate slope has been almost completely moved to the noise.

When λ is decreased to 0.1, the weight given to the top part of equation (9.12) is decreased so any event that does not fit the lower part of equation (9.12) extremely well is pushed into $\mathbf{s}$. This can be seen in Figure 9.11, where the event of intermediate slope is almost entirely contained in the signal.

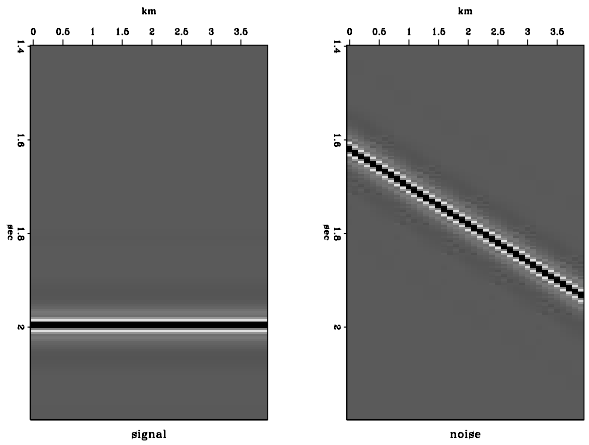

FIG. 9.7. The event on the left is defined as signal, the event on the right is defined as noise. noiserem-signalnoise.a [R]

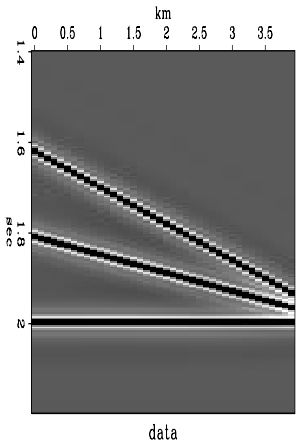

FIG. 9.8. The data, made up of both signal and noise, and an added event. noiserem-data.a [R]

In the previous examples, equation (9.12) has been solved with a zero estimated value of $\mathbf{s}$. This was possible since the signal was not significantly attenuated by the filter $\tilde{\mathbf{N}}$. In

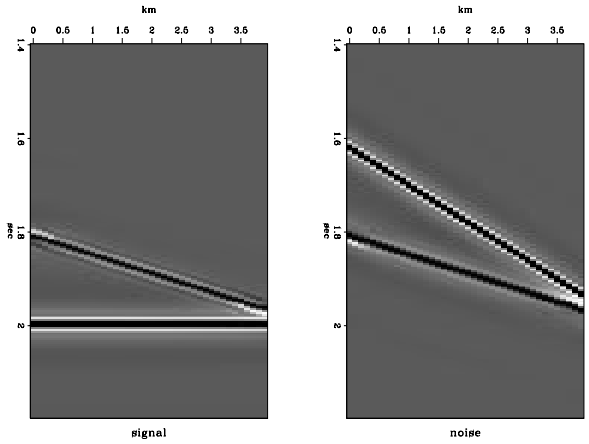

FIG. 9.9. The calculated signal and noise using $\lambda = 1$. noiserem-separ7sal.a.1 [R]

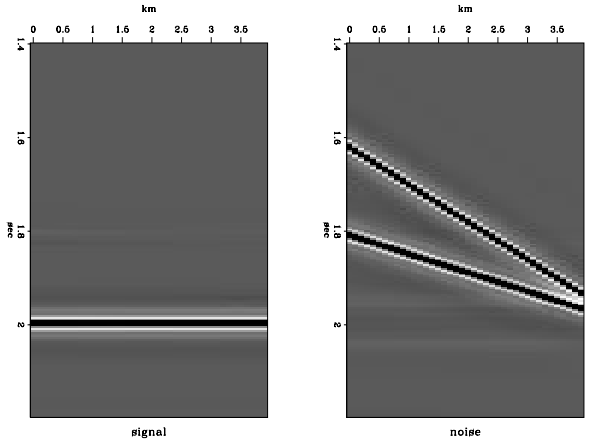

FIG. 9.10. The calculated signal and noise using $\lambda = 10$. noiserem-separ7sal.a.10 [R]

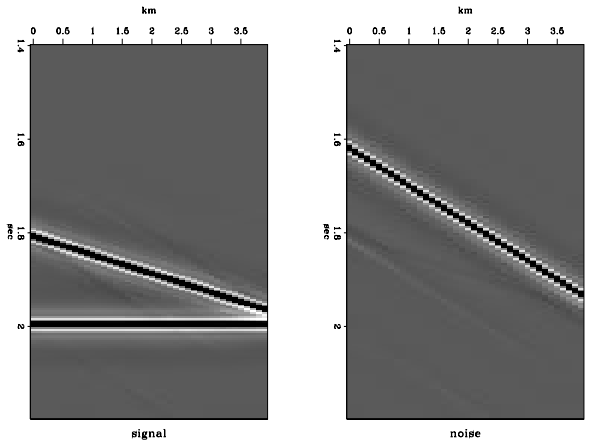

FIG. 9.11. The calculated signal and noise using $\lambda = 0.10$. noiserem-separ7sal.a.0.1 [R]

the next examples, equation (9.12) has been solved with a preliminary estimate of $\mathbf{s}$ being the data $\mathbf{d}$, since both filters $\underset{\sim}{\mathbf{S}}$ and $\underset{\sim}{\mathbf{N}}$ can completely eliminate one part of the data. For Figures 9.13 to 9.15, the signal filter $\underset{\sim}{\mathbf{S}}$ is a two-dimensional prediction-error filter with the form

$$\begin{matrix} 0 & a_{-2,1} & a_{-2,2} & a_{-2,3} & a_{-2,4} \\ 0 & a_{-1,1} & a_{-1,2} & a_{-1,3} & a_{-1,4} \\ 1 & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \end{matrix} \quad . \tag{9.13}$$

The noise filter $\underset{\sim}{\mathbf{N}}$ is a one-dimensional prediction-error filter with the form

$$\begin{matrix} 1 \\ a_1 \\ a_2 \end{matrix} \quad . \tag{9.14}$$

Figure 9.12 shows the events defined as the signal and noise. The signal is a series of horizontal events with random amplitudes. The noise is mono-frequency sine waves with random shifts. Both filters (9.13) and (9.14) will eliminate the sine waves, since a prediction is done along the time axis, but only filter (9.13) can predict the signal, since the amplitudes in time are random and unpredictable by filter (9.14). To allow any of the noise in the output, equation (9.12) must be solved with a preliminary estimate of $\mathbf{s}$ being the data, or all the sine waves will be removed from the system.

When equation (9.12) is solved with $\lambda = 1$, Figure 9.13 shows that the noise is evenly distributed between the calculated signal and the calculated noise. Increasing λ to 10 moves the sine waves into the noise section, producing the excellent separation of signal and noise seen in Figure 9.14. Decreasing λ to 0.1 moves the sine waves almost completely into the signal. This weighting gives a useful tool in distributing events between calculated signal and noise.

9.4 Conclusions

Inversions using two annihilation filters that remove common events require careful weighting to obtain the desired results. While this weighting introduces extra complications to this process, it also offers extra flexibility in determining the output. Nevertheless, while events common to both the signal and noise may be distributed between the calculated

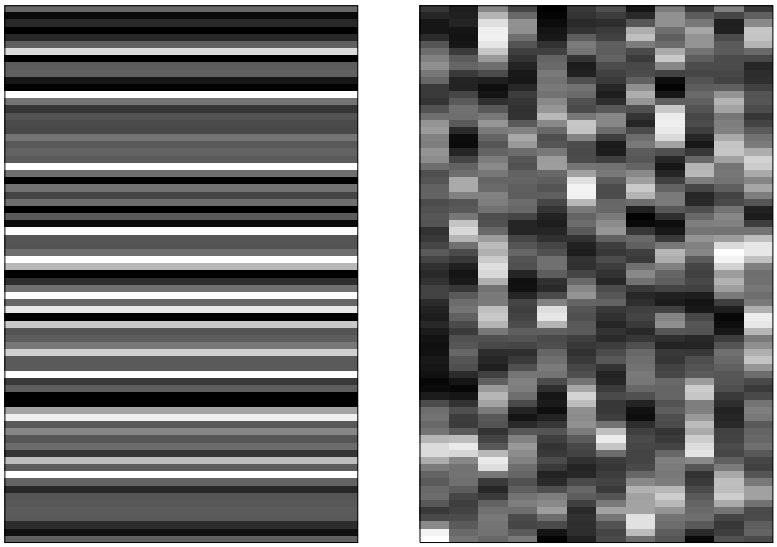

FIG. 9.12. The events on the left are defined as signal, the events on the right are defined as noise. noiserem-input [R]

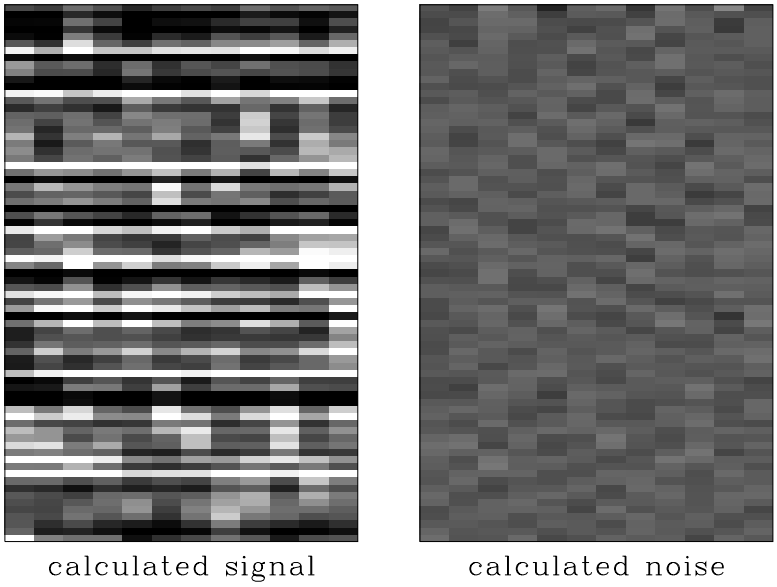

FIG. 9.13. The calculated signal and noise using $\lambda = 1$. noiserem-signois.1 [R]

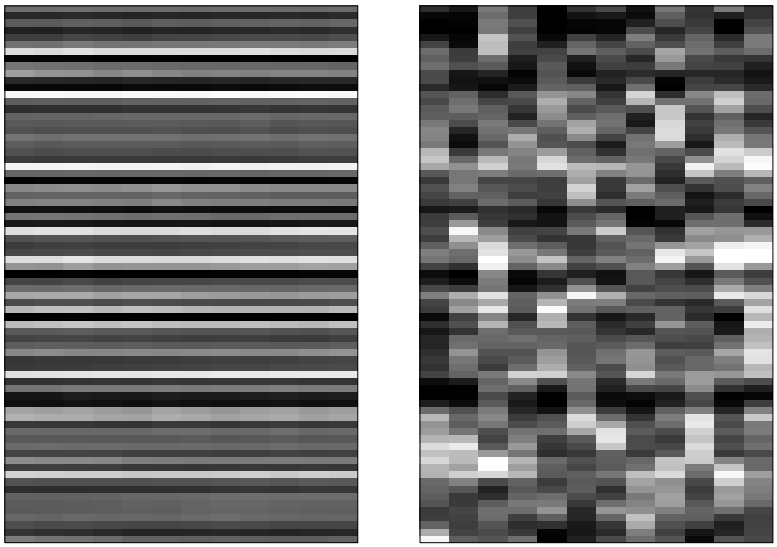

FIG. 9.14. The calculated signal and noise using $\lambda = 10$. noiserem-signois.10 [R]

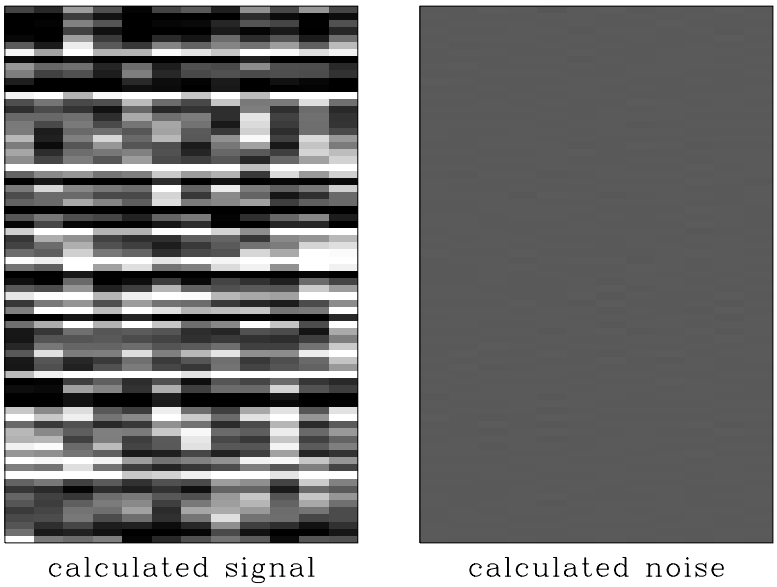

FIG. 9.15. The calculated signal and noise using $\lambda = 0.10$. noiserem-signois.0.1 [R]

signal and noise according to the user's specifications, these events cannot be uniquely determined to be either signal or noise.

Chapter 10

Noise removal by characterizing both noise and signal: applications

In the previous chapter, some of the theory behind solving a system of regressions by characterizing both the signal and the noise was presented. It was shown there that the weighting of the various parts of the system and the initialization of the system are important considerations.

In this chapter I apply the ideas of the previous chapter to remove two different types of noise. The first applications involve removing noise that is limited to either a single trace or to a small number of nearby traces within a given shot. The second example is the removal of coherent noise, in this case, ground roll.

The signal and noise separation techniques in this chapter are developed in a sequence of four steps. The first step is characterizing the signal and noise by amplitude only, the second step adds characterization of the signal and noise by one-dimensional filters, the third step makes the signal filter a two-dimensional filter, and the fourth step characterizes both the signal and the noise by two-dimensional filters as well as by amplitude. Examples of the results of the various techniques are demonstrated on synthetics and real data.

10.1 Amplitude estimation of signal and noise

10.1.1 Least-squares amplitude estimation

I assume that a collection of seismic data $\mathbf{d}(t,x)$ consists of signal $\mathbf{s}(t,x)$ and noise $\mathbf{n}(t,x)$ so that

$$\mathbf{d}(t,x) = \mathbf{s}(t,x) + \mathbf{n}(t,x). \tag{10.1}$$

The variable t corresponds to time and x corresponds to the offset of a recorded sample for the shot data considered in this chapter. For simplicity, since I assume the signal varies by approximately t^2, a scaled signal $\mathbf{s}'$ is defined as

$$\mathbf{s}' = t^2\mathbf{s}. \tag{10.2}$$

so that

$$\mathbf{d}(t,x) = \frac{1}{t^2}\mathbf{s}' + \mathbf{n}(t,x). \tag{10.3}$$

The use of the t^2 in equation (10.7) indicates that a sample of $\mathbf{s}$ is multiplied by the time of that sample squared to get the corresponding output sample in $\mathbf{s}'$. Although the function t^2 is actually a matrix with the corresponding gain values of t^2 on the diagonal, for simplicity I will continue to use this notation for amplitude terms that are connected with time.

In this chapter I assume the amplitude of the signal varies by approximately t^2 and is zero above the start time. This amplitude variation is represented in this chapter as the function $T(t,x,v)$. $T(t,x,v)$ is not necessarily fixed as the t^2 scalar, or even fixed in x, but may be modified to fit the data being considered. For simplicity, I assume in this discussion that the data require a t^2 scalar. The v in $T(t,x,v)$ represents the start time velocity. It is unlikely that an exact amplitude representation for the signal is needed, since I am looking only for an approximation of the amplitude response, but the scaling for $T(t,x,v)$ should fit the data at least approximately.

Replacing t^2 in equation (10.3) with a generalized time scalar $T'(t,x,v)$ gives

$$\mathbf{d}(t,x) = \frac{1}{T'(t,x,v)}\mathbf{s}' + \mathbf{n}(t,x), \tag{10.4}$$

where $T'(t,x,v)$ is just $T(t,x,v)$, except that T' produces large values before the start time, so that $1/T'(t,x,v)$ is effectively zero there. Thus, in equation (10.4), $\mathbf{n}(t,x)$ becomes $\mathbf{d}(t,x)$ before the start time as $1/T'(t,x,v)$ goes to zero.

I further assume that the RMS amplitude σ_n of the noise is independent of time, while the RMS amplitude σ_s of $\mathbf{s}'$, the signal scaled by time squared, is constant. Here, σ_n is a function of trace position, or x, making it $\sigma_n(x)$, while σ_s is a constant for the entire shot record. Both $\sigma_n(x)$ and σ_s are calculated over zones of the input that are considered typical of the noise and signal, respectively. Each $\sigma_n(x)$ is calculated over the part of the trace before the first arrival as

$$\sigma_n(x) = \sqrt{\frac{1}{N}\sum_{t=0}^{x/v} \mathbf{d}(t,x)^2}, \tag{10.5}$$

where N is the number of samples between $t = 0$ and x/v is the start time for that offset, x being the offset and v being the start time velocity. For traces with small offsets and little data before the first breaks, it may be possible to get a measure of $\sigma_n(x)$ from the deeper data, after the signal has died off.

Next, σ_s is approximated from the shot gather using the data after the first breaks, assuming that the reflection amplitudes scale as t^2, so

$$\sigma_s = \sqrt{\frac{1}{N}\sum_{x}\sum_{t=x/v}^{t_{\text{end}}} \left(t^2\mathbf{d}(t,x)\right)^2}, \tag{10.6}$$

where N is the total number of samples after the first break time on all traces, and t_{end} is the end of trace time. If the mean value of each trace is zero, σ_s^2 and $\sigma_n(x)^2$ are the estimated variances of the signal and noise.

To make an amplitude estimation of the signal and noise, I scale the noise and signal by the corresponding RMS amplitudes and minimize the result, so that

$$0 \approx \frac{1}{\sigma_s}\mathbf{s}' = S\mathbf{s}' \tag{10.7}$$

and

$$0 \approx \frac{1}{\sigma_n(x)}\mathbf{n} = N\mathbf{n}, \tag{10.8}$$

where $S = 1/\sigma_s$ and $N = 1/\sigma_n(x)$. Adding the condition that

$$\mathbf{d} = \mathbf{s} + \mathbf{n}, \tag{10.9}$$

or

$$\mathbf{d} = \frac{\mathbf{s}}{T'} + \mathbf{n}, \tag{10.10}$$

where $\mathbf{s}' = T'\mathbf{s}$, the regression in equation (10.8) becomes

$$\mathbf{0} \approx N\left(\mathbf{d} - \frac{\mathbf{s}'}{T'}\right), \tag{10.11}$$

making the system of regressions in equations (10.7) and (10.8)

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix} \approx \begin{pmatrix} S\mathbf{s}' \\ N\left(\mathbf{d} - \frac{\mathbf{s}'}{T'}\right) \end{pmatrix}. \tag{10.12}$$

Rearranging this expression produces

$$\begin{pmatrix} \mathbf{0} \\ N\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} S \\ \frac{N}{T'} \end{pmatrix} \mathbf{s}'. \tag{10.13}$$

Given that the least-squares solution for the expression $\mathbf{y} = A\mathbf{x}$ is $\mathbf{x} = (A^\dagger A)^{-1}A^\dagger \mathbf{y}$, the least-squares solution for $\mathbf{s}$ is

$$\mathbf{s}' = \left(\begin{pmatrix} S & \frac{N}{T'} \end{pmatrix} \begin{pmatrix} S \\ \frac{N}{T'} \end{pmatrix} \right)^{-1} \begin{pmatrix} S & \frac{N}{T'} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ N \end{pmatrix} \mathbf{d}, \tag{10.14}$$

which, when multiplied, gives

$$\mathbf{s}' = \frac{\frac{N^2}{T'^2}}{S^2 + \frac{N^2}{T'^4}}\mathbf{d}, \tag{10.15}$$

or expressed as the estimated RMS amplitudes is

$$\mathbf{s}' = \frac{T(x,t,v)\sigma_s^2}{T(x,t,v)^2\sigma_s^2 + \sigma_n^2(x)}\mathbf{d}. \tag{10.16}$$

The data $\mathbf{d}$ and signal $\mathbf{s}'$ are functions of $\mathbf{x}$ and t and may be expressed as $\mathbf{d}(x,t)$ and $\mathbf{s}'(x,t)$, but for the rest of this discussion they will be expressed as simply $\mathbf{d}$ and $\mathbf{s}'$ to avoid some complexity. The original unscaled signal becomes

$$\mathbf{s} = \frac{T(x,t,v)^2\sigma_s^2}{T(x,t,v)^2\sigma_s^2 + \sigma_n^2(x)}\mathbf{d}. \tag{10.17}$$

The expression for the noise is

$$\mathbf{n} = \frac{\sigma_n(x)^2}{T(x,t,v)^2\sigma_s^2 + \sigma_n^2(x)}\mathbf{d}. \tag{10.18}$$

It can been seen from equations (10.17) and (10.18) that $\mathbf{s} + \mathbf{n} = \mathbf{d}$, which is consistent with equation (10.1).

In the present case, S and σ_s are constant over all traces, while $N(x)$ and $\sigma_n(x)$ are a function of space only. In the more general case, these values may be calculated from some spatial separation of the signal and noise on the data.

The scale factors in equations (10.17) and (10.18) only change the amplitude of the data to what I assume would be the correct result if there existed only noise or signal. I have done no real separation of noise and signal other than estimating the amplitudes. I attempt to take the separation one step farther in sections 10.2, 10.3, and 10.4 where the spectral characteristics of the noise and signal are included.

10.1.2 Examples of amplitude estimation

To demonstrate how the amplitude scalars appear for a real data case, the data in Figure 10.1 were analyzed. Figure 10.2 shows the calculated scale factor for the noise. Notice that the strongest values follow what appears to be noisy traces on the input file in Figure 10.1. The noise in Figure 10.1 was scaled by the scale factor for the noise, which is displayed at the bottom of Figure 10.3. The distribution of amplitudes seems reasonable. The top of Figure 10.3 shows the original file scaled with the calculated signal scale factors. Once again, the distribution of amplitudes seems reasonable.

10.2 Separation by 1-D spectral estimation

10.2.1 1-D Separation theory

To improve the separation, more information is required. Here I attempt to use the time spectrum of the signal and the noise as the extra information. In a manner similar to the process used to determine the amplitudes, the spectrum of each trace is calculated from the data above the first break, where the data above the first breaks are assumed to be noise. Each trace has a separate filter calculated. For the noise, a single filter is derived over the data after the first breaks.

The filters used here annihilate the signal and noise. The signal filter is represented by

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_s \mathbf{s}', \tag{10.19}$$

where $\mathbf{s}$ is the scaled signal and $\underset{\sim}{\mathbf{F}}_s$ is the matrix form of the filter. A single filter is used

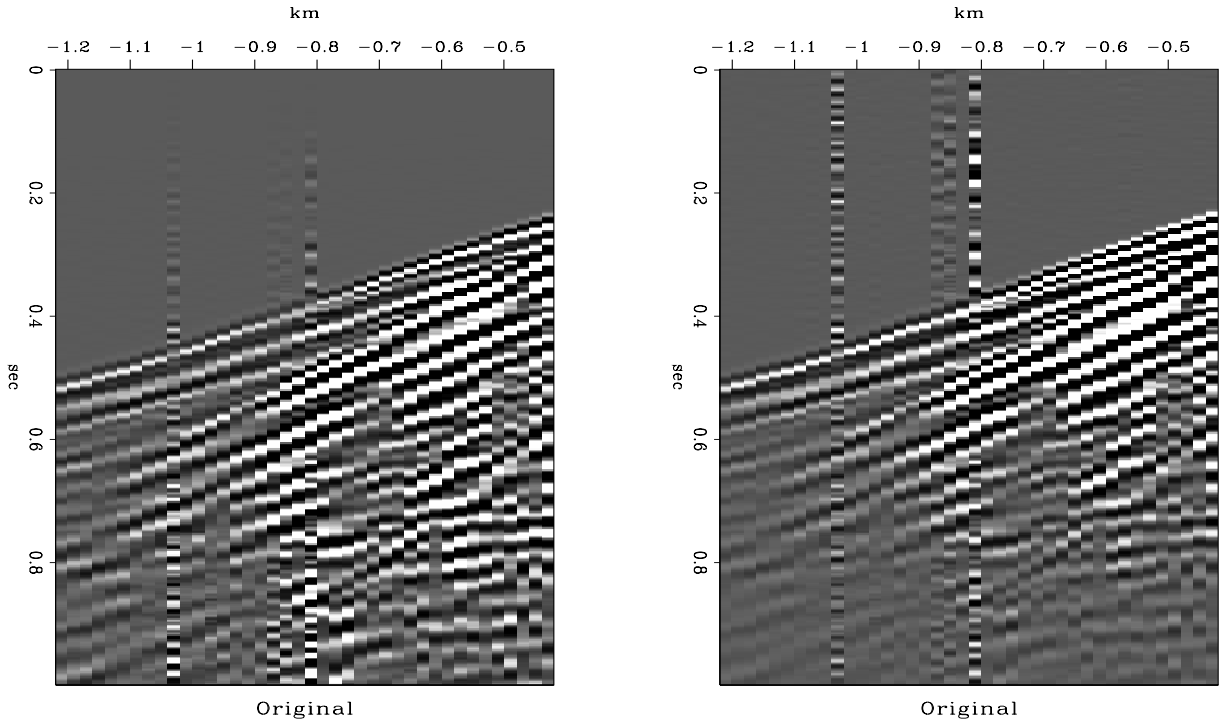

FIG. 10.1. A shot gather with noise. The plot on the left has t^2 scaling, the plot on the right does not. prestack-originalf [R]

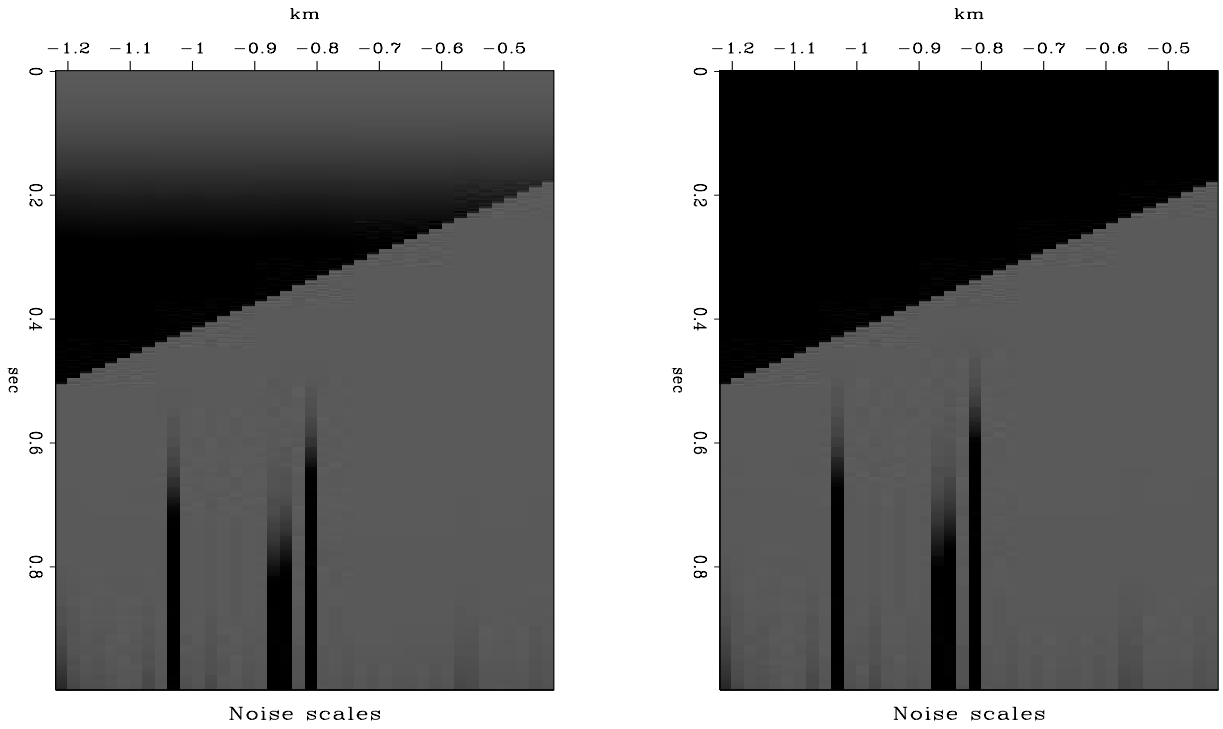

FIG. 10.2. The noise amplitudes based on the calculated RMS amplitudes of the signal and noise from equation 10.18. The plot on the left has t^2 scaling, the plot on the right does not. prestack-noisediag [R]

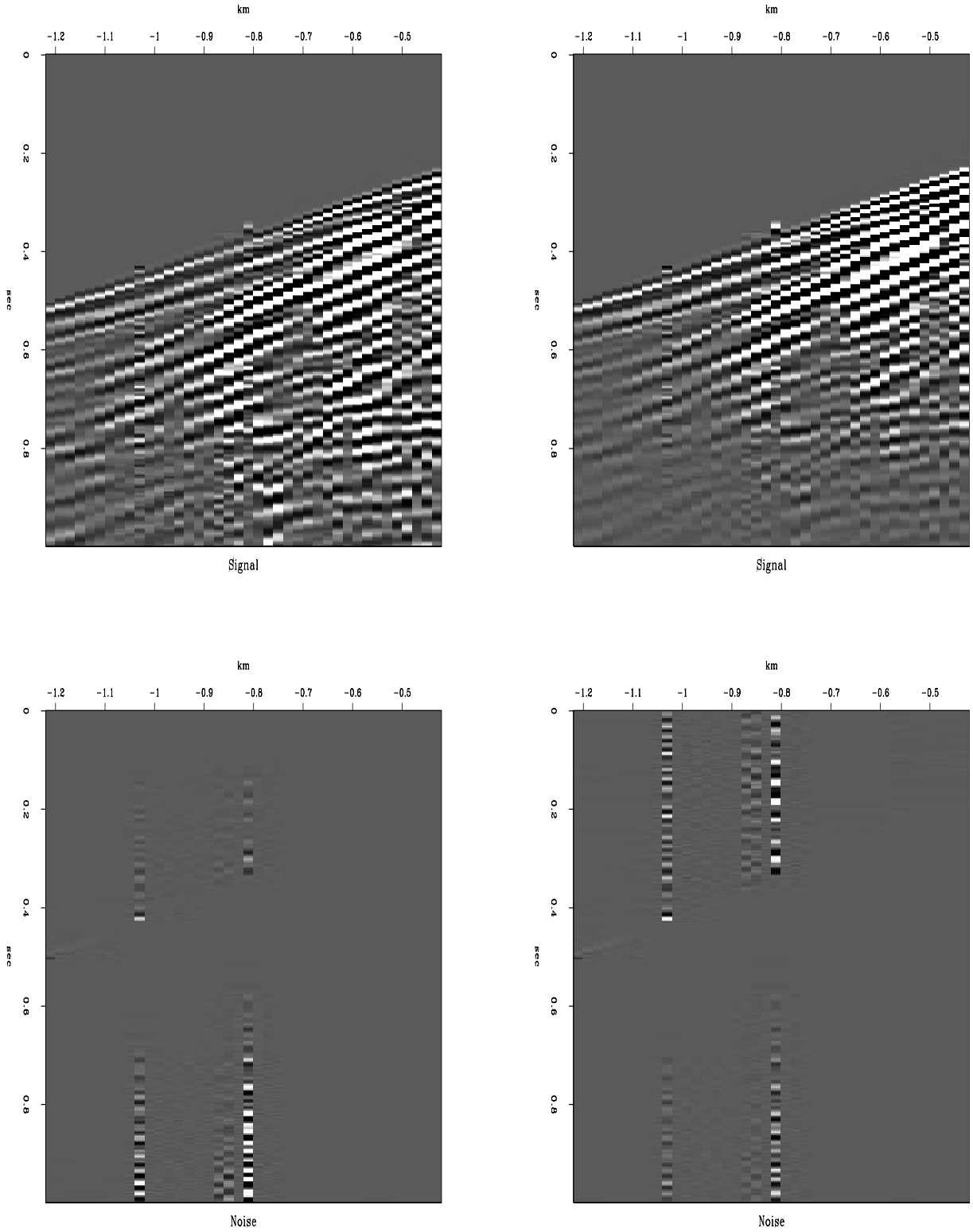

FIG. 10.3. The original data scaled by the scale factors for the noise and signal. The top row shows the data scaled by the signal scale factor. The bottom row shows the data scaled by the noise scale factor. The plot on the left has t^2 scaling, the plot on the right does not. prestack-aamp [R]

for the signal in all traces. The noise filter appears as

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_n(x)\mathbf{n}. \tag{10.20}$$

$\underset{\sim}{\mathbf{F}}_n(x)$ is a function of x because a separate filter is computed for each trace. For this section and the next, the noise filter is calculated on the data above the start times. In this section both these filters are one dimensional.

I would like to incorporate what I know about the amplitudes of the signal and noise from section 10.1 with the filters derived here to improve the separation of signal and noise. Since the signal loses amplitude as about t^2, I multiply the signal by t^2 and force the signal to zero above the start time by using the function $T'(x,t,v)$, which varies as t^2 below the start time and is a large fixed constant above the start time. The relative amplitudes of the noise and signal are also available from section 10.1. To minimize the signal and the noise together, I use

$$\mathbf{0} \approx \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v)\mathbf{s}, \tag{10.21}$$

and

$$\mathbf{0} \approx \frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x)\mathbf{n}. \tag{10.22}$$

Modifying equation (10.22) using $\mathbf{n} = \mathbf{d} - \mathbf{s}$ gives

$$\frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x)\mathbf{d} \approx \frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x)\mathbf{s}. \tag{10.23}$$

Expressing both systems as a single system gives

$$\begin{pmatrix} \mathbf{0} \\ \frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x)\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v) \\ \frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x) \end{pmatrix} \mathbf{s}. \tag{10.24}$$

An alternative approach is to solve for the noise. By repeating the same set of steps, but solving for the noise instead of the signal with $\mathbf{s} = \mathbf{d} - \mathbf{n}$, equation (10.21) becomes

$$\mathbf{0} \approx \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v)(\mathbf{d} - \mathbf{n}) \tag{10.25}$$

or

$$\frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v)\mathbf{d} \approx \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v)\mathbf{n}. \tag{10.26}$$

This expression, combined with equation (10.22), gives

$$\begin{pmatrix} \mathbf{0} \\ \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v)\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \frac{1}{\sigma_n(x)}\underset{\sim}{\mathbf{F}}_n(x) \\ \frac{1}{\sigma_s}\underset{\sim}{\mathbf{F}}_s T'(x,t,v) \end{pmatrix} \mathbf{n}. \tag{10.27}$$

The results of either system (10.24) or (10.27) may be used. Appendix A of Abma (1994) shows that the results of either system of equations will produce equivalent results when $\mathbf{d} = \mathbf{n} + \mathbf{s}$, but it can be seen that this does not apply when null spaces containing the signal and noise occur. Although it might be argued that the filters $\underset{\sim}{\mathbf{F}}_n(x)$ and $\underset{\sim}{\mathbf{F}}_s$ are never perfect and so actual null spaces are not created, given a reasonable number of iterations of the least-squares solver and the finite capabilities of the floating point number representation used, effective null spaces will be created in these cases. Since the filters $\underset{\sim}{\mathbf{F}}_n(x)$ and $\underset{\sim}{\mathbf{F}}_s$ are fairly effective in removing the noise and signal, they create effective null spaces in systems (10.24) and (10.27).

Another way of looking at this null space problem is to examine the information that is available in the system to calculate a solution. In system (10.24), any information about the noise is eliminated from the system since the filter $\underset{\sim}{\mathbf{F}}_n(x)$ has removed the noise from the data going into the left-hand side of the system. Therefore, the signal calculated from system (10.24) will not contain any information removed by $\underset{\sim}{\mathbf{F}}_n(x)$. If any information in the signal falls in the null space created by $\underset{\sim}{\mathbf{F}}_n(x)$, the solution for the signal from system (10.24) will not contain that information. In system (10.27), any information about the signal is eliminated from the system since the filter $\underset{\sim}{\mathbf{F}}_s$ has removed the signal from the data going into the left-hand side of the system. Therefore, the noise calculated from system (10.27) will not contain any information removed by $\underset{\sim}{\mathbf{F}}_s$. If any information in the noise falls in the null space created by $\underset{\sim}{\mathbf{F}}_s$, the solution for the noise from system (10.27) will not contain that information. In short, if the signal and noise have overlapping null spaces, the overlap is eliminated in systems (10.24) and (10.27).

Another aspect of solving these inversions involves initialization of the inversion (Abma, 1995). This initialization reduces the number of iterations of the solver significantly, thus reducing the cost of the inversion. It also modifies the action of the inversion. If system (10.24) has the signal initialized with the original data, any data in the common null space create by $\underset{\sim}{\mathbf{F}}_n(x)$ and $\underset{\sim}{\mathbf{F}}_s$ will not be removed from the signal. For system (10.27) initialized with the original data, any data in the common null space create by $\underset{\sim}{\mathbf{F}}_n(x)$ and $\underset{\sim}{\mathbf{F}}_s$ will not be removed from the noise. It can then be seen that the noise calculated from system (10.24) without initialization subtracted from the data will be equal to the signal calculated from system (10.27) with initialization. Also, the signal calculated from system (10.27) without initialization subtracted from the data will be equal to the noise calculated from system (10.24) with initialization.

For one-dimensional filters, the overlap of the action of $\underset{\sim}{\mathbf{F}}_n(x)$ and $\underset{\sim}{\mathbf{F}}_s$ is likely to be a problem, especially when both the signal and noise are broadband. One approach to solving the problem of this overlap is to modify the systems so that the effective null spaces do not occur. One method I have tried is to separate the amplitude and the filtering effects into separate equations. These equations were

$$\mathbf{0} \approx \frac{1}{\sigma_s}\mathbf{s}', \tag{10.28}$$

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_s\mathbf{s}', \tag{10.29}$$

$$\mathbf{0} \approx \frac{1}{\sigma_n(x)}\mathbf{n}, \tag{10.30}$$

and

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_n(x)\mathbf{n}. \tag{10.31}$$

Expanded with relative weights to predict the signal, these minimizations become the single system of regressions that follows:

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \lambda_3\frac{1}{\sigma_n}\mathbf{d} \\ \lambda_4\underset{\sim}{\mathbf{F}}_n\mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \lambda_1\frac{1}{\sigma_s}T'(x,t,v) \\ \lambda_2\underset{\sim}{\mathbf{F}}_sT'(x,t,v) \\ \lambda_3\frac{1}{\sigma_n(x)} \\ \lambda_4\underset{\sim}{\mathbf{F}}_n(x) \end{pmatrix}\mathbf{s}. \tag{10.32}$$

This system has no effective null spaces. Unfortunately, (10.28) and (10.30) assume that the signal and noise are constant functions, not the sinusoidal functions normally found in seismic data. While system (10.32) worked well for the peaks of the noise and signal, it failed near the zero crossings. Because of these failures, I will no longer consider approaches such as that in system (10.32) in this thesis.

10.2.2 1-D Separation examples

Figure 10.4 shows a very simple synthetic where the signal is a constant frequency sine wave beginning at the start time with an amplitude that varies as time squared. The noise is also a constant frequency sine wave, but has a different frequency than the signal. The noise also has a constant amplitude instead of the t^2 decrease in amplitude seen in the signal. There is a background of weak random noise.

Figure 10.5 shows the result of predicting the signal using equation (10.24). The trace that contained the noise is almost completely zeroed. This zeroing was caused by

the common effective null space of the noise and signal filters. In spite of the signal overwhelming the noise in the area the signal filter was designed in, the remaining noise was eliminated by the filter designed there. When applied to the real shot data, the noisy traces are not well separated, and the traces with noise are significantly weakened, as seen in Figure 10.6.

While other approaches may by taken, such as shortening the filters or attempting to completely zero the noise before designing the signal filter, separation of signal and noise using one-dimensional filters does not seem practical, mainly because of the overlap of the null spaces created by the signal and noise filters. In the next sections, I address these shortcomings by making the signal filter two-dimensional.

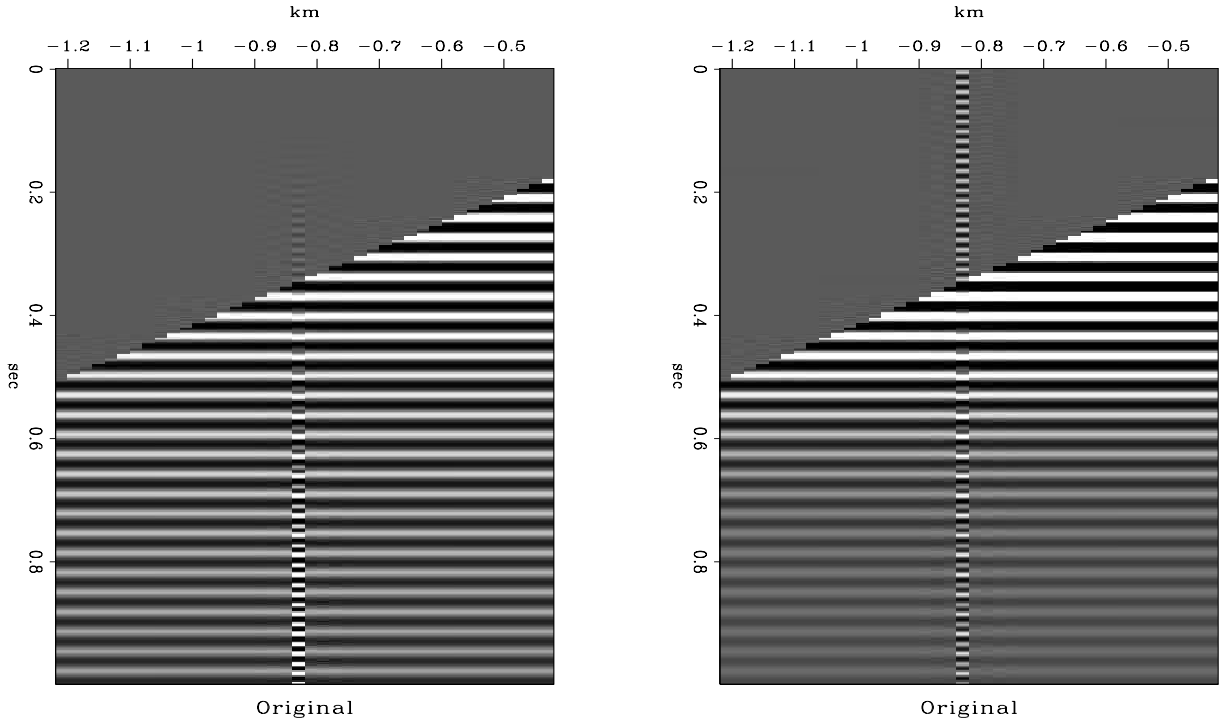

FIG. 10.4. A simple synthetic showing noise and signal with different spectra. The plot on the left has t^2 scaling, the plot on the right does not. prestack-sinefile [R]

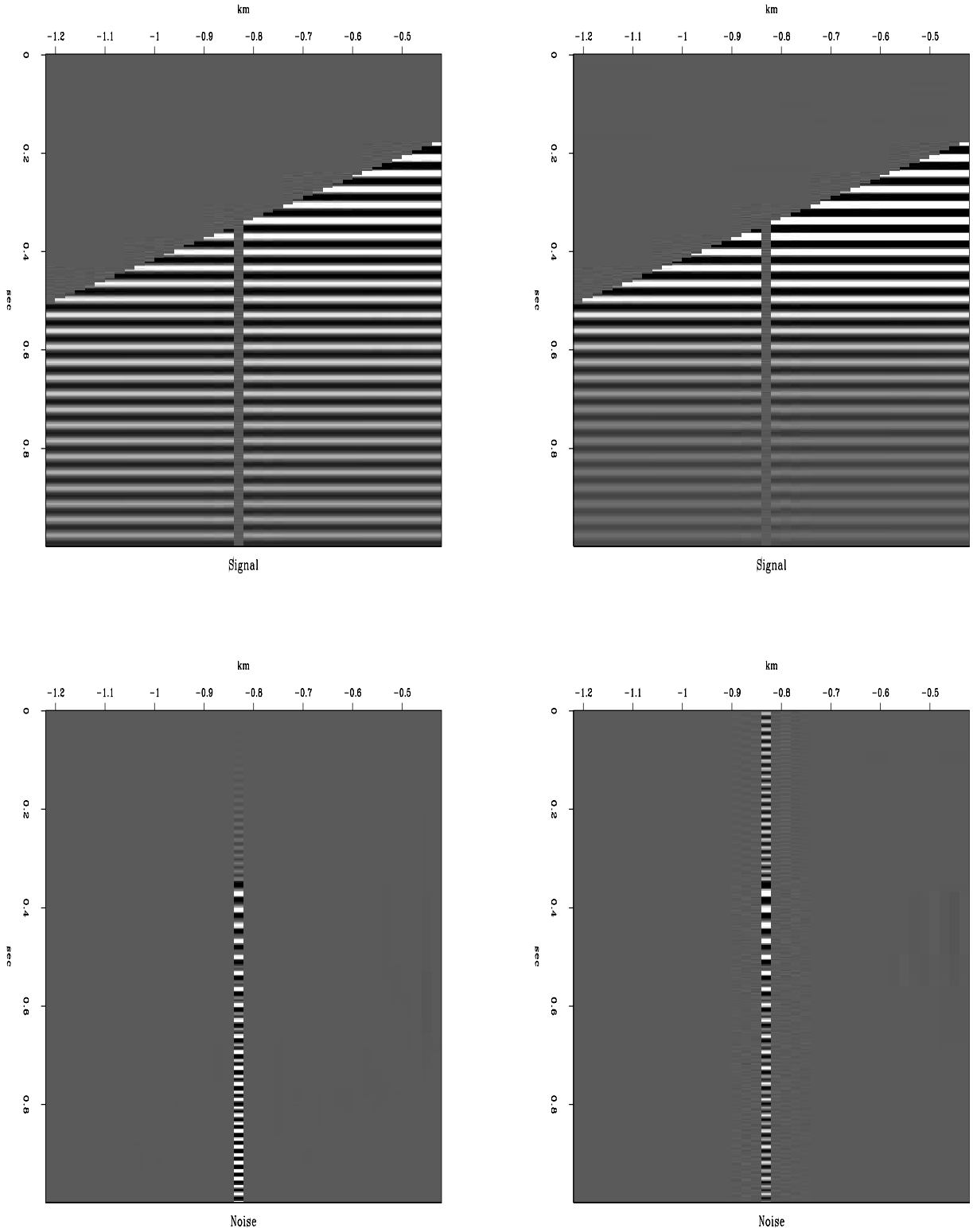

FIG. 10.5. A simple example of separation of signal and noise using sine waves of different frequencies. The method of equation (10.24) was used. The plot on the top is the calculated signal, and the plot on the bottom is the noise that remained. The plot on the left has t^2 scaling, the plot on the right does not. prestack-sf1w.b [R]

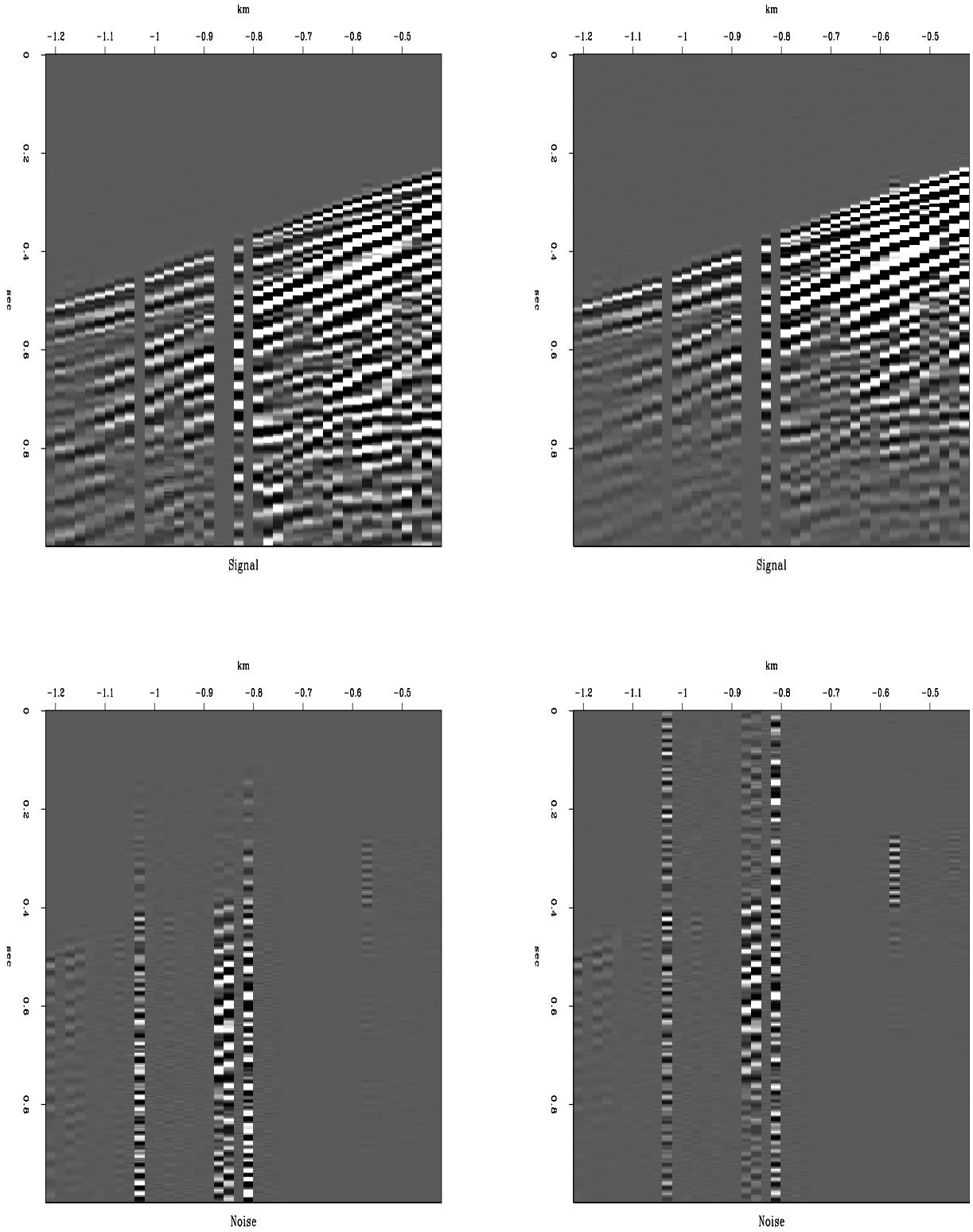

FIG. 10.6. A simple example of separation of signal and noise using the real data from Figure 10.1. The method of equation (10.24) was used. The plot on the top is the calculated signal, and the plot on the bottom is the noise that remained. The plot on the left has t^2 scaling, the plot on the right does not. prestack-sflw.a [R]

10.3 Separation by 2-D spectral signal estimation

10.3.1 2-D spectral signal estimation theory

To further enhance the characterization of the signal, a two-dimensional annihilation filter is used. This filter is represented in this section by

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_s \mathbf{s}, \tag{10.33}$$

where $\mathbf{s}'$ is the scaled signal and $\underset{\sim}{\mathbf{F}}_s$ is the matrix form of the filter. As before, I design a one-dimensional filter on each trace to annihilate the noise. This appears as

$$\mathbf{0} \approx \underset{\sim}{\mathbf{F}}_n(x) \mathbf{n}. \tag{10.34}$$

$\underset{\sim}{\mathbf{F}}_n(x)$ is a function of x because a separate filter is computed for each trace. At present, this filter is calculated on the data above the start times.

Since the 2-D filter is expected to have much different characteristics than the 1-D noise filter, the regressions are scaled to each other as follows:

$$\mathbf{0} \approx \lambda_1 \underset{\sim}{\mathbf{F}}_s \frac{1}{\sigma_s} \mathbf{s}', \tag{10.35}$$

$$\mathbf{0} \approx \lambda_2 \underset{\sim}{\mathbf{F}}_n(x) \frac{1}{\sigma_n(x)} \mathbf{n}. \tag{10.36}$$

It is interesting to note that the regression in equation (10.35) is just a weighted version of t-x prediction (Abma and Claerbout, 1993).

Expanded to predict the signal, these minimizations become the single system of regressions that follows:

$$\begin{pmatrix} \mathbf{0} \\ \lambda_2 \underset{\sim}{\mathbf{F}}_n \frac{1}{\sigma_n} \mathbf{d} \end{pmatrix} \approx \begin{pmatrix} \lambda_1 \underset{\sim}{\mathbf{F}}_s \frac{1}{\sigma_s} T'(x,t,v) \\ \lambda_2 \underset{\sim}{\mathbf{F}}_n(x) \frac{1}{\sigma_n(x)} \end{pmatrix} \mathbf{s}. \tag{10.37}$$

If the inversion of system (10.37) is attempted with the initial value of $\mathbf{s}$ being zero, many iterations of the solver are needed to produce a reasonable result. In Abma(1995), I showed that initializing the value of $\mathbf{s}$ to the result obtained from prediction-error filtering reduced the cost of the inversion and improved the results. Although the estimated signal from prediction-error filtering is not a perfect fit, it appears close enough to reduce the number of iterations by an order of magnitude, making this process a practical production technique.

10.3.2 2-D spectral signal estimation examples

Figure 10.7 shows the results of using equation (10.37) to predict the noise from the data from Figure 10.4. Notice that the result is significantly better than that shown in Figure 10.5, although some artifacts are seen near the first breaks. Almost no noise is seen in the signal section.

While the program using the 2-D filter version of equation (10.37) with the initial solution of zero required 250 iterations to get a reasonable result, the program using the initialized estimate needed only 25.

Figure 10.8 shows the results of using equation (10.37) to predict the signal from the data seen in Figure 10.1. The separation is also good, and only a little weakening of the data traces can be seen where the noise appears. A little of the signal that has not been perfectly predicted has leaked into the noise section. An improved result might be produced by using the output of this process as the input to another step of the same process, with the next step using the previous step's results as the first guess of the amplitude and noise. The new noise and amplitude scale factors might then be more accurate.

10.4 Separation by 2-D signal and noise spectral estimation

Coherent noise is often strong enough to interfere with the interpretation or analysis of seismic data. Coherent noises often encountered in reflection seismic data are ground roll, air blasts, and trapped waves. Typically, these noises show velocities that are different from the desired reflection events and may also have frequency spectra that can be used to distinguish them from the desired events. In this section I demonstrate a method to separate coherent noise and signal.

10.4.1 2-D signal and noise estimation theory

As before, the signal $\mathbf{s}$ is described by a two-dimensional signal annihilation filter $\underset{\sim}{\mathbf{F}}_s$, so that $\underset{\sim}{\mathbf{F}}_s\mathbf{s} \approx \mathbf{0}$. The noise $\mathbf{n}$ is described by a noise annihilation filter $\underset{\sim}{\mathbf{F}}_n$, so that $\underset{\sim}{\mathbf{F}}_n\mathbf{n} \approx \mathbf{0}$, where $\underset{\sim}{\mathbf{F}}_n$ is now a single two-dimensional filter. As before, the recorded data $\mathbf{d}$ is the sum of the signal $\mathbf{s}$ and noise $\mathbf{n}$, making $\mathbf{d} = \mathbf{s} + \mathbf{n}$.

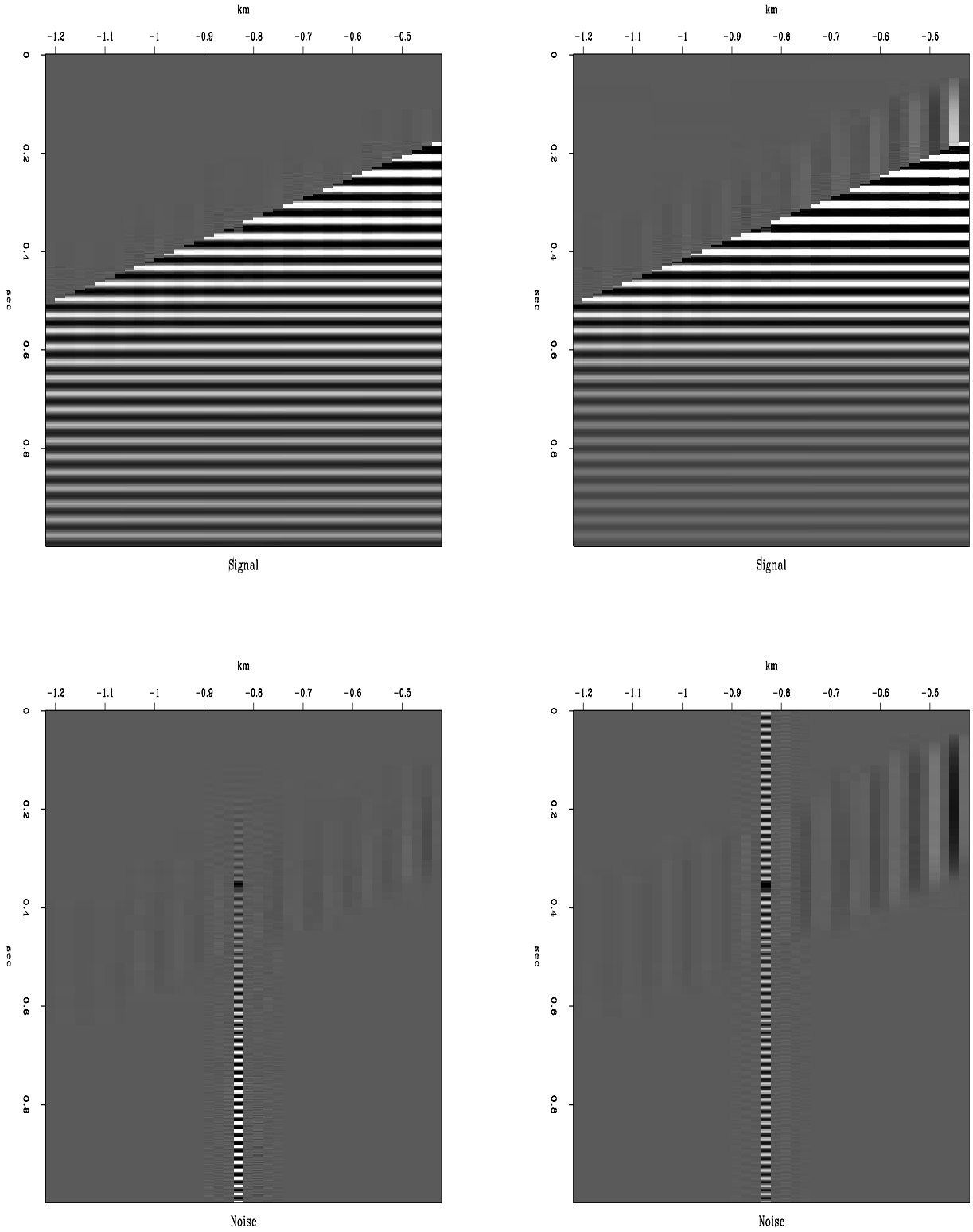

FIG. 10.7. A simple example of separation of signal and noise using sine waves of different frequencies. The method of equation (10.37) was used. The plot on the top is the calculated signal, and the plot on the bottom is the noise that remained. The plot on the left has t^2 scaling, the plot on the right does not. prestack-sf2wf.b [R]

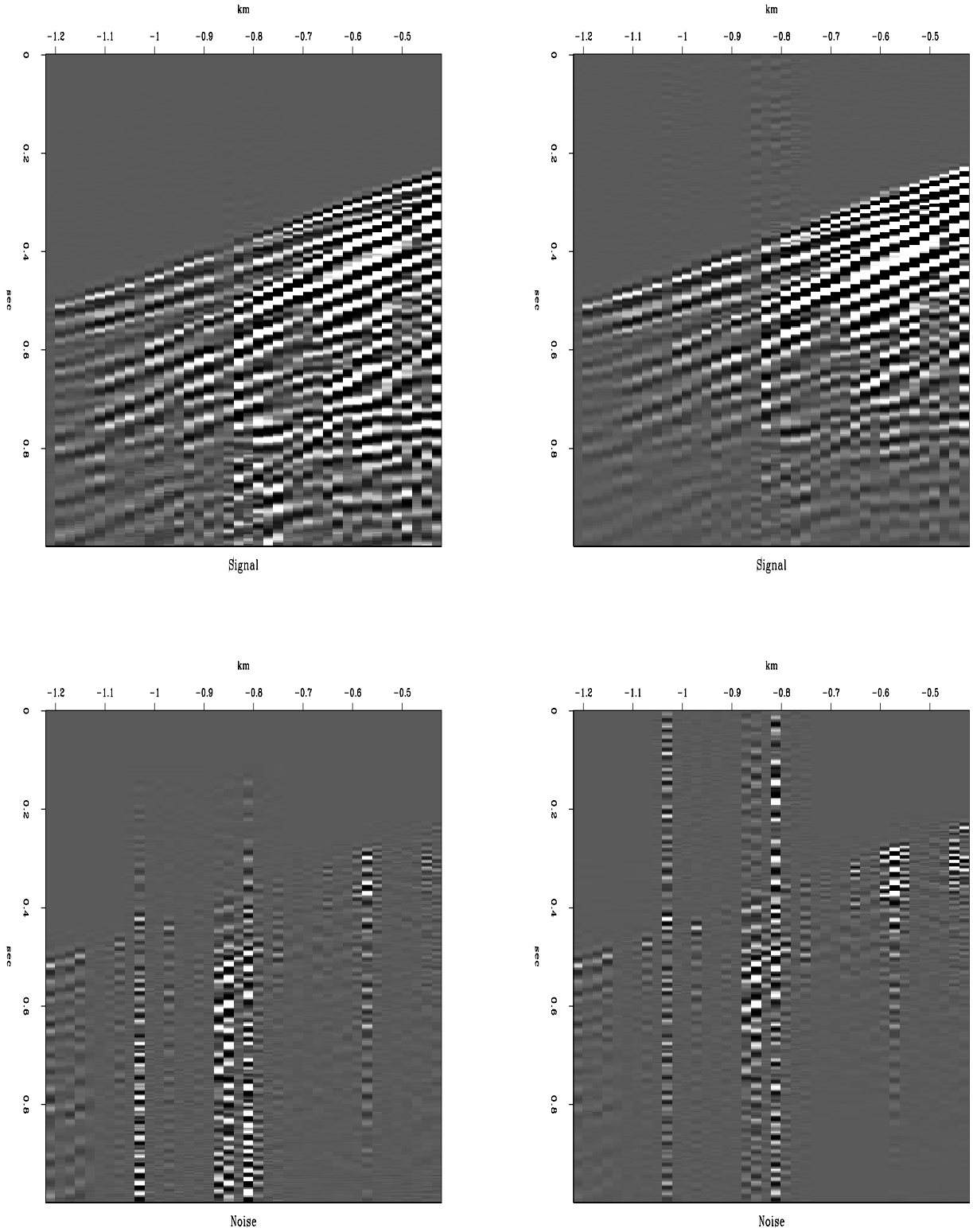

FIG. 10.8. A simple example of separation of signal and noise using real data. The method of equation (10.37) was used. The plot on the top is the calculated signal, and the plot on the bottom is the noise that remained. The plot on the left has t^2 scaling, the plot on the right does not. prestack-sf2wf.a [R]

The use of filters to characterize the signal and noise is similar to the methods used to separate the signal and noise in the simple three-dip cases shown in chapter 9. Here, both the signal and noise are characterized by two-dimensional filters, one filter for the noise and one filter for the signal. Although the signal and noise in the three-dip case considered in chapter 9 did not overlap spatially, in the general case the noise and signal will always overlap. Because of this overlap, calculating the signal filter $\underset{\sim}{\mathbf{F}}_s$ and the noise filter $\underset{\sim}{\mathbf{F}}_n$ may be a problem, since the signal $\mathbf{s}$ and noise $\mathbf{n}$ are unavailable before the separation takes place. A combination of two techniques is used to produce reasonable estimates of $\underset{\sim}{\mathbf{F}}_s$ and $\underset{\sim}{\mathbf{F}}_n$ here.

The first technique is to separate spatially the noise and the signal into regions where one or the other dominates. For example, in the case of the ground-roll noise considered later in this section, the noise may be isolated to a narrow wedge within a shot gather. The noise filter calculated over the data in this wedge will be influenced primarily by the ground roll. A similar technique is used to calculate the signal filter. After the ground roll and the first breaks are muted, the data is dominated by the signal, so that a filter calculated over this data will be influenced primarily by signal.

The second technique used to produce reasonable estimates of $\underset{\sim}{\mathbf{F}}_s$ and $\underset{\sim}{\mathbf{F}}_n$ is to control the shape of the filters to produce the desired prediction. The signal filter $\underset{\sim}{\mathbf{F}}_s$ is a purely lateral two-dimensional filter as described in chapter 4. If this filter is kept short in time, the steeply dipping ground roll becomes fairly difficult to predict. The noise filter $\underset{\sim}{\mathbf{F}}_n$, on the other hand, is shaped to follow the dip of the ground roll and has the form

$$\begin{matrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ a & a & a & 0 \\ a & a & a & 0 \\ a & a & a & 0 \\ a & a & a & 0 \end{matrix} , \tag{10.38}$$

where each of the as is a different filter coefficient. Notice that the output point is the upper right-hand side of the filter and there is a gap between the output point and the free filter coefficients. This gap prevents the prediction of reflection events that fall inside the window where the ground roll dominates. Also notice that the column that contains

the one has no free filter coefficients in it. This prevents using the samples from within a given trace to predict anything within that trace. This is needed since ground roll is often an almost mono-frequency noise. If the predictions within a trace are allowed, the filter $\underset{\sim}{\mathbf{F}}_n$ might become a simple frequency filter, and that frequency would also be removed from the signal. Not allowing predictions from within a trace also keeps the predictions from within the trace from overwhelming the predictions from the other traces. Since the noise is recognized by its coherence between traces, the trace-to-trace predictions must be maintained. The combination of the filter shapes and the spatial separation of the signal and noise provides good estimates of the signal and noise filters.

Since this method will generally be used on prestack data, either filter may be corrupted by high-amplitude noise. Following the method presented in chapter 8, high-amplitude noise may be removed by a trace-to-trace prediction method where samples that are not well predicted are thrown out. These missing data samples, as well as data not recorded, are then restored during the inversion for the signal and noise.

To review chapter 8, the data $\mathbf{d}$ is separated into the known data $\mathbf{k}$ and the missing data $\mathbf{m}$, so that $\mathbf{d} = \mathbf{k} + \mathbf{m}$. Two masks, $\underset{\sim}{\mathbf{K}}$ and $\underset{\sim}{\mathbf{M}}$, are used to describe the missing data, making $\mathbf{k} = \underset{\sim}{\mathbf{K}}\mathbf{d}$ and $\mathbf{m} = \underset{\sim}{\mathbf{M}}\mathbf{d}$, where $\underset{\sim}{\mathbf{K}} + \underset{\sim}{\mathbf{M}} = \underset{\sim}{\mathbf{I}}$, where $\underset{\sim}{\mathbf{I}}$ is the identity matrix.

Summarizing the previous definitions:

$\mathbf{d}$ = data

$\mathbf{s}$ = signal

$\mathbf{n}$ = noise

$\mathbf{k}$ = known data

$\mathbf{m}$ = missing data

$\underset{\sim}{\mathbf{K}}$ = known data mask

$\underset{\sim}{\mathbf{M}}$ = missing data mask

$\underset{\sim}{\mathbf{F}}_s$ = signal annihilation filter

$\underset{\sim}{\mathbf{F}}_n$ = noise annihilation filter.

The relationships between these factors are as follows:

$\underset{\sim}{\mathbf{F}}_s\mathbf{s} \approx \mathbf{0}$

$\underset{\sim}{\mathbf{F}}_n\mathbf{n} \approx \mathbf{0}$

$\mathbf{d} = \mathbf{s} + \mathbf{n}$

$\mathbf{d} = \mathbf{k} + \mathbf{m}$

$\underset{\sim}{\mathbf{I}} = \underset{\sim}{\mathbf{K}} + \underset{\sim}{\mathbf{M}}$ or $\mathbf{d} = \underset{\sim}{\mathbf{K}}\mathbf{d} + \underset{\sim}{\mathbf{M}}\mathbf{d}$.

Taking $\underset{\sim}{\mathbf{F}}_s\mathbf{s} \approx \mathbf{0}$ and $\underset{\sim}{\mathbf{F}}_n\mathbf{n} \approx \mathbf{0}$ and replacing $\mathbf{s}$ with $\mathbf{d} - \mathbf{n}$ produces the system

$$\begin{pmatrix} \underset{\sim}{\mathbf{F}}_s\mathbf{d} \\ \mathbf{0} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{F}}_s \\ \underset{\sim}{\mathbf{F}}_n \end{pmatrix} \mathbf{n}. \tag{10.39}$$

Replacing the data $\mathbf{d}$ with $\mathbf{k} + \mathbf{m}$ and moving the terms that depend on the missing data $\mathbf{m}$ to the right-hand side gives

$$\begin{pmatrix} \underset{\sim}{\mathbf{F}}_s\underset{\sim}{\mathbf{K}}\mathbf{d} \\ \mathbf{0} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{F}}_s & -\underset{\sim}{\mathbf{F}}_s\underset{\sim}{\mathbf{M}} \\ \underset{\sim}{\mathbf{F}}_n & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ \mathbf{m} \end{pmatrix}. \tag{10.40}$$

Finally, system (10.40) is modified to weight the noise prediction so that the noise tends to fall only in the zone where the noise is expected. In this case, the weighting is done by a simple set of weights, $\underset{\sim}{\mathbf{W}}_s$ and $\underset{\sim}{\mathbf{W}}_n$, $\underset{\sim}{\mathbf{W}}_s$ being small where signal is expected and large where only noise is expected, and $\underset{\sim}{\mathbf{W}}_n$ being small where noise is expected and large where only signal is expected. The modified system then becomes

$$\begin{pmatrix} \underset{\sim}{\mathbf{F}}_s\underset{\sim}{\mathbf{K}}\underset{\sim}{\mathbf{W}}_s\mathbf{d} \\ \mathbf{0} \end{pmatrix} \approx \begin{pmatrix} \underset{\sim}{\mathbf{F}}_s\underset{\sim}{\mathbf{W}}_s & -\underset{\sim}{\mathbf{F}}_s\underset{\sim}{\mathbf{M}}\underset{\sim}{\mathbf{W}}_s \\ \underset{\sim}{\mathbf{F}}_n\underset{\sim}{\mathbf{W}}_n & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ \mathbf{m} \end{pmatrix}. \tag{10.41}$$

In the example shown here, the weight $\underset{\sim}{\mathbf{W}}_n$ has been set to follow the ground roll by a single velocity and a single window length. An alternative approach that might work well when the ground roll is very strong would be to make $\underset{\sim}{\mathbf{W}}_n$ the inverse of the envelope of the data. This weighting function can also be used to describe the spatial extent of the noise for calculating the noise and signal filters. For signal and noise that vary as functions in time, the two sets of weights can become functions such as the t and t^2 in section 3.2.3.

To reduce the numbers of iterations required to solve system (10.41), the initial estimate of $\mathbf{n}$ is set to be the wedge of data dominated by noise filtered by the signal annihilation filter $\underset{\sim}{\mathbf{F}}_s$ to remove the signal that underlies the noise. This estimate reduces the number of iterations and improves the final estimate of $\mathbf{n}$ significantly.

10.4.2 2-D signal and noise estimation examples

Figure 10.9 shows a shot gather with strong coherent noise at a velocity of about 1800 feet per second. This noise train is relatively strong and narrow so it is easy to separate spatially. For this file, the noise filter was calculated from the data in a window 0.4 seconds long with a starting velocity of 1800 feet per second. This window can be seen from the

distribution of the background noise seen around the ground roll in Figure 10.10. The signal filter was calculated from the data with the previous window zeroed out, as well as having a start time mute of 3900 feet per second.

The most expensive part of this process is calculating the noise filter $\mathbf{F}_n$, since it is a fairly large filter. Using a smaller filter reduced the effectiveness of the process, probably because the noise train is more complicated than it appears. If the noise is consistent from shot to shot, the noise filter might be reused to reduce the cost. Calculating an effective noise filter is likely to be a problem on many land lines, since the noise will vary from shot to shot, and since the coupling of the receivers to the ground is variable. This variability of the coupling will show up as unpredictable parts of the ground roll. Since the ground roll is very strong when compared to the signal, the unpredictable part of the ground roll is likely to have significant energy. One method of correcting for the variable coupling was presented by Berlioux and Lumley(1994). If the variations in the coupling are not corrected for, the results of this process are likely to be unsatisfactory. The shot file shown in Figure 10.9 appears to have an unusually uniform receiver coupling.

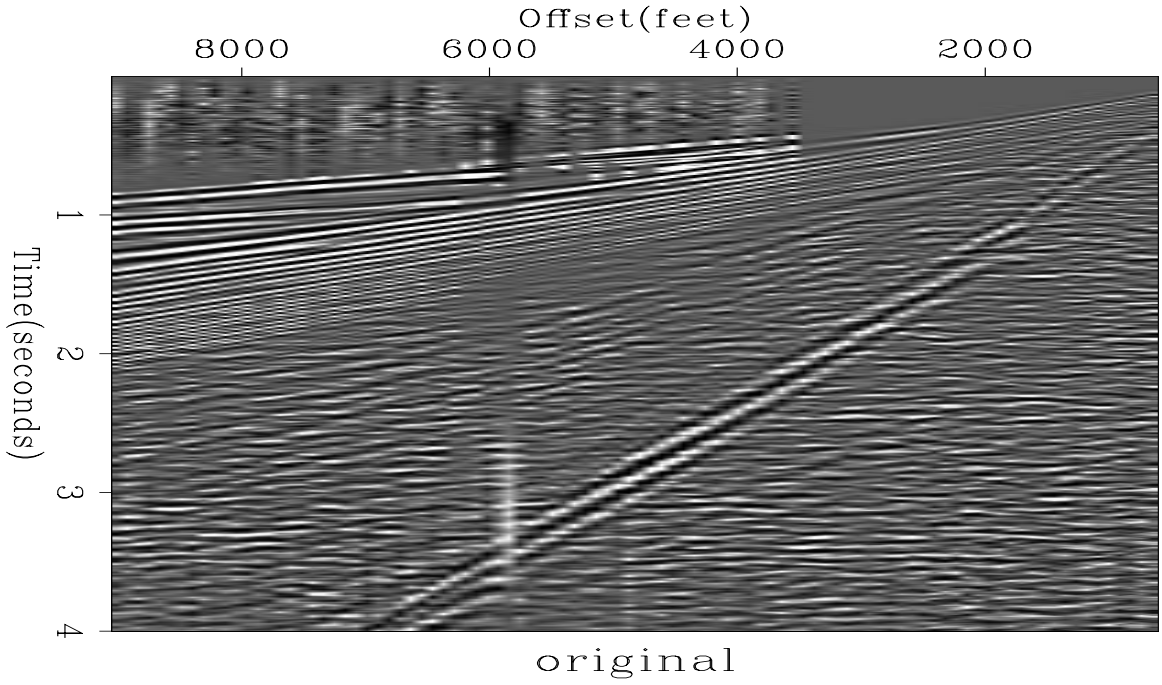

FIG. 10.9. A shot gather showing strong ground roll. prestack-WFshot [R]

Once the signal and noise filters were calculated, system (10.41) was inverted for the

noise and the missing data. As mentioned above, the noise was initialized with the noise window filtered by the signal annihilation filter. If the inversion was attempted with the noise initialized to zero, many iterations were required to get a reasonable result. Even with many iterations, the result was not as good as using a few iterations with the noise initialized with a good estimate. For the results shown here, only ten iterations were used. Little improvement was found when using more iterations.

Figure 10.10 shows the noise estimated by the inversion. Notice that there is little energy outside the zone where the noise dominated. There is no obvious signal showing in the noise section. When the noise in Figure 10.10 is subtracted from the original data in Figure 10.9, the resulting signal is seen in Figure 10.11. Almost all of the coherent noise was removed.

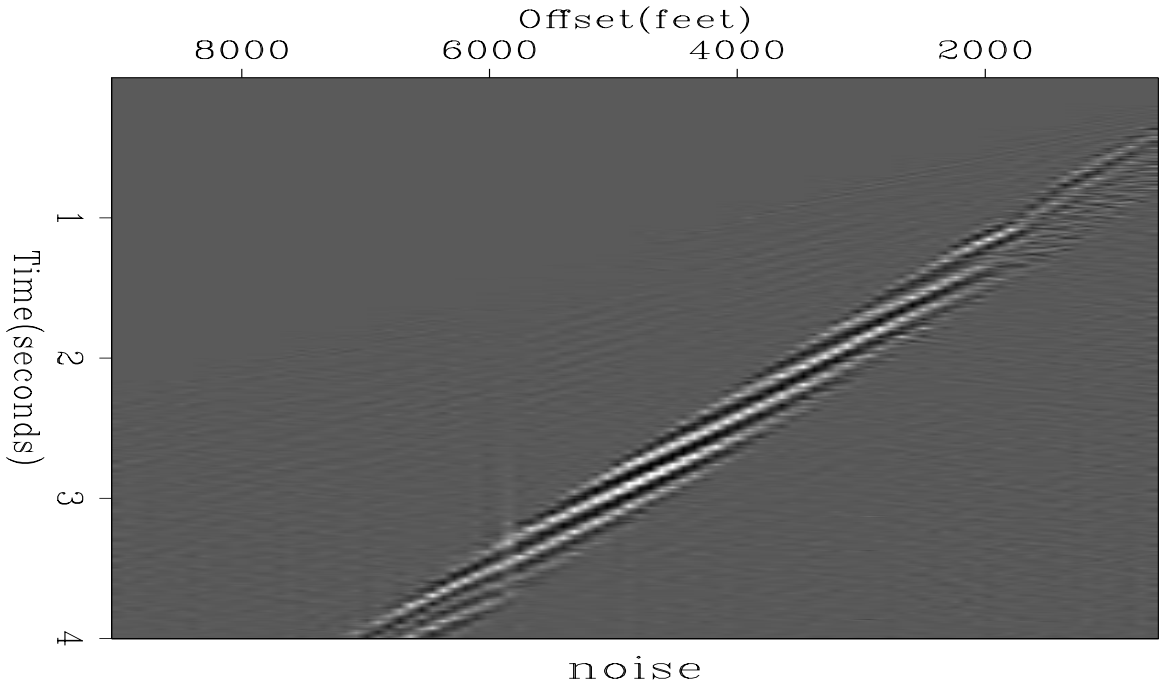

FIG. 10.10. The noise estimated from the inversion of system (10.41) using the data from the previous figure. prestack-condif [CR]

There is a small change in the data character of Figure 10.11 between the zone where the noise originally dominated and the area outside the zone. This was the result of the treatment of noise not predicted by either the signal filter $\underset{\sim}{\mathbf{F}}_s$ or the noise filter $\underset{\sim}{\mathbf{F}}_n$, especially random noise. The distribution of unpredicted noise is controlled by the weighting

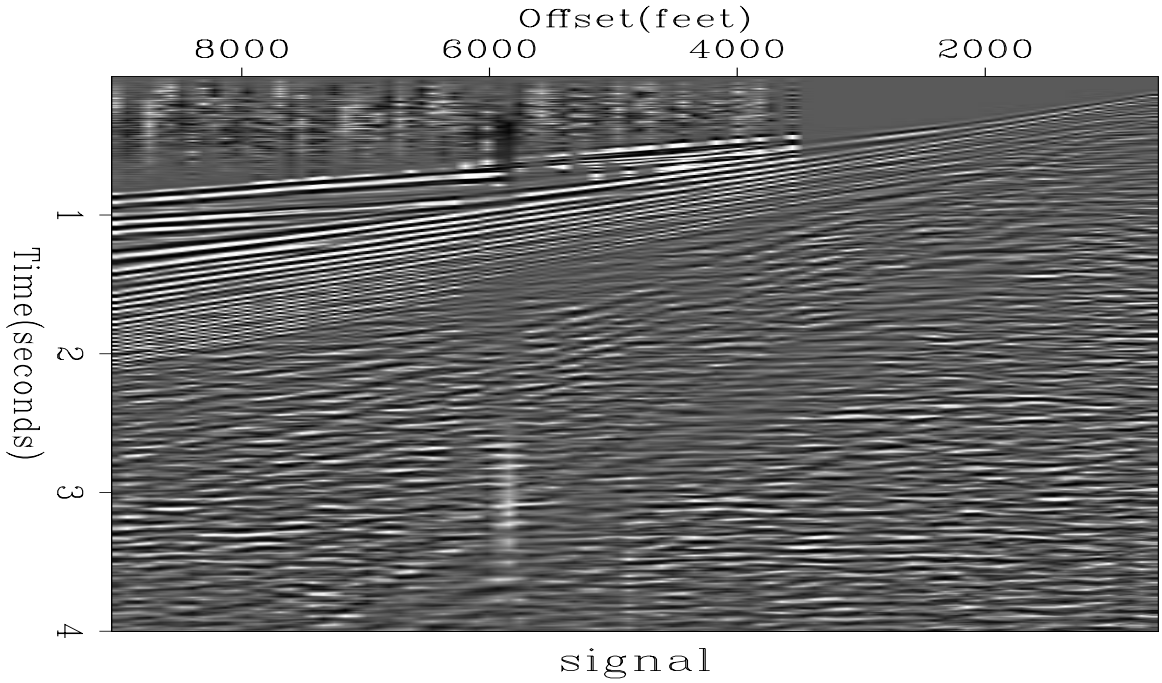

FIG. 10.11. The estimated signal obtained by subtracting the noise from the original data. prestack-cohinv [CR]

functions $\underset{\sim}{\mathbf{W}}_s$ and $\underset{\sim}{\mathbf{W}}_n$, as shown in chapter 9. Since the value of $\underset{\sim}{\mathbf{W}}_n$ was small in the noise zone, most of the unpredictable noise fell there. Outside the noise zone, the value of $\underset{\sim}{\mathbf{W}}_n$ was large, keeping the unpredictable noise out of the noise section and putting it into the signal section. If this change of character affects analysis of the data, removing random noise from the entire file using the techniques of chapter 8 will eliminate the effect.

The values of the weighting functions $\underset{\sim}{\mathbf{W}}_s$ and $\underset{\sim}{\mathbf{W}}_n$ will also control the distribution of events that are predictable with both $\underset{\sim}{\mathbf{F}}_s$ and $\underset{\sim}{\mathbf{F}}_n$, as shown in chapter 9. If an event is equally predictable with either $\underset{\sim}{\mathbf{F}}_s$ or $\underset{\sim}{\mathbf{F}}_n$, the event will fall into the noise zone because of the small values of $\underset{\sim}{\mathbf{W}}_n$ there, while outside the noise zone, equally predictable events will tend to fall into the signal. Generally, events are unlikely to be well predicted by both $\underset{\sim}{\mathbf{F}}_s$ and $\underset{\sim}{\mathbf{F}}_n$ because of the different shapes of the filters.

The results of the inversion were relatively insensitive to the exact values used in $\underset{\sim}{\mathbf{W}}_s$ and $\underset{\sim}{\mathbf{W}}_n$. As long as the values outside the noise zone were between 8 and 40 times the values inside the noise zone, the results appeared basically the same. This insensitivity simplifies the calculation of $\underset{\sim}{\mathbf{W}}_s$ and $\underset{\sim}{\mathbf{W}}_n$ considerably, since the signal can be corrected

for spherical spreading before the process, while the noise, even though it will not have its amplitude perfectly represented in $\mathbf{W}_n$, can be removed effectively.

As a comparison, an F-K filter was run on the previous data, muting out the noise in the F-K domain. The results, as shown in Figure 10.12, appear similar to the inversion results, although some of the ground roll is left in the shallow section in the F-K plot. The inversion result shows more character change than the F-K result in the area where the ground roll originally dominated, as discussed above. The F-K filter shows some artifacts of the mute, which is expected since the mute will produce a long impulse response in the time-space domain. Even when the inversion method uses a long filter, the response of this filter is removed from the output. In cases where the noise is aliased over a significant part of its bandwidth, the inversion may be able to characterize and remove noise more effectively than simple F-K muting.

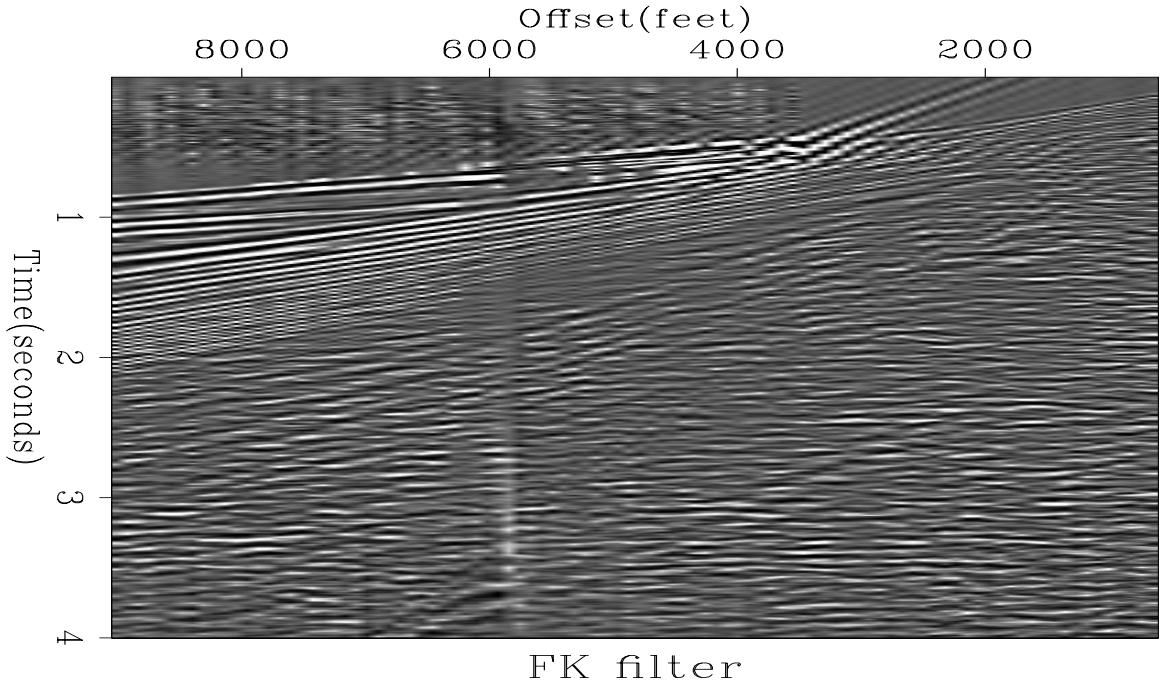

FIG. 10.12. An F-K filter of the previous example. prestack-fk [R]

10.5 Discussion

The results of separating noise and signal using amplitude and spectral estimations appear promising. In the cases considered here, the noise and signal can be separated fairly well, especially in the case where both the noise filter and the signal filter were two-dimensional.

I found that if, before entering the conjugate-gradient solver, the desired result is initialized to a good approximation of the desired result, the number of iterations required by the solver is reduced tremendously. The resulting reduction of the cost makes this technique a practical noise removal method.

In removing coherent noise, I found that using filters of different shapes to characterize the signal and noise worked well. These shaped filters, along with the amplitude weighting, did a good job of removing the coherent noise without harming the signal. The reflection data was well preserved both within the zone where the noise was removed and outside the zone.

Chapter 11

Conclusions

This thesis covers two methods of separating signal and noise using multidimensional filters. The first method is simple multidimensional filtering. The second is using multidimensional filters to characterize the signal and noise in an inversion process.

As a preparation for the application of these two methods, a data editing step to remove high-amplitude noise from the data is presented in chapter 2. This removal of the worst noise improves the calculation of the filter for simple multidimensional filtering and prevents the noise from overwhelming the inversion when using filters to characterize the signal and noise. The muted data may also be predicted later during the inversion for signal and noise.

Filters can be designed to characterize the signal or noise they predict by either changing the filter shape or by limiting the data the filter is designed from to that data dominated by either the signal or the noise. An example of modifying the filter shape is seen in chapter 4, where the filters used for predicting laterally continuous signals are purely lateral filters, since predictions within a trace do not necessarily apply to nearby traces. Another example of a filter shaped to predict a characteristic noise is seen in chapter 10, where ground roll is predicted using a filter with a large gap to avoid predicting reflections.

Although other methods may be used, the filters in this thesis were almost always calculated using a conjugate-gradient routine. This method simplifies and generalizes the filter calculation process, since the conjugate-gradient technique only requires the filtering process and its adjoint, the cross-correlation, to be specified. Filters with dimensions higher than one are easily calculated without the need to build the autocorrelation matrices for the higher dimensional filters.

For the signal and noise separation by simple filtering, I demonstrated in chapter 5 that the f-x prediction technique of Canales (1984) and Gulunay (1986) is equivalent to t-x prediction using a purely lateral filter with a very long time-length. This long time-length allowed more random noise to be passed into the signal section; it also allowed spurious events to be generated far from the original reflection signals.

For both t-x and f-x prediction on three-dimensional data, three-dimensional prediction was best done with a three-dimensional filtering process. Two passes of two-dimensional prediction tended to smooth out details that were preserved with the three-dimensional prediction filtering process.

Simple filtering to separate signal and noise has several shortcomings. The most serious is the generation of spurious events. While the spurious events generated by f-x prediction are more obvious since they are well separated in time, even the t-x prediction generates spurious events, although these spurious events are constrained to be local. These often appear as distortions in the wavelet of reflections. In both f-x and t-x predictions, spurious events are caused by the corruption of the filter used to predict the signal by noise in the data from which the filter is predicted. This corruption also reduces the amplitude of reflection events, since they are imperfectly predicted. By recalculating a filter from an estimate of the signal obtained from a previous pass of the inversion, an improved filter is obtained. This improved filter may then be used to obtain a better separation of the signal and noise.

Even with a perfect filter, simple prediction-error filtering does not produce the desired separation of signal and noise. This is because a perfectly predictable signal $\mathbf{s}$ removed with a signal-annihilation filter $\underset{\sim}{\mathbf{S}}$ so that $\underset{\sim}{\mathbf{S}}\mathbf{s} = \mathbf{0}$ gives $\underset{\sim}{\mathbf{S}}\mathbf{d} = \underset{\sim}{\mathbf{S}}\mathbf{n}$, not the assumed prediction-error filtering result of $\underset{\sim}{\mathbf{S}}\mathbf{d} = \mathbf{n}$. In short, the response of the noise to the filter is left in the signal calculated by prediction filtering. The inversion eliminates this response of the noise to the filter from the calculated signal. For low amplitude noise, the response of the noise to the filter will be small enough to ignore, but for high amplitude noise, this response will be significant.

The random noise removal problem described in chapters 4 and 5 as a simple prediction-error filtering process can be changed to an effective inversion process by using the prediction-error filter result as a starting point and a stabilizer for the inversion, as shown in chapter 7. Using the prediction-error filter result as an initial guess for the inversion

significantly reduces the number of iterations needed by the inversion routine, thus making the process only moderately more expensive than prediction-error filtering. Using the prediction-error filter result as a stabilization also keeps noise out of the signal section when the signal is not perfectly predicted by an imperfect signal-annihilation filter.

The inversion process of the previous paragraph may be extended to allow for missing data, as shown in chapter 8. This is a useful feature, since bad samples that have amplitudes high enough to spoil the least-squares inversion process may be removed from the data as show in chapter 2 and then recovered by the inversion. Also, unrecorded data may be predicted by the inversion. Prestack data tends to have more problems with both high-amplitude bad samples and unrecorded data than the poststack data these techniques are generally applied to. The inversion allows random noise removal to be extended to these prestack cases that were previously unsuited to prediction-error filtering methods.

Finally, when both the signal and the noise can be characterized by filters, inversions can be set up to take advantage of the predictability. Chapters 9 and 10 describe some applications of such inversions. Perhaps the most impressive application is that of the removal of coherent noise such as ground roll, as shown in chapter 10.

The techniques described in this thesis are not limited to the problems addressed here. In particular, the problem of multiple attenuation might be attacked using these techniques (Taner et al., 1995). The methods demonstrated here may also be extended from the two- and three-dimensional forms shown here to three-, four-, or five-dimensional applications corresponding to three-dimensional seismic acquisition with either simple inline acquisition or with full three-dimensional prestack geometries. In general, these extensions are simply a matter of extending the convolution routines to higher dimensions. Although the cost of higher-dimensional processing increases with the number of dimensions, and the amount of data recorded in some direction may not be sufficient to allow useful predictions, increasing the dimensionality of the predictions allows more data to be used and better predictions to be made.

Bibliography

Abma, R., and Claerbout, J. F., 1993, Lateral prediction for noise attenuation by t-x and f-x techniques: SEP–**79**, 13–30.

Abma, R., and Claerbout, J. F., 1994, Signal and noise separation applications: SEP–**82**, 213–234.

Abma, R., 1993, Lateral prediction techniques: FX-decon versus two-D deconvolution: SEP–**77**, 257–270.

Abma, R., 1994, Spurious event generation with f-x and t-x prediction: SEP–**82**, 235–244.

Abma, R., 1995, Enhanced prestack noise removal: SEP–**84**, 351–356.

Arfken, G., 1985, Mathematical Methods for Physicists, Third Edition: Academic Press, Inc, San Diego.

Berlioux, A., and Lumley, D., 1994, Amplitude balancing for AVO analysis: SEP–**80**, 349–360.

Bracewell, R. N., 1978, The Fourier Transform and its Applications, Second Edition: McGraw-Hill Book Co., New York.

Briggs, W. L., and Henson, V. E., 1995, The DFT: An Owner's Manual for the Discrete Fourier Transform: Society for Industrial and Applied Mathematics, Philadelphia.

Canales, L. L., 1984, Random noise reduction: 54th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 525–527.

Chase, M. K., 1992, Random noise reduction by 3-D spatial prediction filtering: 62nd Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1152–1153.

Childers, D. G., 1978, Modern spectrum analysis: The Institute of Electrical and Electronics Engineers, New York.

Claerbout, J. F., and Abma, R., 1994, Signal and noise separation fundamentals: SEP–**82**, 209–212.

Claerbout, J. F., 1985, Imaging the Earth's Interior: Blackwell Scientific Publications.

Claerbout, J. F., 1992a, Earth soundings analysis: Processing versus inversion: Blackwell Scientific Publications.

Claerbout, J. F., 1992b, Nonstationarity and conjugacy: Utilities for data patch work: SEP–**73**, 391–400.

Claerbout, J. F., 1993, Steep-dip deconvolution: SEP–**77**, 245–256.

Claerbout, J. F., 1995, Applications of Three-Dimensional Filtering: Environmental soundings image enhancement: Stanford Exploration Project
A preliminary version of this book is available on the Internet in PostScript form using a WWW browser at http://sepwww.stanford.edu/sep/prof/, or in part in the Stanford Exploration Project reports SEP-82 and SEP-83.

Darche, G., 1989, Iterative L1 deconvolution: SEP–**61**, 281–302.

Dobrin, M. B., and Savit, C. H., 1988, Introduction to Geophysical Prospecting: McGraw-Hill Book Co., New York.

Dongarra, J. J., Moler, C. B., Bunch, J. R., and Steward, G. W., 1979, Linpack users guide: Society for Industrial and Applied Mathematics.

Golub, G. H., and Van Loan, C. F., 1989, Matrix computations: John Hopkins University Press.

Green, P. J., 1984, Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives: Journal of the Royal Statistical Society B, **46**, no. 2, 149–192.

Gulunay, N., Sudhaker, V., Gerrard, C., and Monk, D., 1993, Prediction filtering for 3-D poststack data: 63rd Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1183–1186.

Gulunay, N., 1986, FXDECON and complex Wiener prediction filter: 56th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 279–281.

Hornbostel, S., 1991, Spatial prediction filtering in the t-x and f-x domains: Geophysics, **56**, 2019–2026.

Kanasewich, E. R., 1981, Time Sequence Analysis in Geophysics: University of Alberta Press, Edmonton, Alberta, Canada.

Kanasewich, E. R., 1990, Seismic Noise Attenuation: Pergamon Press, New York.

Korn, G. A., and Korn, T. M., 1968, Mathematical handbook for scientists and engineers: McGraw-Hill Book Co., New York.

Larner, K., Chambers, R., Yang, M., Lynn, W., and Wai, W., 1983, Coherent noise in marine seismic data: Geophysics, **48**, no. 7.

Linville, A. F., and Meek, R. A., 1992, Canceling stationary sinusoidal noise: Geophysics, **57**, no. 11, 1493–1501.

Luenburger, D. G., 1984, Linear and Nonlinear Programming: Addison-Wesley Publishing Company, Reading, Mass.

Makhoul, J., 1975, Linear Prediction: A Tutorial Review: Proc. IEEE, **63**, 561–580.

Menke, W., 1989, Geophysical Data Analysis; Discrete Inverse Theory, Revised Edition: Academic Press, Inc., Harcourt Brace Jovanovich Publishers, San Diego.

Nichols, D., 1994a, A simple example of a null space and how to modify it: SEP–**82**, 177–182.

Nichols, D., 1994b, Velocity-stack inversion using $\mathbf{L_p}$ norms: SEP–**82**, 1–16.

Papoulis, A., 1984, Probability, random variables, and stochastic processes, second edition: McGraw-Hill Book Co., New York.

Parker, R. L., 1994, Geophysical inverse theory: Princeton University Press, Princeton.

Pokhriyal, S. K., Nautiyal, A., and Gupta, O. P., 1991, Computer aided editing of seismic data: 61st Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1452–1456.

Pokhriyal, S. K., 1993, An efficient algorithm for automatic seismic data editing: 63st Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1196–1200.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., 1986, Numerical Recipes – The Art of Scientific Computing: Cambridge University Press.

Robinson, E. A., and Treitel, S., 1980, Geophysical Signal Analysis: Prentice-Hall, Englewood Cliffs, N.J.

Soubaras, R., 1994, Signal-preserving random noise attenuation by the f-x projection: 64th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1576–1579.

Stone, D. G., 1994, Designing Surveys in Two and Three Dimensions: Society of Exploration Geophysicists, Tulsa.

Strang, G., 1986, Introduction to Applied Mathematics: Wellesley-Cambridge Press, Wellesley, Mass.

Strang, G., 1988, Linear Algebra and its Applications: Harcourt Brace Jovanovich Publishers, San Diego.

Taner, M. T., O'Doherty, R. F., and Koehler, F., 1995, Long period multiple supression by predictive deconvolution in the x-t domain: Geophysical Prospecting, **43**, 433–468.

Tarantola, A., 1987, Inverse Problem Theory – Methods for Data Fitting and Model Parameter Estimation: Elsevier, New York.

Treitel, S., 1974, The complex Wiener filter: Geophysics, **39**, 169–173.

Webster, G. M., 1978, Deconvolution: Society of Exploration Geophysicists, Tulsa.

Yilmaz, O., 1987, Seismic Data Processing: Society of Exploration Geophysicists, Tulsa.