Durham E-Theses

Minimum entropy deconvolution: a novel processing technique for refraction seismology

Anderson, Garvey Michael

MINIMUM ENTROPY DECONVOLUTION

- A NOVEL PROCESSING TECHNIQUE

FOR REFRACTION SEISMOLOGY

by

GARVEY MICHAEL ANDERSON

A thesis submitted as part of the requirements

for the degree of Master of Science

in the University of Durham.

Graduate Society   September 1983

13. AUG. 1984

ABSTRACT

Most deconvolution techniques developed for reflection seismology suffer from the need to make assumptions about the disturbing function and the reflection series. In refraction seismology such assumptions are generally not valid. The Minimum Entropy Deconvolution (MED) technique of Wiggins (1978) requires no a priori knowledge of the phase characteristics of the disturbing function, nor does it assume the impulse response of the Earth's transmission path to be a white noise series. As such, it may be applied to short windows of refraction data containing only a few arrivals. The process seeks to simplify the representation of the input data, yielding an output of a small number of spikes. In this way the picking of arrivals on a refraction record is made much easier. By applying the technique to each trace independently, true arrivals may be distinguished from spurious spikes by correlation from one trace to the next.

CONTENTS

Chapter 5 APPLICATION OF MED TO REAL DATA

Chapter 6 DISCUSSION AND CONCLUSIONS

Acknowledgements

References

Appendix - Computer Programs

CHAPTER 1

INTRODUCTION

1.1 Aim

Minimum entropy deconvolution as developed by R.A.Wiggins (Wiggins,1978; Ooe & Ulrych,1979) has a number of properties which highlight it as being a potentially useful processing technique for application in the field of refraction seismology. However there seems to be no evidence in the literature of such an application being tested. The aim of this project was to apply the method to refraction data from the Caledonian Suture Seismic Project, and to develop any necessary modifications in order to produce a practical processing technique capable of yielding useful results.

1.2 Processing Techniques in Refraction Seismology

In general the processing of refraction records is at present restricted to the application of a bandpass filter and the production of a reduced travel-time plot. The latter process is applied simply to enable the data to be presented in a reasonably sized plot.

Typically the seismometer in a refraction survey has a lower frequency cut-off at about 1 Hertz, whilst the geostore sets the upper frequency cut-off at about 30 Hertz. The predominant frequency of the signal of interest depends on the shot-detector distance and the time of the

arrival on the record ; the longer the path travelled the greater the attenuation of high frequencies. The range 2 - 10 Hertz should generally cover all signals of interest. Consequently , applying the appropriate bandpass filter can do much to improve the signal-to-noise ratio of the record.

In general, only the onset times of first arrivals are employed directly to derive the velocity in the refractor and the depth to the top of the refractor , using, for example, the plus-minus method (Hagedoorn, 1959). Interpretation of later arrivals is normally done using synthetic seismograms. The aim of any novel processing technique must be to enhance all arrivals on a refraction record - not just refractions but also reflections , diving waves and channel waves.

Warren (1981) discusses why processing techniques developed for use in reflection seismology are not generally applicable to data produced from a refraction survey. This is primarily a result of the fact that the assumptions inherent in the use of such processes are not valid for refraction data.

A commonly accepted model for a seismic reflection trace is one where a time series consisting only of spikes separated by zeroes (the desired signal) is smoothed by convolution with some disturbing function or 'source wavelet'. By definition the source wavelet is considered constant along the trace. The Wiener filter

2

process based on this model further requires the source wavelet to be minimum delay , and assumes the spike series to be a white noise series (Robinson & Treitel, 1967).

For a refraction record, this model becomes invalid. Since a refraction survey generally involves much greater distances (hundreds of kilometres) compared to a reflection survey ($\sim$10 km), the disturbing function does not remain constant along the record; each arrival has generally followed a very different path, so that different arrivals have suffered different degrees of dispersion and attenuation of high frequencies. Arrivals from wide angle reflections additionally suffer phase distortion. This problem can effectively be solved by deriving filters independently for short windows of data, within which the wavelet character is essentially fixed. If in addition each window contains only one arrival, the assumption of a white noise output impulse series becomes unnecessary and the Wiener process may be applied. However, the separation of arrivals in this way is possible only when the arrivals are distinct and may be easily picked from the unprocessed data. It is clearly impossible for the case of two or three interfering arrivals, and in such a case the desired output for the window is not a white noise series. As a result, the use of the Wiener technique is not always a realistic proposition.

Warren (1981) showed how a matched filter may be

3

used with some success to follow a certain arrival from one trace to the next. The technique involves cross-correlating the known wavelet shape with each trace, and as such requires a good initial estimate of the wavelet from one of the unprocessed traces. This is generally possible for first arrivals provided there is available a trace in which the first and second arrivals do not interfere. However, it becomes increasingly difficult, if not impossible, for later arrivals. In addition the success of the technique depends on the assumption that the arrival shows little change in character from one trace to the next. This may be reasonable in the case of good data obtained using a repeatable source such as an airgun, although even under such favourable circumstances the technique is considered by some to be of dubious value (Summers, 1982). The assumption is unlikely to hold for common station data using a non-repeatable source such as dynamite.

In contra-distinction the minimum entropy deconvolution (MED) technique (Wiggins, 1978) requires no a priori knowledge of the disturbing function. Nor is it based on the assumption that the desired output (the impulse response of the transmission path) is a white noise series. However, its application is limited to data within which the disturbing function does not change. Therefore, as in Wiener filtering, windows must be applied to a refraction trace. Unlike Wiener, the window may

4

contain more than one arrival since the desired output need not be a white noise series ; consequently MED should be capable of resolving interfering arrivals. It is this point, together with the fact that no knowledge of the disturbing function is necessary, which is hoped will prove minimum entropy deconvolution to be of particular value in the field of refraction seismology.

1.3 The Caledonian Suture Seismic Project

The aim of the Caledonian Suture Seismic Project (CSSP) is to examine the crustal structure along strike just south of the inferred position of the Caledonian suture crossing north Britain (Bott & Long, 1981). The line of the survey is indicated in Figure 1.1. It is hoped that since the line runs along a fairly uniform geology the results will yield much better velocity information of the deep crustal structure than earlier projects. Closely spaced stations and closely spaced shots have resulted in good common depth point wide angle reflection information, which should produce much new information on the crust beneath this line.

C. S. S. P.

Shot and Station Locations

o Explosive Shots
· Seismic Stations

Fig. 1.1

KM

0 20 40 60 80 100
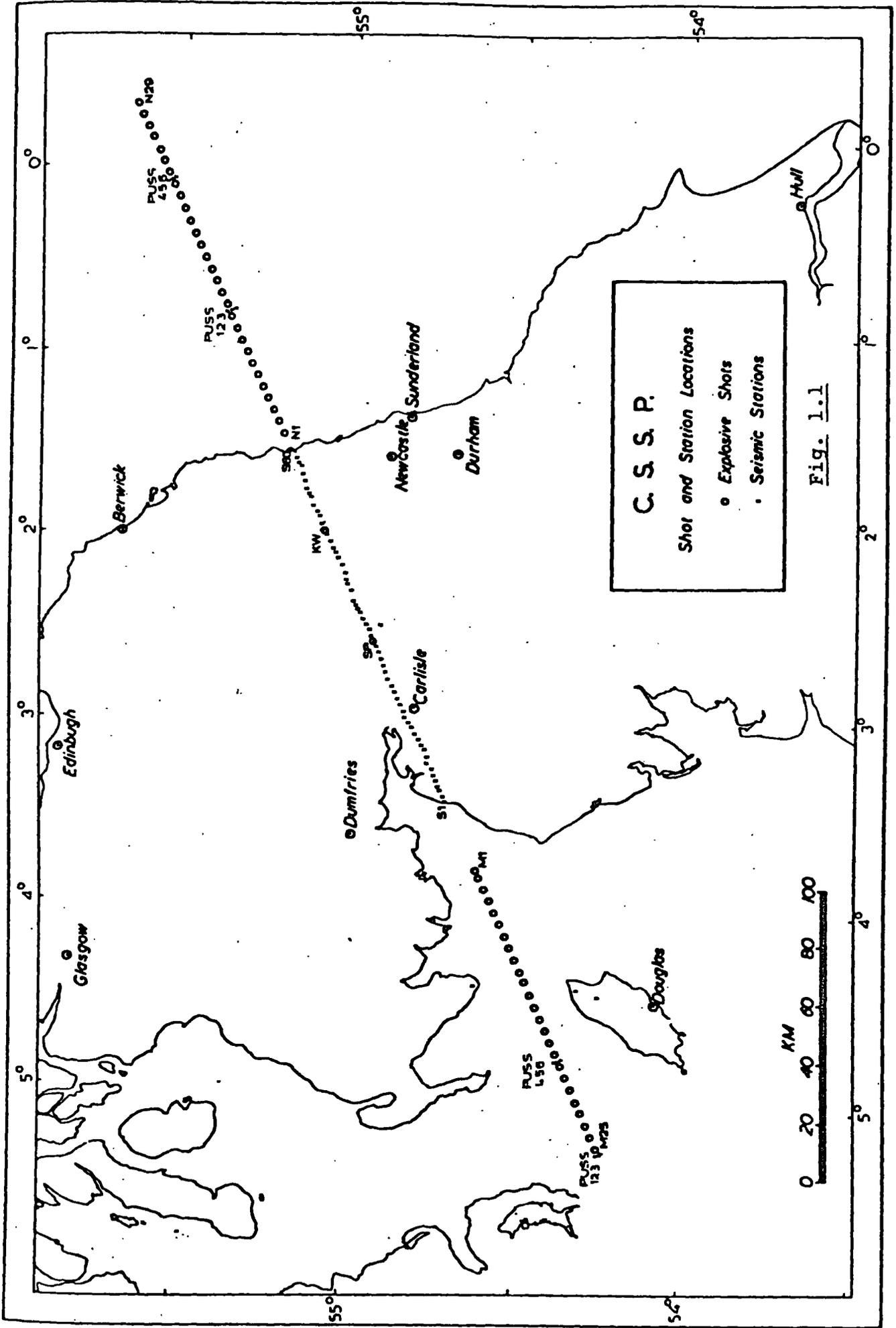
CHAPTER 2

MINIMUM ENTROPY DECONVOLUTION

2.1 Introduction

By convention a (noise-free) seismic trace is modelled as:

$$x(t) = s(t) * r(t) \qquad \text{(Al-Sadi, 1980)}$$

where $x(t)$ is the time series representing the seismic trace, $s(t)$ is the 'source wavelet' or disturbing function, and $r(t)$ is the impulse response of the Earth's transmission path.

A common feature of all deconvolution techniques is that they exploit some difference between the disturbing function or 'source wavelet' $s(t)$ and the impulse series $r(t)$, in order to separate the two components. The process of homomorphic deconvolution (Stoffa, Buhl & Bryan, 1974) depends on the fact that the power spectrum of the source wavelet is generally smooth compared to that of the impulse series. Predictive deconvolution (Peacock & Treitel, 1969) performs the separation on the basis that the source wavelet is minimum delay and the impulse series is white.

Minimum entropy deconvolution (Wiggins, 1978) is no exception. It is based on the idea that the simple structure of the impulse series is complicated by the disturbing function $s(t)$. Just as the process of

6

homomorphic deconvolution draws on the mathematical concept of the auto-correlation function as a measure of smoothness of the power spectrum, so minimum entropy deconvolution draws on the statistical concept of the Varimax Norm as a measure of simple structure.

2.2 The Varimax Norm

The Varimax Norm is a well-established means of measuring the simplicity of representation of some data set $\{a_i, i = 1,.....n\}$ (Kaiser, 1958 ; Carroll, 1953). The fundamental statistical theory behind the derivation of the norm is well beyond the scope of this thesis. However, it is straightforward to follow the arguments through for a simple case, and to accept the logical extension of the result to more difficult situations.

Consider the case where the data set consists of only two values (a_1, a_2). This can be represented as a single point in a two-dimensional vector space defined by co-ordinate axes x and y :

$$\quad (a_1, a_2)$$

Fig. 2.1

7
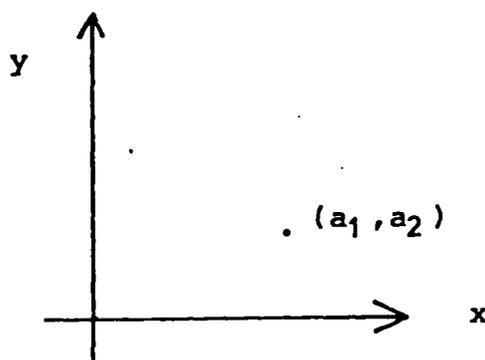
By means of an orthogonal transformation, the axes may be rotated so that one of them passes through the point. The representation of the point in terms of this new set of co-ordinate axes is intuitively the most simple representation possible (Ferguson, 1954). By this definition the data set has been simplified by applying an orthogonal transformation (a rotation, in fact) such that the product of the squares of the co-ordinates is minimised.

A property of such a transformation of the axes is that the sum of the squares of the co-ordinates of a fixed point remains constant under the transformation. If the data set is represented with respect to the new axes as (b_1, b_2) this means that

$$b_1^2 + b_2^2 = a_1^2 + a_2^2 = \text{constant}$$

Extending this to the case of an n-length data set $\{a_i, \ i = 1, \ldots \ldots n\}$ leads to :

$$\sum_{i=1}^{n} b_i^2 = \sum_{i=1}^{n} a_i^2 = \text{constant}$$

$$\Longrightarrow \quad (\sum_{i=1}^{n} b_i^2)^2 = \sum_{i=1}^{n} b_i^4 + 2 \sum_{i<j=1}^{n} b_i^2 b_j^2 = \text{constant}$$

The second sum in the addition is simply the extension of the criterion for simple structure developed above for the case n = 2. It can be seen that maximising the first sum is equivalent to minimising the second sum. In other words, the simplest representation of the data set is that for which the frame of reference has been

rotated so as to maximise the value of $\sum_{i=1}^{n} b_i^4$. This leads to the Varimax Norm, defined as :

$$V = \frac{\sum_{i=1}^{n} b_i^4}{(\sum_{i=1}^{n} b_i^2)^2} \qquad \ldots\ldots\text{(Eqn. 2.1)}$$

2.3 The Varimax Norm Applied to a Seismic Trace

A digitised seismic trace of n samples is just a data set of length n, where each sample may be regarded as specifying one co-ordinate in some n-dimensional vector space. The complete trace, then, is represented as a single point in this space. As a result the Varimax Norm may be legitimately applied to measure the simplicity of representation of the data set.

The transformation law for Cartesian co-ordinates in 3-dimensional space is given by Kreyszig (Chapter 8) as :

$$x^* = c_{11}x + c_{12}y + c_{13}z$$
$$y^* = c_{21}x + c_{22}y + c_{23}z \qquad \ldots\ldots\ldots\text{(Eqn. 2.2)}$$
$$z^* = c_{31}x + c_{32}y + c_{31}z$$

Considering the result of convolving some 3-length filter with the seismic trace leads to :

$$y_1 = f_1 x_1$$
$$y_2 = f_2 x_1 + f_1 x_2 \qquad \ldots\ldots\ldots\text{(Eqn. 2.3)}$$
$$y_3 = f_3 x_1 + f_2 x_2 + f_1 x_3$$

Comparison of (2.2) with (2.3) shows that the process of applying a filter to a seismic trace is exactly equivalent to applying some transformation. In terms of

9

Section 2.2, suitable choice of the filter coefficients will perform the rotation spoken of there. Consequently, the use of the Varimax Norm for our purposes is justified. It will be demonstrated in the following section that the norm lends itself to a tractable computational procedure whereby the appropriate filter can be derived.

Comments

Visualising the process of rotation in an n-dimensional vector space is a difficult exercise for n greater than three. However, it is useful to appreciate the ideas behind the Varimax Norm, since it forms the very basis of the MED technique. Indeed, some useful information may be derived from Sections 2.2 and 2.3 without too much mathematics.

Given the fact that complete freedom of rotation in an n-dimensional space requires n parameters in the transformation (Arfken, Chapter 1), Equations 2.2 and 2.3 demonstrate that a filter of length n is required to reduce an n-length data set to its most simple form ; n is, in fact, the maximum length required since complete freedom of rotation will not always be necessary, depending on the simplicity of the original data set. Wiggins (1978) states that the MED process seeks the smallest number of spikes that is consistent with the data. From our investigation of the Varimax Norm his statement is shown to be somewhat ambiguous ; the smallest number of spikes consistent with any data set is always

10

one. To clarify the situation it is necessary to add that the smallest number of spikes to which the data may ultimately be reduced using his process depends on the length of filter used. In particular, too long a filter may oversimplify the data from the seismologist's point of view in that it may yield one spike from two (or more) arrivals.

The relationship between filter length and resolving power is of considerable importance. Wiggins (1978) does not discuss filter length, whilst in their single trace examples Ooe & Ulrych (1979) choose a filter length of twice the source wavelet length with the qualification that "much experience needs to be gained" with respect to this choice. Certainly the discussion above indicates that if the input data consists of two interfering arrivals, and the filter length is equal to the length between the onset of the first arrival and the tail of the second arrival, then that filter will ultimately reduce the input to a single spike. This effectively sets an upper limit to the length to use. Indeed a filter length even shorter than this may lead to such over-simplification, but how much shorter depends on the input data itself and cannot be accurately predicted. A lower limit on the filter length is set by the ability of the filter to reduce a single arrival to a sharp spike. This lower limit will be equal to, or more likely less than, the length of the arrival wavelet.

11

2.4 Derivation of the Normal Equations

For the purposes of application to a digitised seismic trace the Varimax Norm is defined as :

$$V = \frac{\sum\limits_{i=1}^{m+n-1} y_i^4}{\left(\sum\limits_{i=1}^{m+n-1} y_i^2\right)^2} \qquad \ldots\ldots\ldots(\text{Eqn. } 2.4)$$

where $\quad y_i = \sum\limits_{s=1}^{m} f_s\, x_{i-s}$, $\quad i = 1,\ldots\ldots,m+n-1$

with f_s , $s = 1,\ldots,m$ as the filter coefficients,

and x_i , $i = 1,\ldots,n$ as the raw trace.

Maximising the norm with respect to the filter coefficients :

$$\frac{\partial V}{\partial f_l} = 0$$

Simple differentiation leads to :

$$4Vu^{-1}\sum\limits_{i=1}^{m+n-1} y_i \cdot \frac{\partial y_i}{\partial f_l} - 4u^{-2}\sum\limits_{i=1}^{m+n-1} y_i^3 \frac{\partial y_i}{\partial f_l} = 0 \qquad \ldots\ldots.(\text{Eqn. } 2.5)$$

where $u = \sum\limits_{i=1}^{m+n-1} y_i^2$

Now $\dfrac{\partial y_i}{\partial f_l} = \dfrac{\partial \left(\sum\limits_{s=1}^{m} f_s\, x_{i-s}\right)}{\partial f_l}$

$$= x_{i-l}$$

so that Equation 2.5 reduces to :

$$Vu \sum\limits_{s=1}^{m} f_s \sum\limits_{i=1}^{m+n-1} x_{i-s}\, x_{i-l} = \sum\limits_{i=1}^{m+n-1} y_i^3\, x_{i-l}, \qquad l = 1,\ldots\ldots,m$$

which can be written in matrix form as :

$$\underline{R}\,\underline{f} = \underline{g} \qquad \ldots\ldots.(\text{Eqn.2.6})$$

12

where $\underline{R}$ is a Toeplitz autocorrelation matrix consisting of autocorrelations of the input data weighted by Vu, $\underline{g}$ is a column vector of cross-correlations of the outputs cubed with the inputs and $\underline{f}$ is the column vector of the filter coefficients. It is immediately obvious that this system of normal equations is highly non-linear. As a result they must be solved iteratively starting with an initial filter $(0,\ldots,1,\ldots,0)$ so that y_i, $i = 1,\ldots,m+n-1$ may be deduced. These values of y_i are substituted into the normal equations which are then solved to yield new filter coefficients. The iterative process is continued by applying the new filter to the data and solving the normal equations again, using the new values of y_i produced. Normally iteration would continue until the result represents a good approximation to the true solution of the set of equations. It will be demonstrated in Chapter 4 that this is not always desirable in the case of minimum entropy deconvolution.

Examination of the normal equations reveals that the right-hand side for the first iteration is just the cross-correlation of the input with the input cubed ; the normal equations attempt to find a filter which shapes the inputs to the spikey appearance of the cubed traces. In this respect the MED technique may be compared to a Wiener spiking filter (Section 2.7).

Wiggins (1978) warns that the Toeplitz matrix $\underline{R}$ may be nearly singular in some applications which means

13

that the normal equations may be ill-conditioned. This problem was, in fact, encountered (Chapter 5) and was solved by increasing the diagonal terms of the matrix by 0.5%.

2.5 Properties of MED

At this stage several important properties of the technique can be deduced :

1) From the derivation presented it is clear that the filter coefficients are derived from the data itself and no a priori estimate of the disturbing function is required.

2) There is no requirement to make any phase assumptions about the disturbing function, nor must the impulse series be assumed to be a white noise series.

3) Since the design criterion for finding an MED operator refers only to the simplicity of the output, the polarity or delay of the output spikes cannot be accurately predicted.

4) The process leaves white noise unaffected. This can be most clearly seen by considering an extreme example where the input data x_i, $i = 1,....,n$ is white noise with no signal present. A property of white noise is that its autocorrelation function is non-zero only at lag zero. Consequently, the autocorrelation matrix $\underline{R}$ has non-zero values only along its diagonal. Now assume without loss of generality that the initial filter is $(1,0,0,...,0)$, with

14

the result that for $i = 1, \ldots \ldots, n$ $y_i = x_i$; $y_i = 0$ for $i = n+1, \ldots \ldots, m+n-1$. Recall the normal equations (Equation 2.6) :

$$\underline{R} \ \underline{f} = \underline{g}$$

In the first step of the iterative process $\underline{g}$ is simply the cross-correlation of x_i^3 with x_i, $i = 1, \ldots \ldots, n$. Hence only the zero-lag component of $\underline{g}$ is non-zero. The normal equations, then, reduce to simply :

$$\emptyset_{x\,x}(0).f_1 = \emptyset_{x^3x}(0)$$

where $\emptyset(\tau)$ represents the process of cross-correlation at lag τ. f_1 is scaled by some factor whilst all other coefficients remain equal to zero. Therefore, when the 'new' filter is applied to the data it has only a scaling effect on each term. In other words the data is unaffected by the process.

5) The source wavelet shape must remain constant along the seismogram. This is most easily understood by considering how the MED filter is applied rather than how it is developed. Assuming the filter length to be less than the data length, as it is passed along the data it is clear that if it does a good job of spiking up one particular wavelet it cannot possibly do an equally good job of spiking up a completely different one.

2.6 Application of MED to Refraction Data

The. MED process may only be applied to short windows of a refraction trace, the criterion for the

15

length of window being that all arrivals within it be of the same character. Since the output impulse series need not be a white noise series there is no need to make any assumptions about the number of arrivals within the window. In any case the above criterion means that the number is likely to be restricted to at most two or three.

If the trace is modulated by a low frequency 'roll', as illustrated in Figure 2.2 below, then those values at the crests and troughs of the roll will be given proportionately more weight as a result of the cubing process.
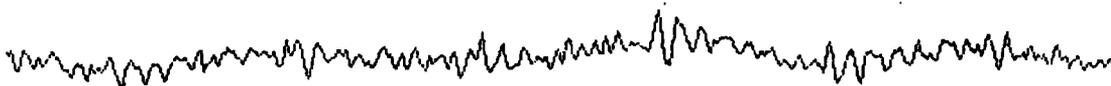
Fig. 2.2

If such modulation does occur it would be desirable to pass the data through a high-pass filter before applying the MED process. The application of a bandpass filter to also remove high frequency noise would not be expedient, at least in the case where the signal-to-noise ratio is already good. In such circumstances, since the process of making the trace spikey necessarily introduces high frequencies into the trace, there seems little point in removing them beforehand. In fact, removal of the high frequency content of the trace reduces the

efficiency of the MED process in that the spikes produced are less sharp (Section 5.7). Of course, in cases where the amplitude of high frequency noise exceeds that of the arrival itself then clearly the application of a bandpass filter is essential, since the MED process enhances large amplitudes. The CSSP data chosen for applying MED was of excellent quality, and required no such filtering.

2.7 A Comparison of MED and the Wiener Spiking Filter

Neither Wiggins (1978) nor Ooe & Ulrych (1979) consider what effect the position of the '1' in the initial filter may have on the output. The problem is best discussed by considering the input data to contain only one arrival, so that the output impulse series is just one spike. Then the normal equations for the Wiener spiking filter may be expressed as :

$$\sum_{s=1}^{m} f_s \sum_{i=1}^{m+n-1} x_i \, x_{i-s} = \sum_{i=1}^{m+n-1} d_i \, x_i$$

$$\sum_{s=1}^{n} f_s \sum_{i=1}^{m+n-1} x_{i-1} x_{i-s} = \sum_{i=1}^{m+n-1} d_i \, x_{i-1}$$

$$\vdots \qquad\qquad \vdots$$

$$\vdots \qquad\qquad \vdots$$

$$\sum_{s=1}^{m} f_s \sum_{i=1}^{m+n-1} x_{i-m} x_{i-s} = \sum_{i=1}^{m+n-1} d_i \, x_{i-m}$$

......(Eqn.2.7)

where d_i , i = 1,.....m+n-1 here is the desired output, which is a single spike (Robinson & Treitel, 1967). The location of the spike in the desired output should be chosen according to the phase characteristic of the arrival ; for a minmimum delay wavelet the desired

17

output should be (1,0,0,...,0), for maximum delay (0,0,...,0,1) and for mixed delay the spike should be chosen in some intermediate location. These choices lead to filter coefficients which, when applied to the data, will result in the minimum error between the actual and desired outputs (Claerbout & Robinson, 1963). The ideas behind Wiener filtering are well-established.

The normal equations for MED are :

$$Vu \sum_{s=1}^{m} f_s \sum_{i=1}^{m+n-1} x_i \; x_{i-s} = \sum_{i=1}^{m+n-1} y_i^3 x_i$$

$$Vu \sum_{s=1}^{m} f_s \sum_{i=1}^{m+n-1} x_{i-1} x_{i-s} = \sum_{i=1}^{m+n-1} y_i^3 x_{i-1}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \ldots\ldots\ldots\ldots(Eqn.2.8)$$

$$Vu \sum_{s=1}^{m} f_s \sum_{i=1}^{m+n-1} x_{i-m} x_{i-s} = \sum_{i=1}^{m+n-1} y_i^3 x_{i-m}$$

as derived earlier.

Comparison of Equations (2.7) and (2.8) shows that, apart from the scaling factor Vu, Equation (2.8) is just (2.7) with d_i replaced by y_i^3. In other words the process for obtaining the MED filter is the same as that for obtaining the Wiener spiking filter, except that in MED the desired output is derived from the data, whilst in Wiener filtering it is chosen by the user.

Consider the example where the input data is the minimum delay wavelet (2,1). The optimum 2-point Wiener spiking filter will be produced for a desired output of (1,0,0). For MED an initial filter of (1,0) leads to y = (2,1,0), so that the 'desired output' becomes (8,1,0), which may be expressed as (1,0.125,0). This is close to

18

the output that one would choose to yield the optimum spiking filter. On the other hand, an initial filter of (0,1) leads to y = (0,2,1) and a desired output of (0,8,1). The filter produced with this initial choice of filter cannot be expected to do such a good job, just as one would not expect a desired output of (0,1,0) to lead to a good Wiener spiking filter (Claerbout & Robinson, 1963). In the same way it can be shown that an initial filter (0,1) is the best choice for the maximum delay wavelet (1,2).

Unfortunately, not all minimum delay wavelets have their maximum value at $i = 1$. The wavelet (4,6,4,1) is one example. Even using an initial filter (1,0,0,0) leads to $y^3 = (64,216,64,1,0,0,0)$. As the iterative process continues the second data value will be progressively amplified with respect to the other values so that the 'desired output' at each step becomes more and more like (0,1,0,0,0,0,0). This is not the desired output (1,0,..,0) one would choose from Wiener theory. However, it is the closest one can get using a purely causal initial filter. Consequently, an initial filter (1,0,0,...,0) is the best possible choice for all minimum delay wavelets.

Summary

It has been shown that in MED the 'desired output' can be delayed with respect to the input by choosing the position of the '1' in the initial filter. Further, as a result of the cubing process the desired

19

output always converges to a single spike. For a minimum delay wavelet this spike should be as close to $i = 1$ as possible, so the initial filter should be chosen with the '1' at $i = 1$ in this case. In general the MED process will not yield a spike at the onset of the wavelet since the position of the spike is defined by the position of the maximum value within the wavelet.

Problems A - Spike location

The fact that the position of the spike chosen by the process is beyond our control, and that the spike is unlikely to be located at the onset of the arrival, is a distinct disadvantage of the MED technique. There would be no problem if the arrival were of the same character all along the trace, as in reflection records. However, in a refraction trace the changing character of the wavelet will inevitably result in the position of the spike for each arrival being delayed from the onset time by different amounts, so that the spacing of spikes on the processed trace will not correspond to the true spacing of arrivals. In addition, when following an arrival from one trace to the next the moveout between spikes will only be equivalent to the moveout between onsets if the wavelet character remains essentially the same between traces. This is unlikely for common-station data using a dynamite source. Since the spacing of arrivals along one trace and the relative moveout between traces are precisely the two parameters used to estimate the velocity

20

and depth of a given refractor, it appears that the MED process will be of little help in quantitative interpretation of data.

Problems B - Data containing more than one arrival

The ideas presented in this section are strictly valid only when the MED process is applied to windows of data which contain just one arrival, since a desired output of more than one spike is incompatible with the Wiener formulation. Since the MED process derives a filter from the complete input data then for a window containing more than one arrival the phase of the impulse series must also be taken into account when considering the phase of the data. The impulse series is generally mixed delay, so that the data is mixed delay, regardless of the phase of the individual arrivals.

At this stage it is difficult to stop the hands from waving. Should the initial filter be chosen according to the phase of the complete data or according to the phase of the disturbing function? Intuitively one would imagine that, since the requirement of the filter is to spike up each arrival, the phase of the disturbing function should be used. Wiggins (1978) and Ooe & Ulrych (1979) are of no help here as their examples use a mixed delay wavelet convolved with a mixed delay impulse series, so that both the complete data and the wavelet itself are mixed delay. Their choice of initial filter with the '1' in the middle location is specified without explanation,

21

and would be chosen on the basis of our discussion in any case. The problem can only be resolved by considering specific examples, which is done in Section 4.2. There it is shown that the choice of initial filter is of considerable importance and may even lead to that ultimate disaster, the loss of arrivals.

CHAPTER 3

PROGRAM DEVELOPMENT

3.1 Introduction

All programs used throughout this project were written in Fortran for use on the IBM 4341 mainframe computer. As such they should be readily transferrable to similar machines with the exception of the plotting programs which call plotting subroutines from the library *GHOST. The procedure for reading data from tapes is likely to differ from one system to another, and depends also on how the tapes are written. Consequently, this procedure must be regarded as applying specifically to the system used at Durham.

3.2 Data Preparation

Data from the CSSP is written onto magnetic tape in binary code using a PDP-11. A sampling rate of 100 samples per second is used, each data value being specified as a 2-byte integer (INTEGER*2), written with the most significant byte last. The block length on tape is 2048 bytes so that each block contains 1024 integer values. 13 blocks make up one file, which contains the data for one trace with the first block of each file containing only header information.

Recording was started at the shot time rounded

down to the nearest second. Given the shot time and shot-detector distance it was possible to work out which data sample for a given file was required as the starting sample for a reduced travel-time plot, using a reduction velocity of 6.0 km s^{-1}. Each file was then copied from tape onto temporary disk space and the one or two data blocks containing the required data were used as the input for program TAPEREAD. Only eight seconds of data per trace were ever used, and TAPEREAD was developed to prompt for the desired starting sample and to read (in binary code) the first 800 samples from this point. TAPEREAD also performs a byte-swopping operation, necessary because the IBM 4341 reads the first byte of each integer as the most significant. Output from TAPEREAD is one line of byte-swopped binary code. The header information, if required, may be decoded using the system subroutine DURH:ATOEB. The output from TAPEREAD may be converted for checking purposes into readable (EBCDIC) integer values using program READ.

Program AV was developed to remove a d.c. level from the data. It calculates the average of the 800 data values in the trace and subtracts this average from each value. Input is just the output from TAPEREAD, i.e. one line of binary code.

In order to apply a bandpass filter to the data, program BANBOX was developed together with program BANKONV. Warren (1981) produced program BANBOX to

calculate the coefficients of a bandpass filter in the time domain. He took the ideal frequency bandpass filter (Fig. 3.1) and by expanding A(f) as a Fourier cosine series was led to a filter with infinitely long causal and non-causal components.
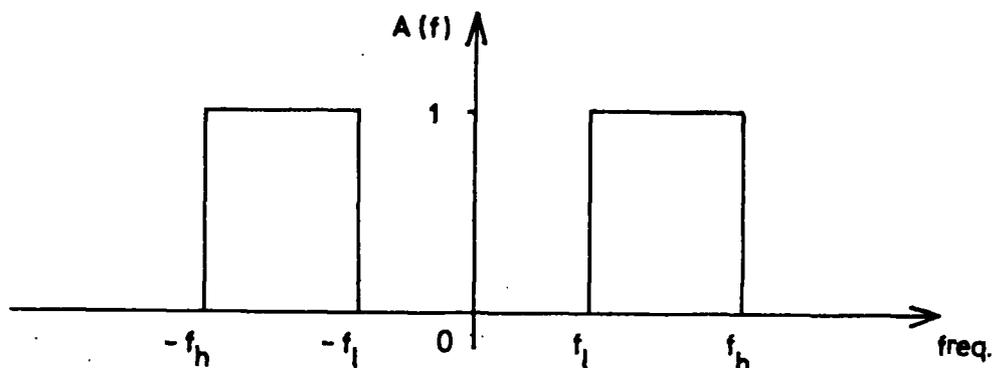
Fig. 3.1

In yielding a finite realisation of the filter, the truncation error was reduced by application of a Hanning window. This seems rather extreme, since a Hanning window leaves only the central value unaffected whilst reducing all other values of the derived filter. Feeling that the application of a box-car window with tapered ends would lead to much less corruption of the filter coefficients whilst still avoiding the problems associated with Gibbs' Phenomenon, Warren's program was modified accordingly. The result is a program which prompts for the length of filter required and outputs the filter in binary code. Program BANKONV may then be used to apply the

filter to the data.

Program EQUALIZE was developed purely for use on Kirkwhelpington data. This is common-shot data from a land-based shot and there is considerable variation in signal amplitude between the stations. In order that a useful plot may be produced, program EQUALIZE scans each trace for the largest data value and sets this equal to some pre-set maximum value which applies to all the traces. The rest of the values on the trace are then scaled up or down by the appropriate scaling factor which, for a given trace, will be equal to the pre-set maximum value divided by the maximum value in the trace. The input to the program is in binary code, as is the output. In practice the output from program AV was used as input for EQUALIZE.

PROGRAM MED FLOW DIAGRAM

SUBROUTINES

AUTCOR
MATRIX

CONVO

VU

CRSCOR

MWIENR

INPUT

DATA x_i $i=1,........,n$

INITIAL FILTER

f_s $s=1,........,m$

CALCULATE AUTOCORRELATION MATRIX $\underline{R}$

CALCULATE y_i $i=1,...,m+n-1$

CALCULATE V AND U

CALCULATE g

SOLVE $\underline{R}\,\underline{f} = \underline{g}$ FOR $\underline{f}$

RECALCULATE y_i

ITERATE?

YES

NO

STOP
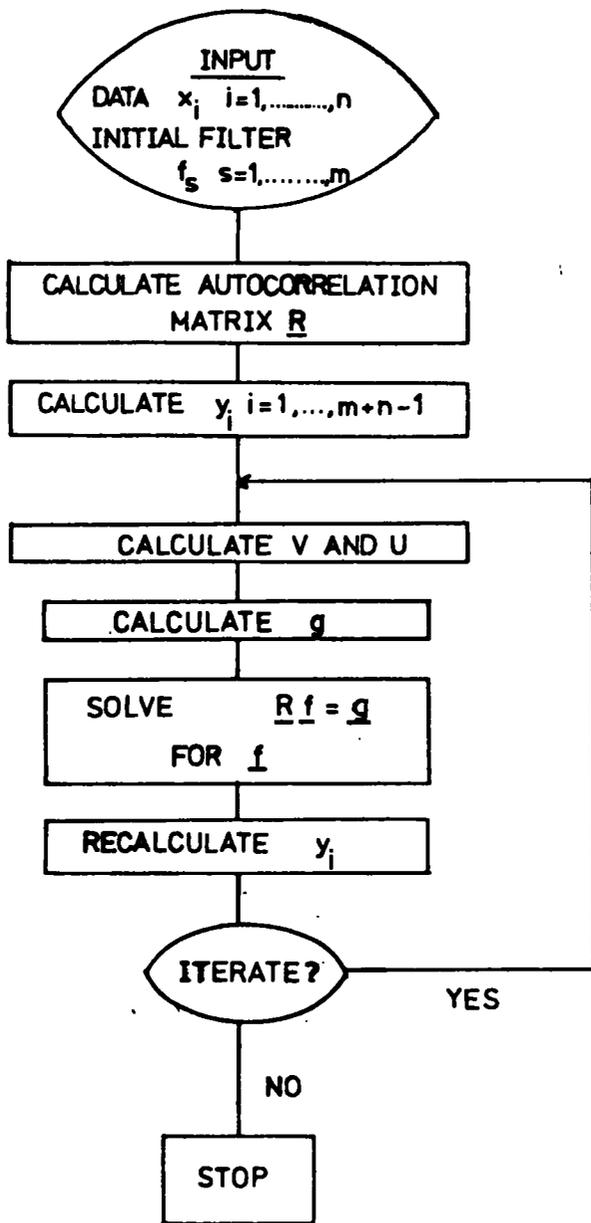
Fig. 3.2

3.3 Minimum Entropy Deconvolution

The development of a program for performing minimum entropy deconvolution (MED) was based on the flow diagram in Fig. 3.2.

The subroutines used for each stage of the process are listed down the left-hand side of the chart. The main program and the subroutines are all original with the exception of subroutine MWIENR, which performs matrix inversion using the Levinson recursion (Claerbout, 1976). This routine was originally designed for use on an array processor to solve the normal equations for a Wiener filter. From Section 2.7 , the solving of the normal equations for MED is basically the same problem, so that after suitable modifications to enable it to run on the IBM mainframe, the routine was transferred and altered where necessary. Subroutine MATRIX was developed purely to arrange the autocorrelation coefficients into the correct sequence for input to MWIENR. The other subroutines perform fairly standard operations and are described in the appendix.

Double precision is used throughout program MED and the input data must be in binary code. The input file may contain data from any number of traces since the program prompts for the number of traces on which to operate. Traces are read in the order in which they appear in the input file, the data for each trace being contained in one line, preceded by six lines of

27

additional information as specified in the comments at the beginning of the program. The program listed in the appendix accepts a maximum number of 800 samples per trace and allows a maximum filter length of 50, but these limits may be easily extended by changing the dimensions of the appropriate arrays. However, moderation should be exercised with respect to the filter length, since the size of arrays AUT, AUTVU (subroutine MWIENR) and AUTMX (subroutine MATRIX) is equal to the square of the filter length. Since double precision is specified the program requires a large amount of memory space, and its position in the execution queue depends on how large the required space becomes.

In practice the output from AV or BANKONV was used as input to MED. However, AV operates only on one trace at a time, so that the output from each trace was copied into one large file and the appropriate titles added using the file editor.

Execution times depended on :

    a) the number of traces

    b) the filter length

    c) the number of iterations performed

Table (3.3) shows the actual times taken for specific cases. It can be seen that the increase in time with respect to filter length is very nearly linear ; indeed for 29 traces and five iterations the CPU time in seconds may be estimated by multiplying the filter length by 12.

28

From the limited amount of information available, the relationship between the number of iterations performed and the time taken is not linear ; the fewer the iterations, the greater the time per iteration.

Filter length	No. of iterations	No. of traces	CPU time (secs)
20	5	29	225
30	5	29	350
40	5	29	500
50	5	29	660
50	2	29	320
50	5	31	720
50	10	29	1235

Table 3.3

Run Times for Program MED

3.4 Plotting Routines

To produce results in a useful form, two plotting programs were developed : SYNPLOTTER, for illustrating the results of each iteration on synthetic data (Chapter 4), and PLOTTER, for use on real data. Both programs call *GHOST subroutines.

Program SYNPLOTTER reads data in F format, with each data value on a new line within the input file. It

takes as input the output from program SYNMED, a version of MED used specifically for application to synthetic data. The output from SYNMED contains all the titles required for the plot, examples of which are shown in Chapter 4.

Program PLOTTER reads data in double precision binary code, which is converted to single precision F format to be compatible with *GHOST subroutines. All information required to produce an annotated reduced travel time plot is contained within the output from program MED.

3.5 Summary

Fig. 3.4 summarises the procedure involved in going from the data on tape to the final plot of the deconvolved traces. Points at which human intervention is necessary are indicated, and these generally highlight those areas where the package may be improved. Fig. 3.5 illustrates the different procedure involved in applying the process to synthetic data.

```
                    ┌─────────────────┐
                    │   TAPE DRIVE    │
                    └─────────────────┘
                ╱─────────────────────────╲
               ╱   Copy from tape to        ╲
               ╲   temporary disk space     ╱
                ╲─────────────────────────╱
                      ┌──────────────┐
                      │   TAPEREAD   │
                      └──────────────┘              Data
                         ┌────────┐               preparation-
        YES              │   AV   │               requires work
                         └────────┘                at terminal
                ╱─────────────────────╲
               ╱   Copy into permanent  ╲
               ╲        file            ╱
                ╲─────────────────────╱
           ◄────────  More traces?

                    NO

                Edit file ; add titles

                                   YES            ┌──────────────┐
                   Plot?──────────────────────────►│   PLOTTER    │
                                                   └──────────────┘
                    NO
                                                   ┌──────────────┐
           ◄────────  Bandpass?                    │  *MTSPLOT    │
      NO            YES                             └──────────────┘

                  ┌──────────────┐                        │
                  │   BANKONV    │                        ▼
                  └──────────────┘               Hardcopy plot
                                                 of raw data
           ◄────────  Equalize?
      NO            YES

                  ┌──────────────┐
                  │  EQUALIZE    │
                  └──────────────┘
                     ┌────────┐
                     │  MED   │
                     └────────┘
                  ┌──────────────┐
                  │   PLOTTER    │
                  └──────────────┘
                  ┌──────────────┐
                  │  *MTSPLOT    │──────────────►Hardcopy plot
                  └──────────────┘                of processed
                                                     data
```
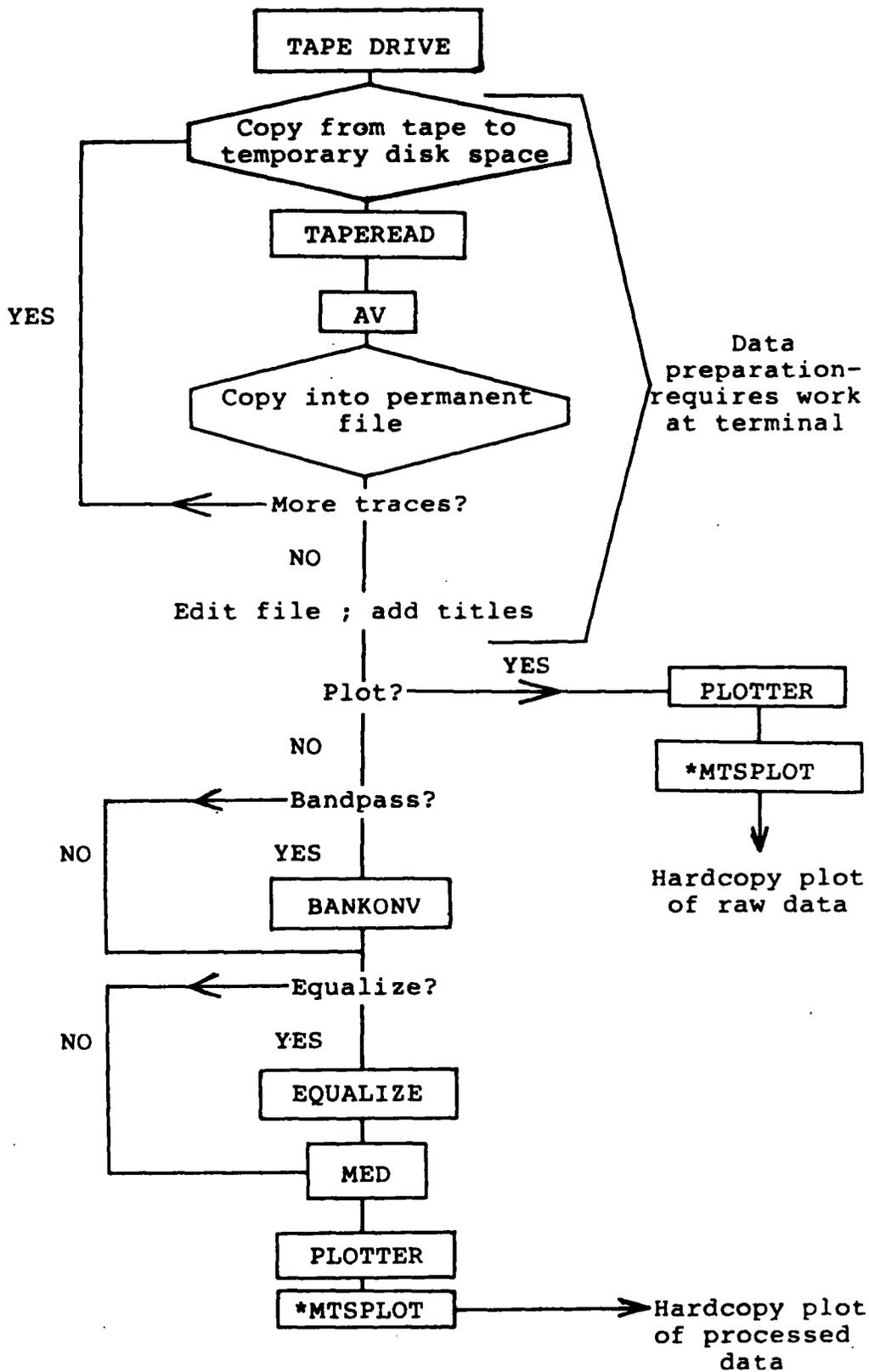
Processing Sequence for Real Data

Fig. 3.4

```
      ╱──────────────╲                          ╱──────────────╲
     ╱  Create wavelet ╲                        ╱  Create impulse ╲
     ╲   time series   ╱                        ╲   time series   ╱
      ╲──────────────╱                          ╲──────────────╱
              ╲                                  ╱
               ╲                                ╱
                ┌─────────────────┐
                │     SYNKONV     │
                └─────────────────┘
                         │
                ┌─────────────────┐
                │     SYNMED      │
                └─────────────────┘
                         │
                ┌─────────────────┐
                │   SYNPLOTTER    │
                └─────────────────┘
                         │
                ┌─────────────────┐
                │    *MTSPLOT     │
                └─────────────────┘
                         │
           ╱─────────────────────────────╲
          ╱   Hardcopy plot of raw data    ╲
          ╲   and successive iterations    ╱
           ╲─────────────────────────────╱
```
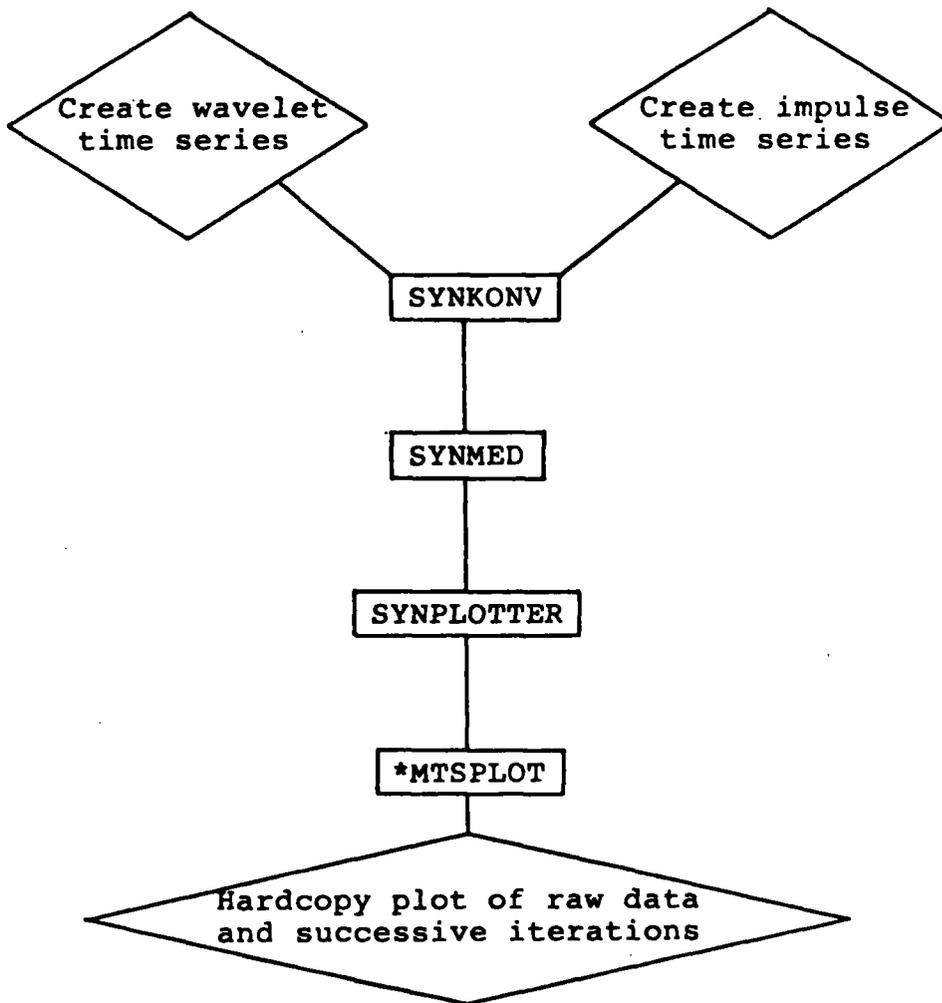
Processing Sequence for Synthetic Data

Fig 3.5

Comments

        It is clear that data preparation takes up a large amount of human time. Consequently, it would be desirable to develop TAPEREAD into a much more comprehensive program which would ideally :

1) Control the tape drive to find files specified by the user.

2) Read the header information in the first block and calculate which data samples would be required for a reduced travel time plot.

3) Copy these data samples onto disk together with appropriate titles derived from the header information.

4) Subtract a d.c. level if required.

5) Bandpass and/or equalize if required.

6) Produce an output file suitable for input to program MED.

        With such developments it would be possible to run the entire package as a batch job. This clearly would be a vast improvement, and would make the bulk processing of data a more realistic proposition.

CHAPTER 4

APPLICATION OF MED TO SYNTHETIC DATA

4.1 Introduction

Synthetic data was used in the early stages of program development in order to test the program and to improve our understanding of the MED process. Since the process was to be applied only to short windows of real refraction data (Section 2.6), the synthetic data was restricted in all cases to contain only one or two arrivals. Careful analysis of the results enables intelligent decisions to be made when applying the process to real data.

4.2 Phase Considerations
Single arrivals

To examine how the phase of the input data may influence the choice of position of the '1' in the initial filter, the data was initially restricted to contain only one arrival. The 18-point minimum delay wavelet of Figures 4.1 to 4.3 was developed by convolution of a number of 2-point minimum delay wavelets ; the maximum delay wavelet of Figures 4.4 to 4.6 is simply the minimum delay wavelet reversed. The choice of an 18-point filter was somewhat arbitrary other than the fact that it is the shortest filter which spans the complete wavelet.
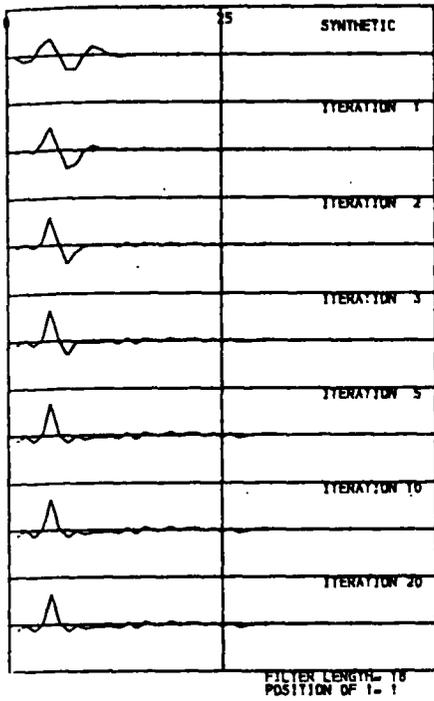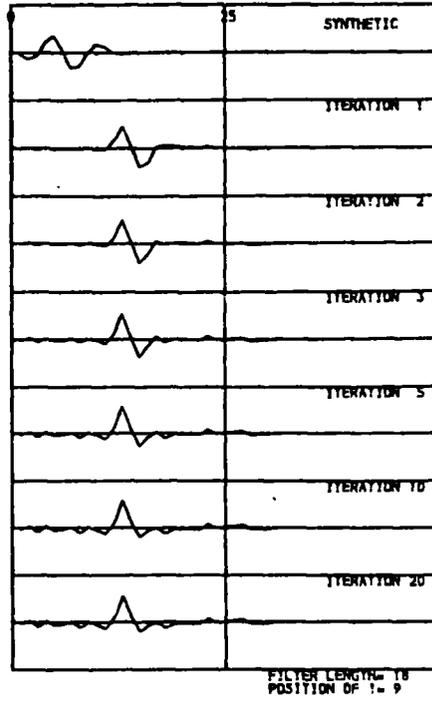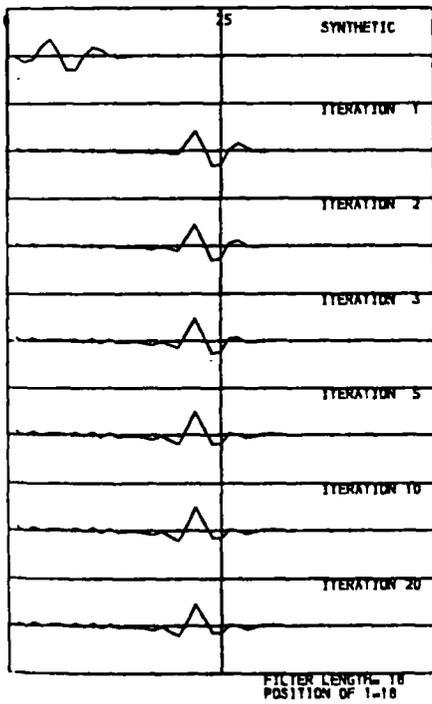
32

Fig. 4.1

Fig. 4.2

Fig 4.3

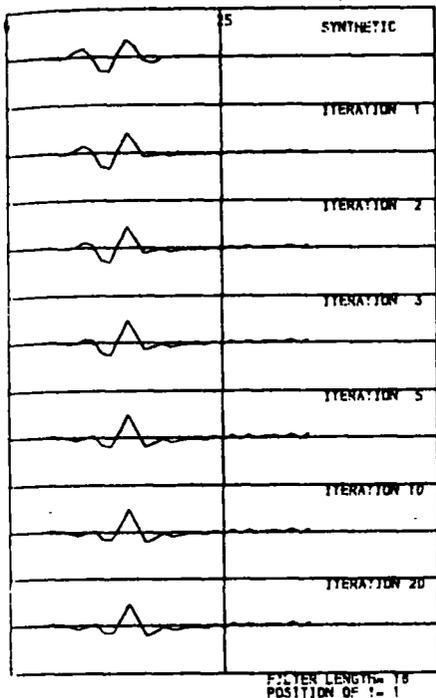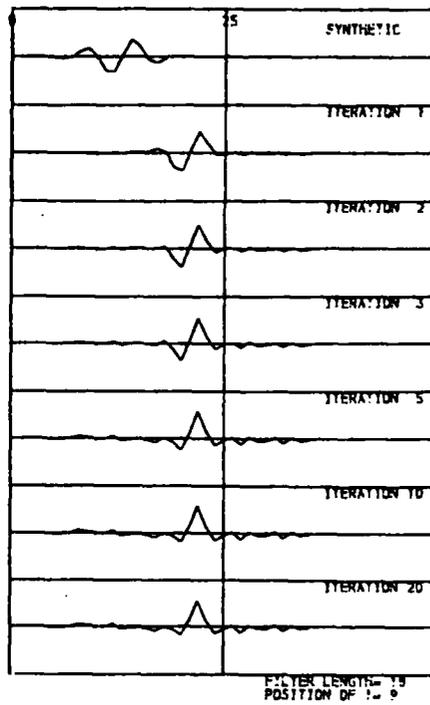Effect of position of 'l' on single minimum delay wavelet.
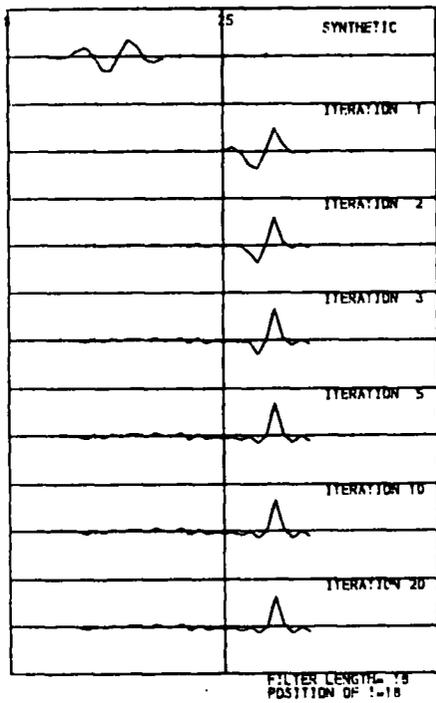
Fig. 4.4

Fig. 4.5

Fig. 4.6

Effect of position of 'l' on single maximum delay wavelet.

Discussion of results

The most obvious point to notice is that the MED process works - the result of its application to a single arrival is a single spike. Closer examination of the results confirms the validity of the ideas developed in Section 2.7. Figures 4.1 through to 4.3 show that the optimum output for a minimum delay input is achieved with the '1' in the leading position of the initial filter. As the '1' is moved from the leading position, so the output 'spike' becomes broader and of lower amplitude, whilst the amplitudes of those samples to either side of the spike are increased ; in other words the resolution becomes poorer. This is precisely what was expected from Section 2.7. Figures 4.4 to 4.6 yield further confirmation of the theory presented in that section by demonstrating that the filter (0,0,...,0,1) produces the best results when applied to a maximum delay input.

The examples also illustrate that changing the position of the '1' in the initial filter does not affect which part of the wavelet is spiked up ; the process always chooses the position of the spike at the maximum amplitude of the wavelet, so that the spike location is beyond our control. This can be seen by examining the outputs of early iterations, although it would not be obvious from comparison of only iterations 5, 10 or 20 with the raw data. The entire output is seen to be delayed by precisely the number of zeroes preceding the '1' in the

33

initial filter ; this is not a surprise, since the application of any initial filter other than (1,0,0,...,0) delays the output at the first stage of the process. This effect will not cause problems in practice since each arrival will be delayed by the same amount, so that the relative spacing between arrivals is preserved.

Two arrivals

In Section 2.7 the question of the optimum position of the '1' in the initial filter was discussed for input data containing more than one arrival. It was suggested there that the position of the '1' may be chosen according to either the phase of the complete input data or the phase of the individual arrivals, and that it was difficult to predict with certainty which would be best. Figures 4.7 to 4.9 show the effect of varying the position of the '1' for input data which was produced by convolving the minimum delay wavelet with an impulse series consisting of two spikes of equal amplitude separated by zeroes. Figures 4.10 to 4.12 are similar except here the maximum delay wavelet has been used. The filter length in all cases is fixed at 18 points.
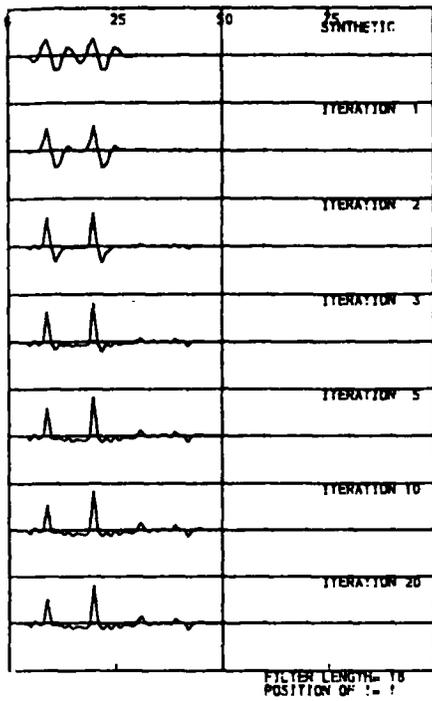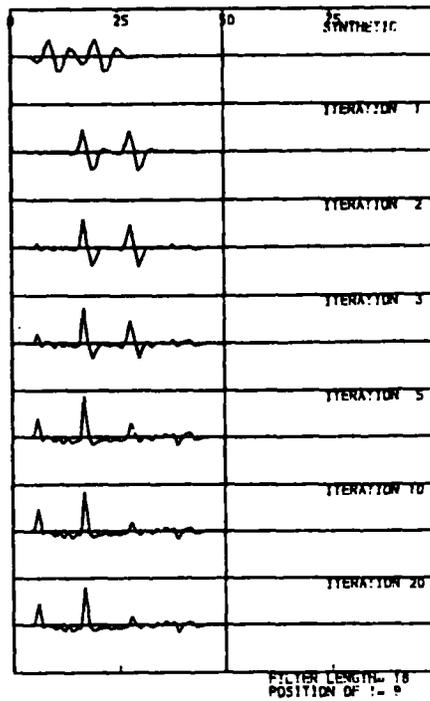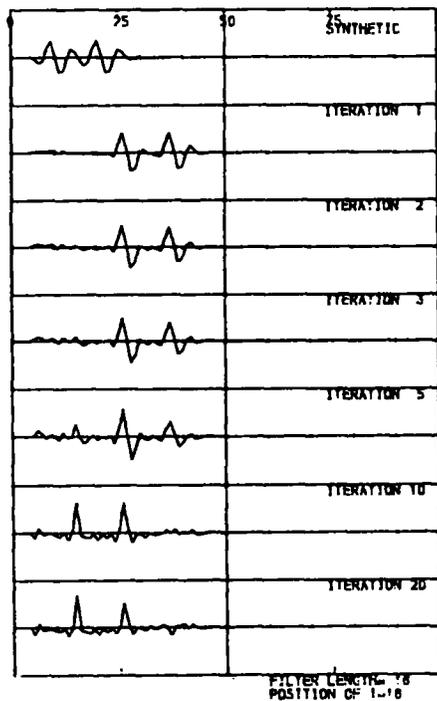
Fig. 4.7

Fig. 4.8

Fig. 4.9

Effect of position of '1'
on two interfering minimum
delay wavelets.
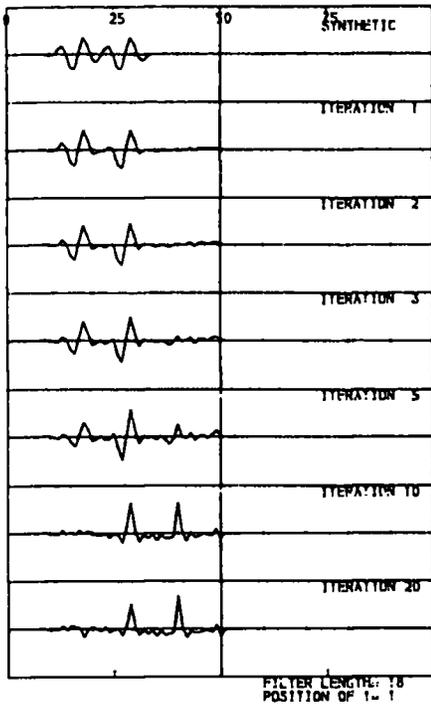
Fig. 4.10
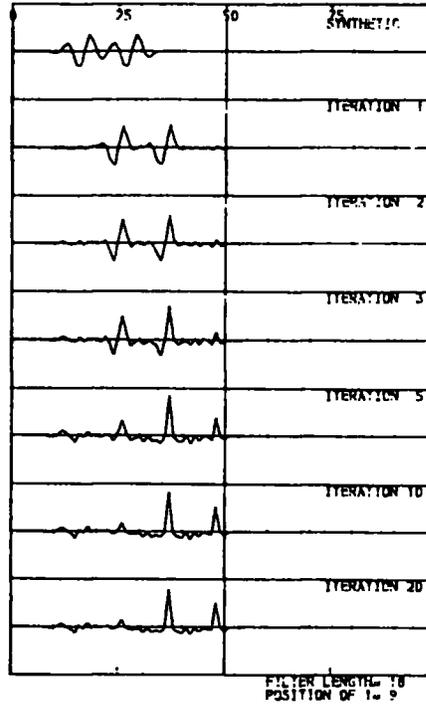
Fig. 4.11
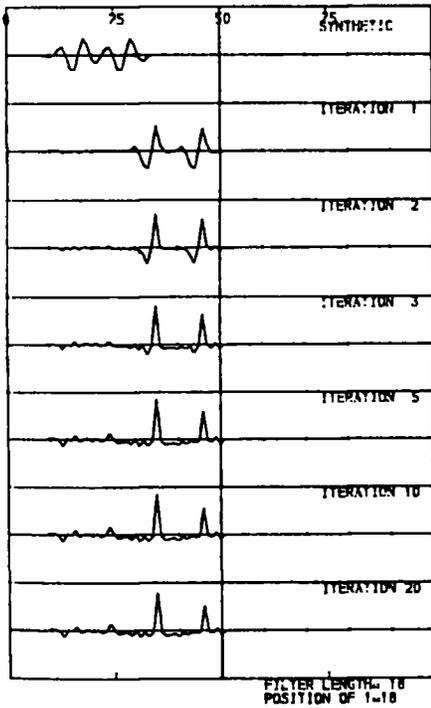
Fig. 4.12

Effect of position of '1' on two interfering maximum delay wavelets.

Discussion of results

Examination of the output after 20 iterations (Figs. 4.7 to 4.12) shows that after such a large number of iterations the position of the '1' does not greatly affect the result. As with single arrivals the output spikes are delayed with respect to the input data when the '1' is not in the leading position of the initial filter, but the delay is no longer simply equal to the number of zeroes preceding the '1'. The spacing between spikes is preserved in all cases at the correct separation of 10 zeroes. Closer examination reveals that the spikes are less sharp in Fig. 4.9 where the initial filter $(0,0,...,0,1)$ is applied to minimum delay arrivals, and the same effect is observed in Fig. 4.10 where the filter $(1,0,...,0,0)$ is applied to maximum delay arrivals. However, comparison of the spikes in Figs. 4.7 and 4.8 shows that they are equally sharp. It must be concluded that if the iterative process is continued for a large number of iterations, the choice of initial filter is essentially of no consequence.

From a practical point of view, however, the greater the number of iterations required the greater the computational time used. For application of the process to large amounts of real data the number of iterations performed becomes physically restricted by the time taken to apply the process. Figures 4.7 and 4.12 show that if the filter is chosen according to the phase of the

35

individual arrivals, the output converges to two sharp spikes after only 3 iterations and exhibits little change as the iterative process is continued. The other figures (4.8, 4.9, 4.10, and 4.11) show considerable change in the form of the output between iterations 3 and 10. Only after 10 iterations has the output stabilised to two spikes of a quality comparable with iteration 3 of Figures 4.7 and 4.12. It can be concluded that the rate of convergence of the process depends on the form of the initial filter. The more rapid convergence of the process associated with the appropriate choice of filter is a major advantage which will be gained by choosing the position of the '1' according to the phase of the individual arrivals.

As well as having the disadvantage of leading to slower convergence, inappropriate choice of the initial filter may lead to loss of arrivals. This is seen by considering Figures 4.8, 4.9, 4.10, and 4.11. For up to 3 iterations one could reasonably deduce that two arrivals are present, but iteration 5 is sufficiently poor to cast some doubt, and the same is probably true of iterations 4 and 6. The problem occurs as the position of the spikes undergoes a radical change - in Fig. 4.8 between iterations 3 and 5, and in Fig 4.9 between iterations 5 and 10 ; within this 'transition zone' only one arrival is particularly evident. No such problem is observed in Figures 4.7 or 4.12. A detailed explanation of this effect is beyond the scope of this thesis - all that can be said

is that in the act of maximising the Varimax Norm the process seems to 'change its mind' about the position of one of the spikes. The danger lies in the fact that it is the intermediate stages of the iterative process which fall within this 'transition zone' ; precisely those stages, in fact, to which one is likely to continue the iterative process in practice. The effect is one further reason for using an initial filter (1,0,0,...,0), since in this case the possibility of arrivals being lost occurs only after an excessive number of iterations. In practice the number of iterations will be restricted so that no such loss is likely.

4.3 Resolution

An important question about the MED process is how its ability to resolve interfering arrivals is affected by the length of filter used (Section 2.3). Theory sugests that if the filter length is such that it spans a signal consisting of two (or more) interfering arrivals, the process will ultimately reduce that signal to just one spike. The closer the arrivals the shorter the signal so that in this respect over-iteration becomes more likely for a given filter length. Strictly speaking the reduction of two arrivals to one spike in these terms is not so much a failure to resolve two arrivals as a tendency to over-simplify the data, but in practice the result is the same. Apart from this over-simplification

37

effect one might wonder if there comes a point when the arrivals become so close as to make them impossible to resolve, irrespective of the filter length used. This is the 'limit of resolution' in the conventional sense.

The problem is investigated using the synthetic examples which follow. In these the same minimum delay wavelet is used throughout, but the separation of the spikes in the impulse series is varied. As discussed above, reducing the separation of the spikes inevitably reduces the length of the complete signal. As a result it becomes difficult to differentiate between effects resulting from narrow separation of the spikes and effects resulting from the fact that the entire signal becomes shorter with respect to the filter length. In an effort to resolve this problem two filter lengths are used - length 18, which spans the complete signal in all cases but the first (Figures 4.13 to 4.18) and length 10, which does not (Figures 4.19 to 4.23). Of course we must be prepared to accept less sharp spikes from the 10-point filter, and this is indeed seen to be the case.
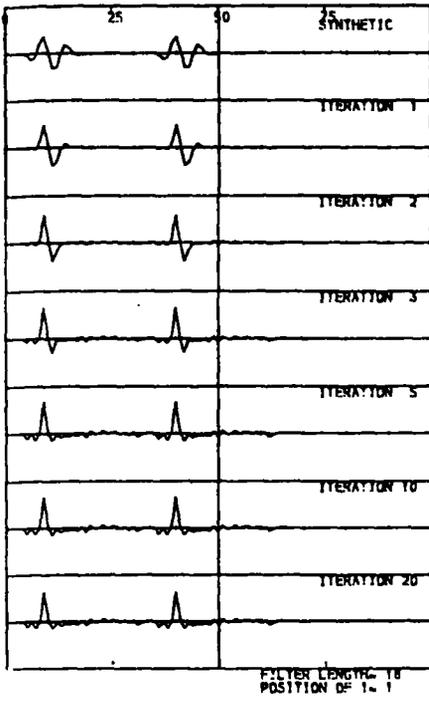
Fig. 4.13

Spike spacing = 30
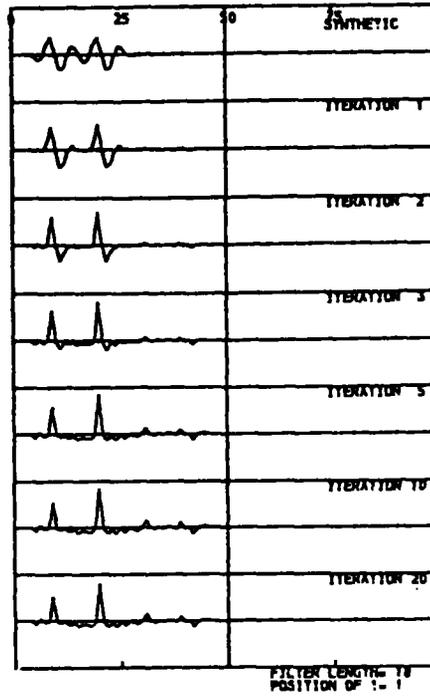
Fig. 4.14

Spike spacing = 10
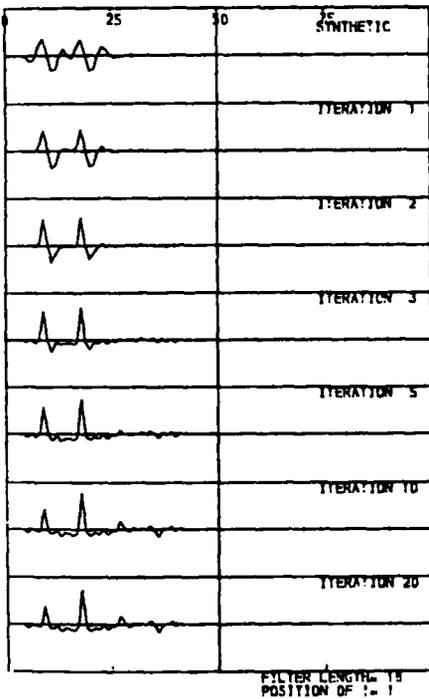
Fig. 4.15

Spike spacing = 8
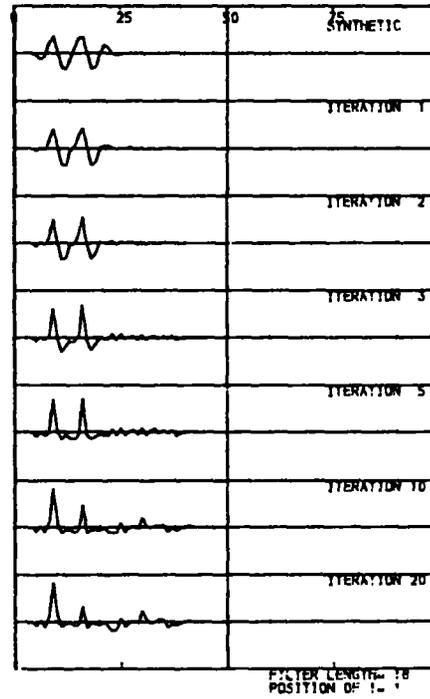
Fig. 4.16

Spike spacing = 6
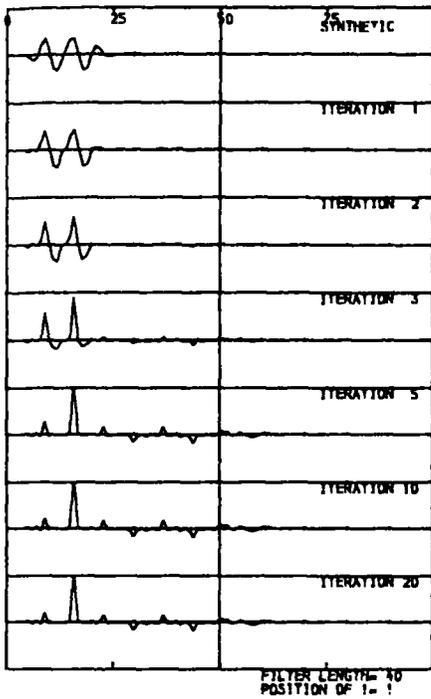
Fig. 4.17

Spike spacing = 6
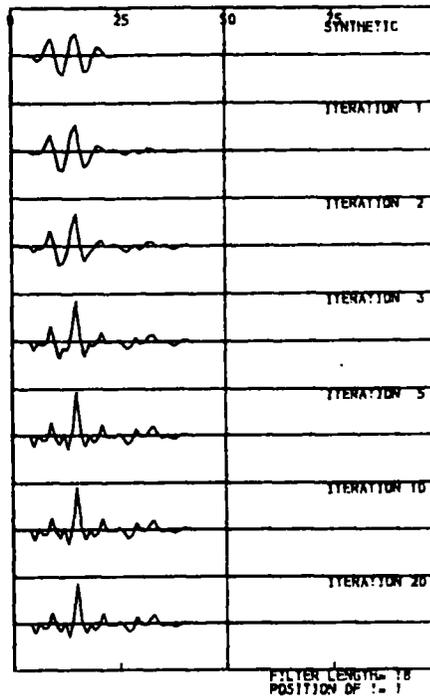
Fig. 4.18

Spike spacing = 5
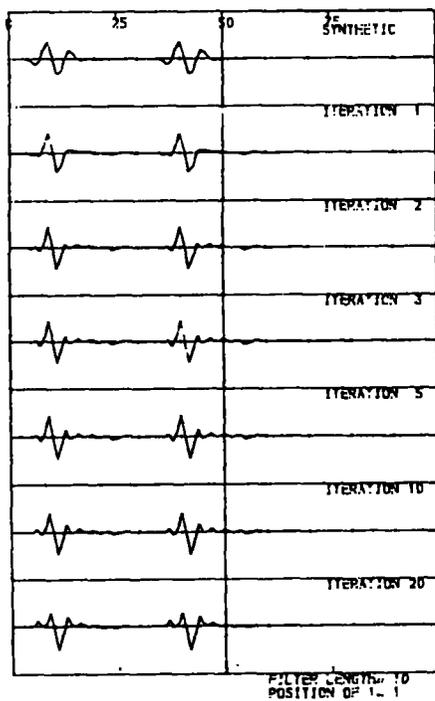
Fig. 4.19

Spike spacing = 30
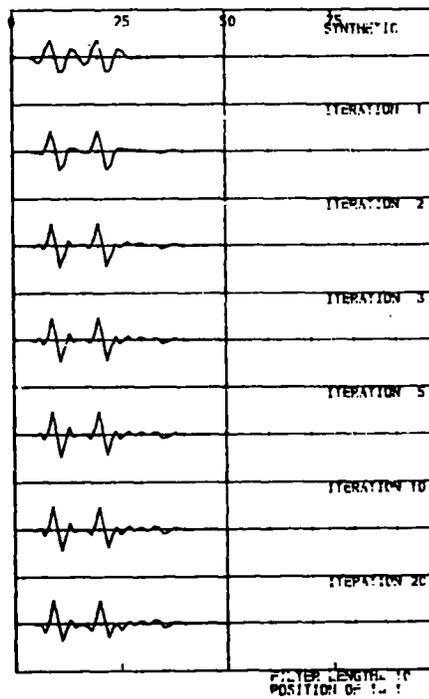
Fig. 4.20

Spike spacing = 10
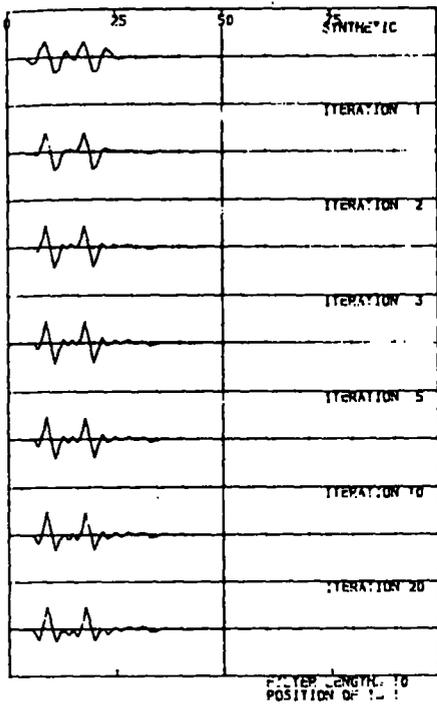
Fig. 4.21

Spike spacing = 8
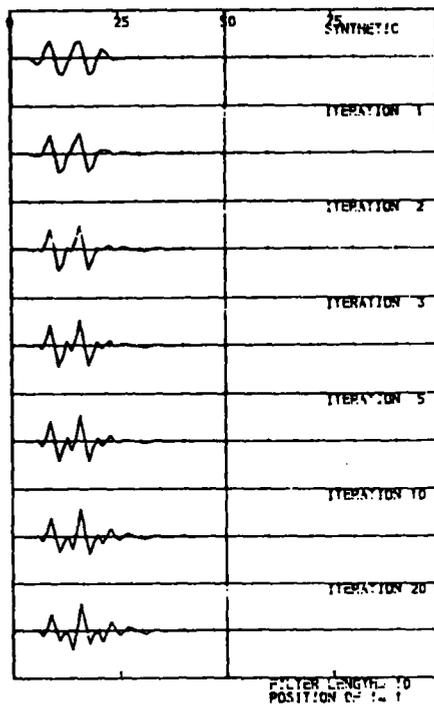
Fig. 4.22

Spike spacing = 6
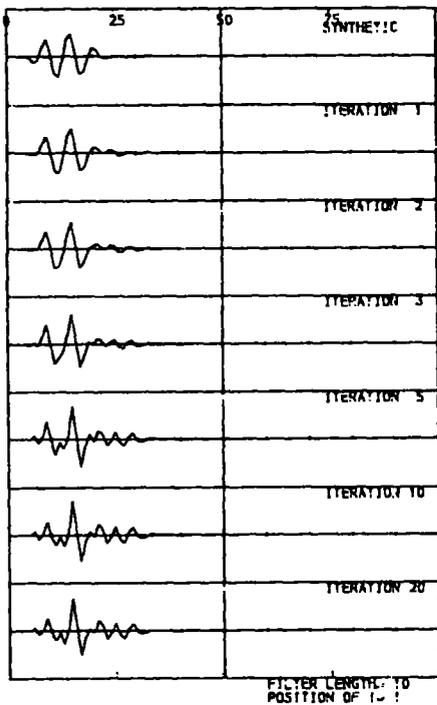
Fig. 4.23

Spike spacing = 5

Discussion of results

In Figure 4.13 the filter of length 18 clearly does not span the complete signal (the two arrivals). As a result the two spikes are very sharp and have equal amplitudes, even after twenty iterations. It is reasonable to suggest that the iterative process could be continued indefinitely without the data being reduced further, so that loss of arrivals is impossible in this case. The observation is the same for Figure 4.19.

In Figures 4.14 to 4.16 the filter effectively spans the complete signal and in all cases continued iteration results in the reduction of amplitude of one of the ouput spikes. If the iterative process were to continue further, eventually only one spike would remain. By increasing the filter length to 40 in Figure 4.17 it is shown that for a longer filter fewer iterations are required before reduction of one of the spikes begins. In practice, restricting the process to a small number of iterations would prevent such a possibility for these synthetic data cases, and the greater the filter length the smaller the number of iterations which should be used.

Figure 4.18 shows that the 18-point filter has been unable to resolve the two arrivals when the spacing is reduced to five points. The effect of reducing the filter length to 10 whilst maintaining the separation of arrivals at five points is illustrated in Figure 4.23. Comparison with Figure 4.18 shows a marginal improvement,

but in practice the reduction in length has done little to improve the resolving power of the process. It is concluded that in practice the ability of the filter to resolve two closely spaced arrivals is restricted not by the filter length but by the form of the signal which results from interference between the arrivals.

The examples shown indicate that there is little danger in using a filter which spans the complete signal and that using a filter shorter than this in an attempt to resolve arrivals which are very close together would be a fallacy. In practice it would be expedient to use a filter which is short compared to the signal containing the interfering arrivals, and compare outputs as the length is increased in order to guard against loss of arrivals.

4.4 Arrivals of Different Character

In Section 2.5 one of the properties of MED was said to be its inability to spike up two different wavelets in the same input data. This particular property is of exceptional importance in refraction seismology since the arrivals undergo considerable change along a given trace ; consequently it was felt desirable to apply the process to appropriate synthetic data for confirmation.

Figures 4.24 and 4.25 show that MED is indeed unable to operate effectively on two arrivals of different character. Little more can be gained from further

40

investigation ; however, it should be noticed that the
quality of the output spike for the second arrival is
improved by using an initial filter with the 'l' in the
middle location, so that presumably this arrival is mixed
delay.
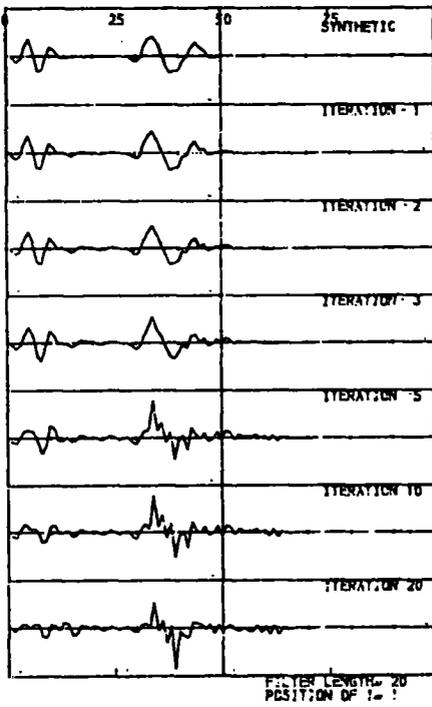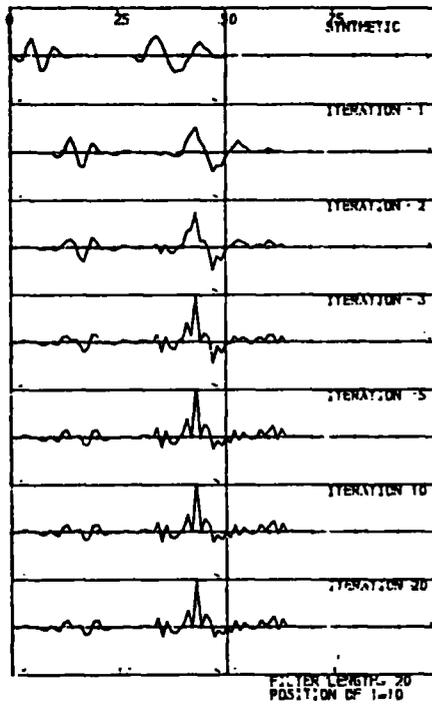
Fig. 4.24   Fig. 4.25

Effect of MED on two arrivals of different character

4.5 Summary

It has been shown that for minimum delay arrivals the initial filter should be chosen as (1,0,0,...,0). In terms of filter length the ability of a filter to resolve two interfering arrivals does not depend directly on the closeness of the arrivals, but rather on the form of the signal resulting from their interference ; for those arrivals which the process is able to resolve the sharpness of the spikes produced is improved as the filter length is increased. The upper limit to the filter length is set by the possibility of over-simplification of the data, leading to loss of arrivals. Suggestions in Section 2.3 that this limit is reached when the filter spans the interfering arrivals have been shown to be misguided in practice, since a very large number of iterations are required before such a problem arises. Finally, we have seen that the process is incapable of spiking up two different arrivals.

The work on synthetic data has enabled the following practical guidelines to be developed for application of the MED process to real refraction data :

1) Windows must be applied to the data within which little change of character of the arrivals occurs.

2) Since the source used in the CSSP produces a 'minimum - delayish ' source wavelet, (1,0,0,...,0) should be used as the initial filter. This is true despite the attenuation, dispersion and phase distortion undergone by

42

the different arrivals on a refraction trace because without detailed investigation of each arrival, minimum delay is the best guess we can make (Anstey, 1981).

3) Examination of interfering arrivals on each trace will enable a good initial estimate of filter length to be made. The filter can safely be chosen to have a length approximately equal to that length of data containing the two closest arrivals. For a small number of iterations this is likely to be an underestimate, and increasing the filter length from this lower limit will almost certainly improve the quality of the output without leading to loss of arrivals.

CHAPTER 5

APPLICATION OF MED TO REAL DATA

5.1 Introduction

Program MED was used to apply minimum entropy deconvolution to a total of three data sets. The two common-station sections, Station 38 and Station 47, were chosen because of the high quality of the data. The third data set, Event Kirkl, is a common-shot gather using a land shot (Kirkwhelpington shot). This was chosen in order to investigate the performance of MED on poor quality data with a low signal-to-noise ratio.

The results presented in this chapter include those which illustrate the effects of filter length, number of iterations and position of 'l' in the initial filter. These are necessary since the guidelines developed in the previous chapter were derived from relatively short, noise-free synthetic data. As a result, although the arguments presented there are likely to be valid in general terms, they really only pinpoint those areas where care must be taken when applying the process. Sections 5.4, 5.5 and 5.6 and associated figures, then, explain how the parameters for yielding the final results were chosen. Section 5.9 contains the final outputs together with a brief interpretation of the data.

5.2 Initial Considerations : Mode of Application

There are effectively two different ways in which the MED process might be applied to real data. On the one hand, each trace might be segmented into individual windows of data and the process applied to each window in turn, so that each trace, and even each window, is treated independently. Alternatively MED might be used as a type of matching filter. The latter technique was tried first.

The mechanism for application of the MED process as a matching filter was to pick a single arrival from a good quality trace. A narrow window was then applied to the signal and this was used as input to program MED. Using a filter length equal to the signal length produced an output of a single sharp spike. Passing on to the next trace a wider window was applied, centred about the position of the spike produced on the previous trace and of sufficient width to contain the same arrival taking into account the likely moveout between traces. By applying the filter derived from the previous trace to this window, it was hoped that any change in character of the arrival from one trace to the next would be small enough to enable the filter to spike up the arrival, so that it could be accurately located. Such accurate location would then enable the window to be made much narrower. Using this narrow window as the input data for the MED process a new up-dated version of the filter would

be derived, and the process continued to the next trace, and so on.

Only a small amount of time was spent in trying to implement the MED process in this way, but the results were disappointing due simply to the fact that the character of each arrival changed too much from trace to trace ; the same problem was encountered by Summers (1982) using common-shot airgun data. In any case, it was felt that none of the properties of MED singled it out as being especially suited to such an application, and since other matched filter techniques already exist (e.g.Warren, 1981) the problem was not considered further. It was decided that the peculiar properties of MED could be used to better advantage by applying the process to individual windows along each trace. With sufficient care in the choice of initial filter the process could be made completely automatic and still enhance not just first arrivals but subsequent arrivals as well; the results which follow illustrate the value of MED as a novel processing technique.

5.3 Windowing Procedure

The need to use only short segments of a refraction trace as input to the MED process has already been fully discussed (Section 2.6), and the length of window chosen must be based on the criterion presented there, namely that little change in character of the

46

arrivals occurs within the window.

In order to maintain MED as an automatic process requiring minimal human interference it was decided that the same window length would be applied all along each trace. As such, the choice of window length was inevitably one of compromise. Examination of the unprocessed data in Figures 5.1(a) and 5.2(a) shows considerable variation in the length of each arrival, the wide angle reflections in particular being of greater length than the refracted arrivals.

It is important to appreciate how the length of window used may affect the output fom the process. As illustrated in Section 4.4, if the window is so long as to contain arrivals of different character only one of the arrivals will be spiked up, and even then the quality of the spike produced may be quite poor. As a result arrivals may be lost. At the other extreme, if the window is so short as to contain only part of a single arrival, the MED process will still yield a spike for that part, and it will also yield a spike in the adjacent window for the other part. No amount of iteration nor increasing of filter length will then be able to reduce these spikes to one ; in this case the process invents arrivals.

In pratice a window length of 100 points was used. This choice was based on the observation that such a length seems to span the wide angle reflection arrivals (see, for example, SHOT N11, Fig.5.1(a)), whilst at the

47

same time allows little change in the character of arrivals to occur (e.g. the first two arrivals in SHOT N1, Fig.5.1(a)). It should be added that little attempt was made to experiment with window length since the length of 100 points produced reasonable results, but there is clearly scope for further experimentation in this area.

In order to reduce truncation problems a cosine-bell taper of 10 points was added to either end of each window. Eight windows were applied to each trace with an overlap of 20 points between windows as indicated in Figure 5.2.
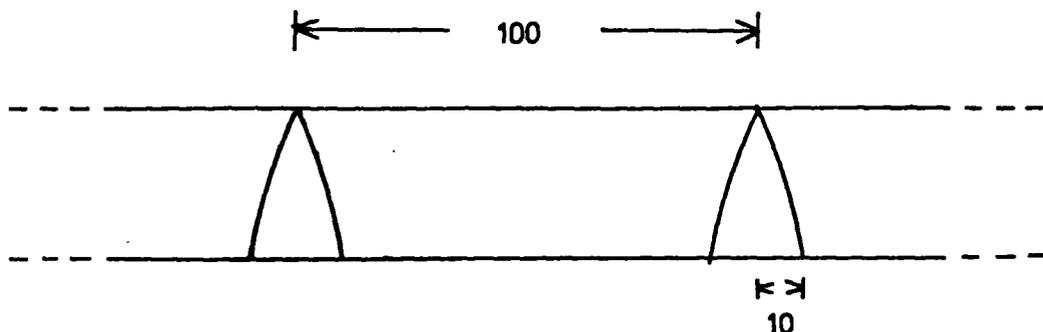
Fig. 5.2

A problem associated with any automatic windowing procedure is that an arrival may lie by chance at the junction of two windows. In the case of MED this produces complications which depend on the length of the arrival concerned. For a short arrival, the effect of the tapered ends of the window is likely to reduce the signal so that arrivals are lost. For a long arrival the splitting caused by the windowing procedure will result in

a spike being produced in each window, as discussed earlier.

The practical solution to this problem was to apply a second set of windows identical to the first set, but shifted along by 50 points. The outputs from the two window sets were then added together and divided by two, to yield the final output from the process.

The solution is by no means perfect, for a number of reasons. Considering a short arrival lying at the junction of two windows in the first window set, this arrival will lie directly in the middle of a window in the second set, so that the arrival will not be lost. However, for this particular arrival, a spike is produced only in the second window set, whilst it is possible that other similar arrivals may be spiked up in both sets. Consequently, the amplitude of the output spike for this arrival is reduced relative to the others. This is not a problem, but it indicates that we should not be surprised if sudden variations in amplitude are observed when following an arrival from one trace to the next on processed sections. One should certainly not try to deduce anything from such variations.

For a long arrival lying at the junction of two windows in the first window set, again this will lie directly in the centre of a window in the second window set. The data from the second window set will be reduced to a single spike, whilst the first window set will yield

49

two spikes from the one arrival. Adding the outputs of the two window sets and dividing by two will not remove the spikes produced from the first window set, which will lie to either side of the spike produced from the complete arrival ; a consequence of the windowing procedure is that for an arrival with a length of the order of the window length, the process will be unable to reduce that arrival to a single spike.

5.4 Filter Length

In Section 4.5 it was considered that a safe lower limit for the length of filter to use is one which is approximately equal to the length of data containing the two closest arrivals. It was further suggested that adhering to such a limit is likely to be over-cautious in that a considerably longer filter is unlikely to lead to loss of arrivals, whilst at the same time is able to produce much sharper spikes.

Examination of the unprocessed section in Figure 5.1(a) shows two very conspicuous arrivals (the first and second arrivals) in SHOT N1 and SHOT N2. Although it becomes more difficult to distinguish the two arrivals, one can reasonably deduce that the first signal in SHOT N7 is a result of interference between them. A filter length of 20 points easily satisfies the lower limit discussed above with respect to the two interfering arrivals. Further examination of the complete section suggests that

50

such a length is unlikely to lead to over-simplification in any of the traces.

The effects of using a filter of length 20 are shown in Figure 5.1(b) after 5 iterations. The number of iterations was set at 5 after a limited amount of experimentation with individual traces, which showed that little change in the output was observed after this number. In this respect the initial choice of the number of iterations to use was not entirely arbitrary, but the important point is that such a small number of iterations was most unlikely to lead to loss of arrivals when using such a short filter. Figures 5.1(c) to 5.1(e) show the results of increasing the filter length in steps of 10 up to a limit of 50 whilst keeping the number of iterations fixed at 5.
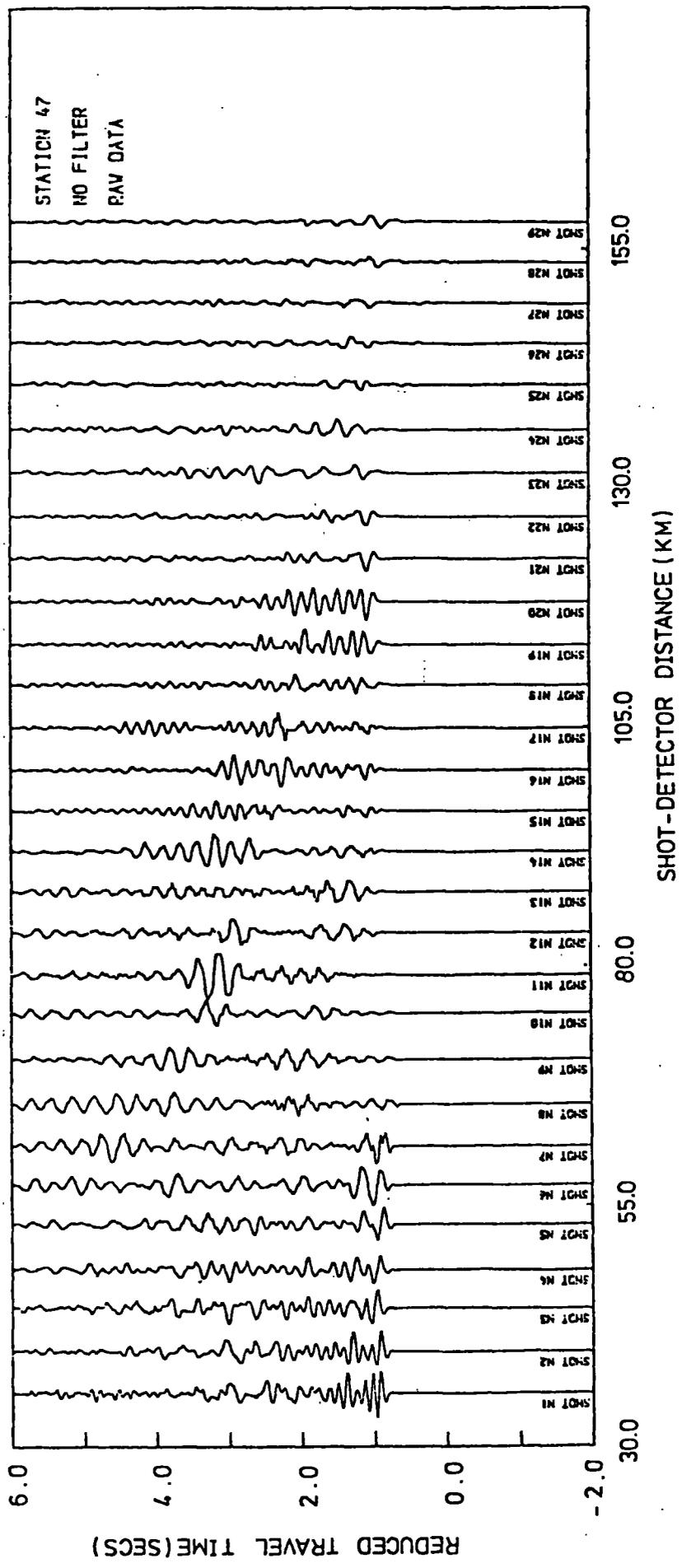
Fig. 5.1(a)

Unprocessed data from CSSP.
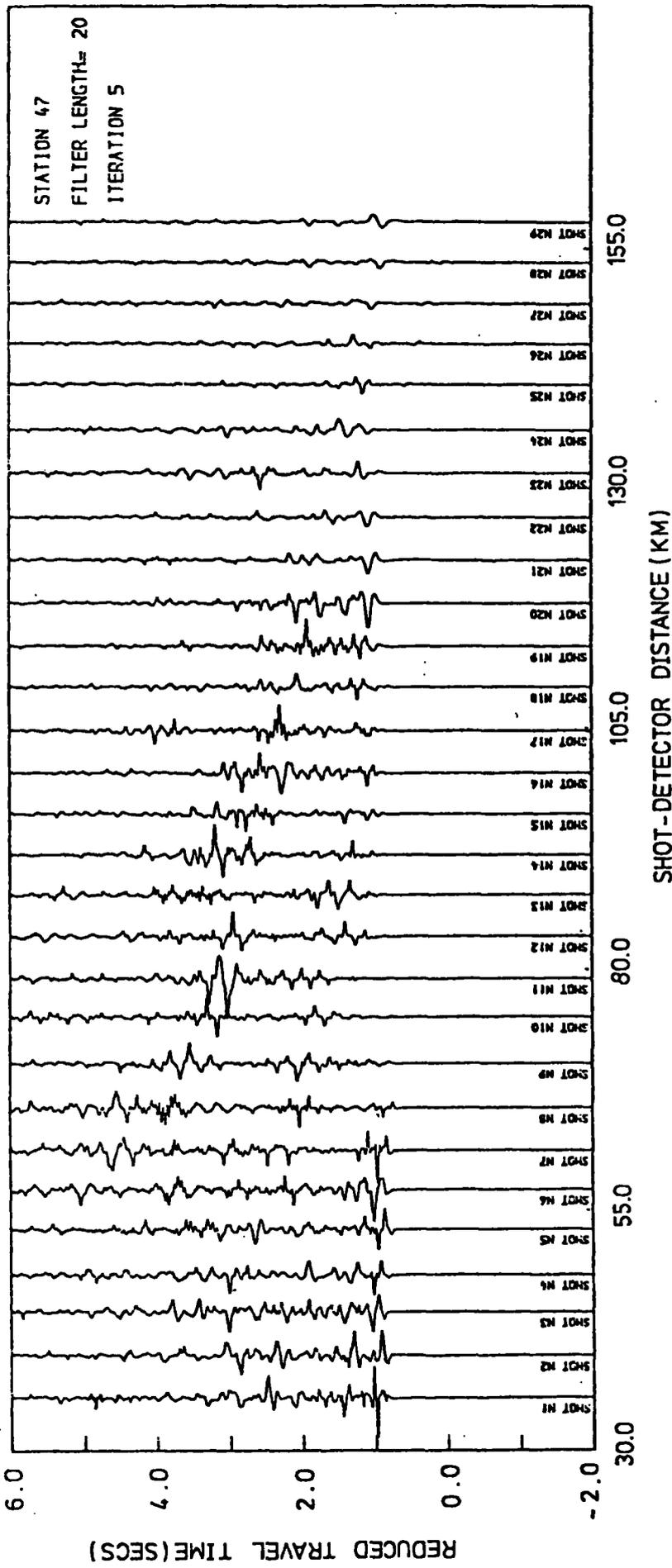
Fig. 5.1(b)

MED applied – effect of filter length.

STATION 47
FILTER LENGTH= 30
ITERATION 5

SHOT-DETECTOR DISTANCE (KM)

REDUCED TRAVEL TIME (SECS)
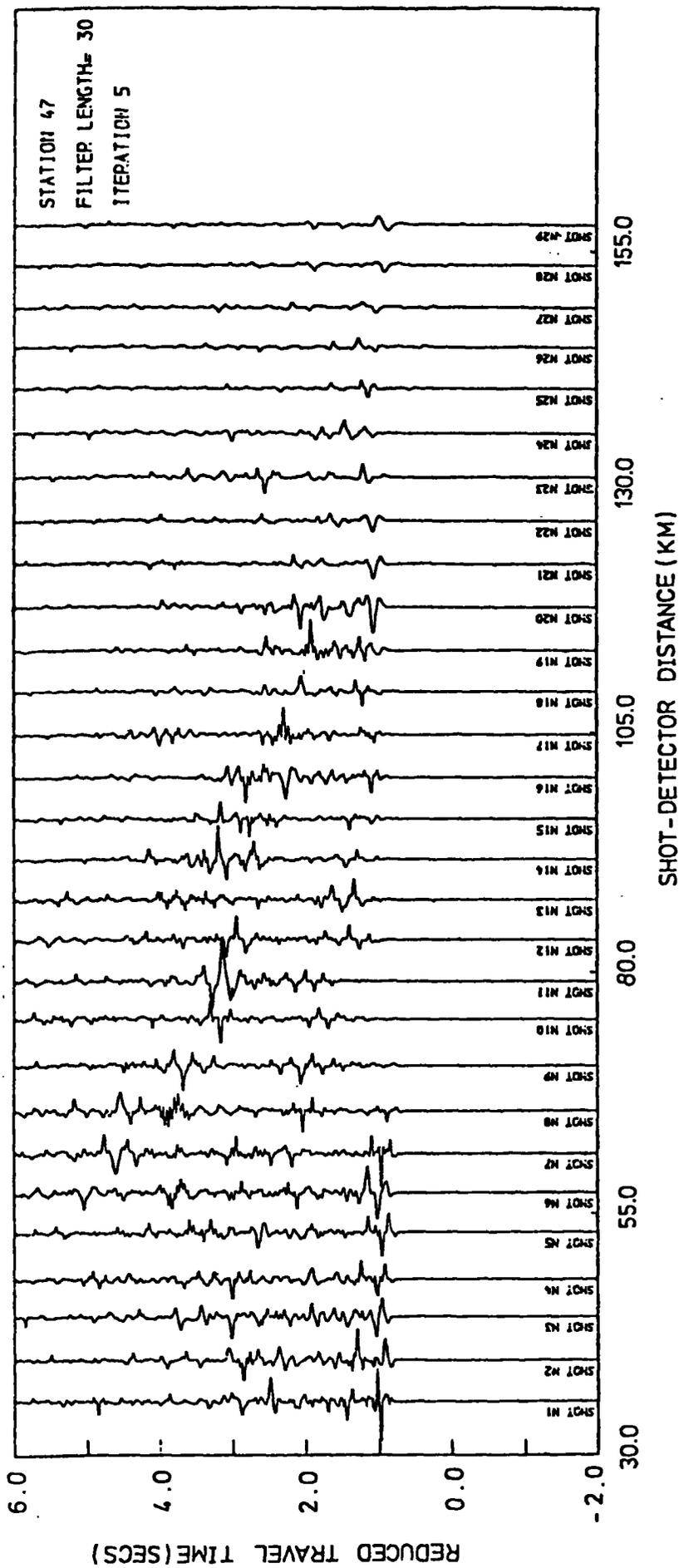
Fig. 5.1(c)

MED applied - effect of filter length.
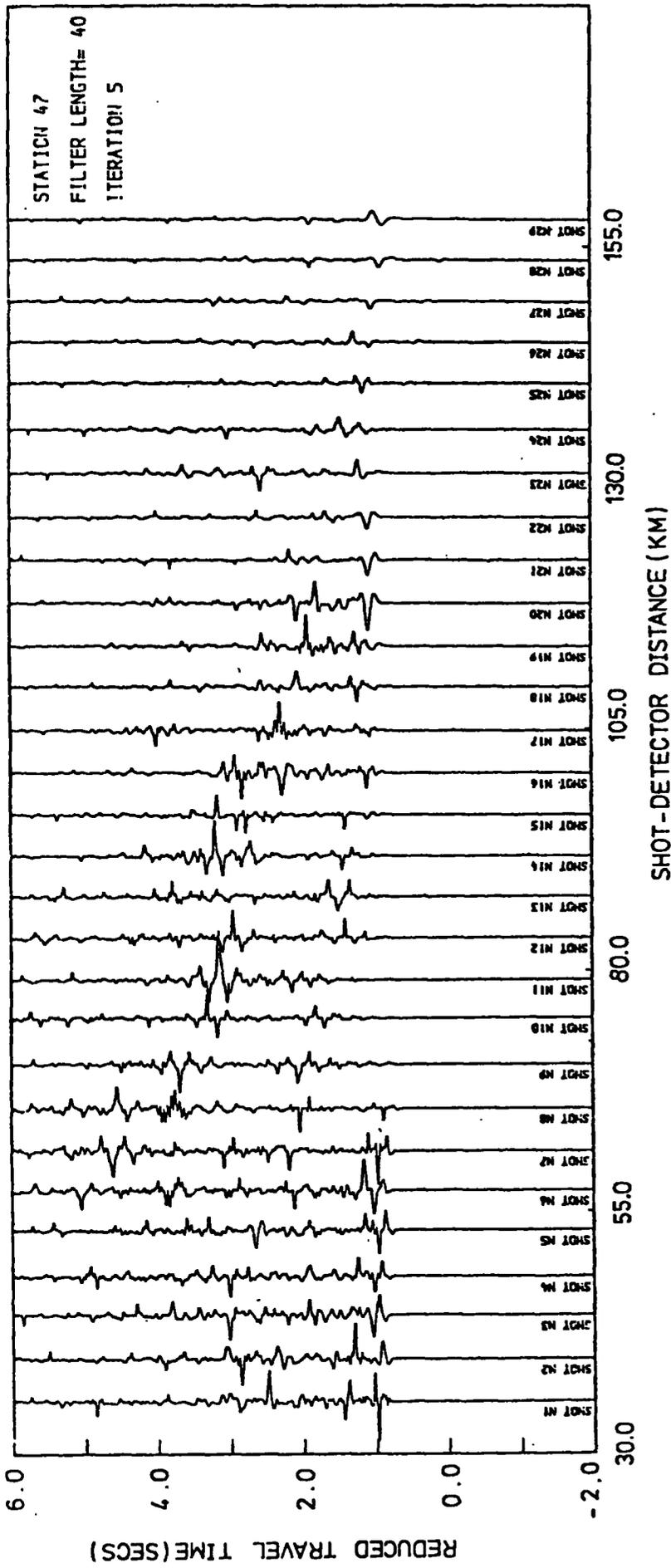
Fig. 5.1(d)

MED applied - effect of filter length.
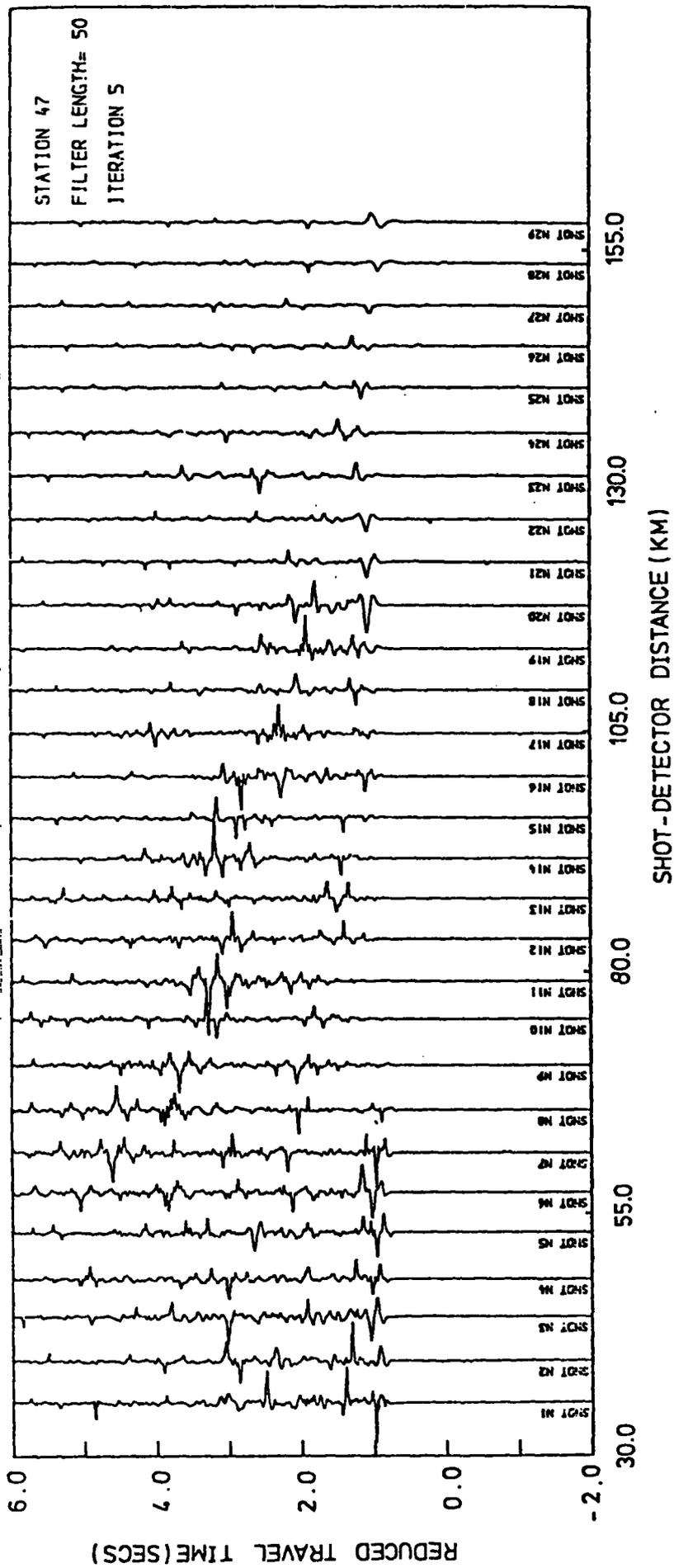
Fig. 5.1(e)

MED applied - effect of filter length.

Discussion of results

Comparison of Figure 5.1(a) with Figure 5.1(b) shows that even using a short filter, the processed data represents an improvement on the unprocessed data in terms of the ease with which arrivals may be picked. In particular, the MED process has done an excellent job of distinguishing the first two arrivals seen in shots N1 to N7. Elsewhere the major peaks in the data have been amplified with respect to adjacent values. Each arrival in the unprocessed section is represented by a spike in the processed section, but each spike in the processed section does not necessarily represent an arrival. Correlation from one trace to the next enables true arrivals to be distinguished from spurious spikes.

As the filter length is increased, so the results are improved. The output from the process using a filter of length 50 is considerably cleaner than that from a filter of length 20. The first two arrivals in shots N1 to N7 have been exceptionally well resolved, and the spikes as a whole have been considerably sharpened throughout the section. In addition the increased length does not seem to have led to over-simplification of the data.

The quality of the results in Figure 5.1(e) led us to believe that there would be little point in increasing the filter length further, bearing in mind the increased computational time this would involve. It was

decided that a filter of length 50 was the best length to use for this data ; since the data from Station 38 is very similar the same length was used there (Figure 5.2(b)).

5.5 Number of Iterations

In Section 4.3 it was seen that up to a certain stage in the iterative process, the greater the number of iterations performed the sharper the spikes produced and the larger their amplitude. Clearly such improvements in the spike quality are desirable in that the arrivals become easier to pick as a result. However, for data containing more than one arrival a reduction in amplitude of one of the spikes resulted when the iterations exceeded some number. If iteration were to continue arrivals would be lost. Comparison of Figure 5.1(e) with 5.1(a) suggests that no arrivals have been lost in this case, but it is possible that iterating five times may have pushed the output past its optimum and reduced the amplitude of some of the spikes. To investigate this possibility, the results from 2, 3, and 4 iterations were plotted, and are shown here as Figures 5.1(f), 5.1(g) and 5.1(h).
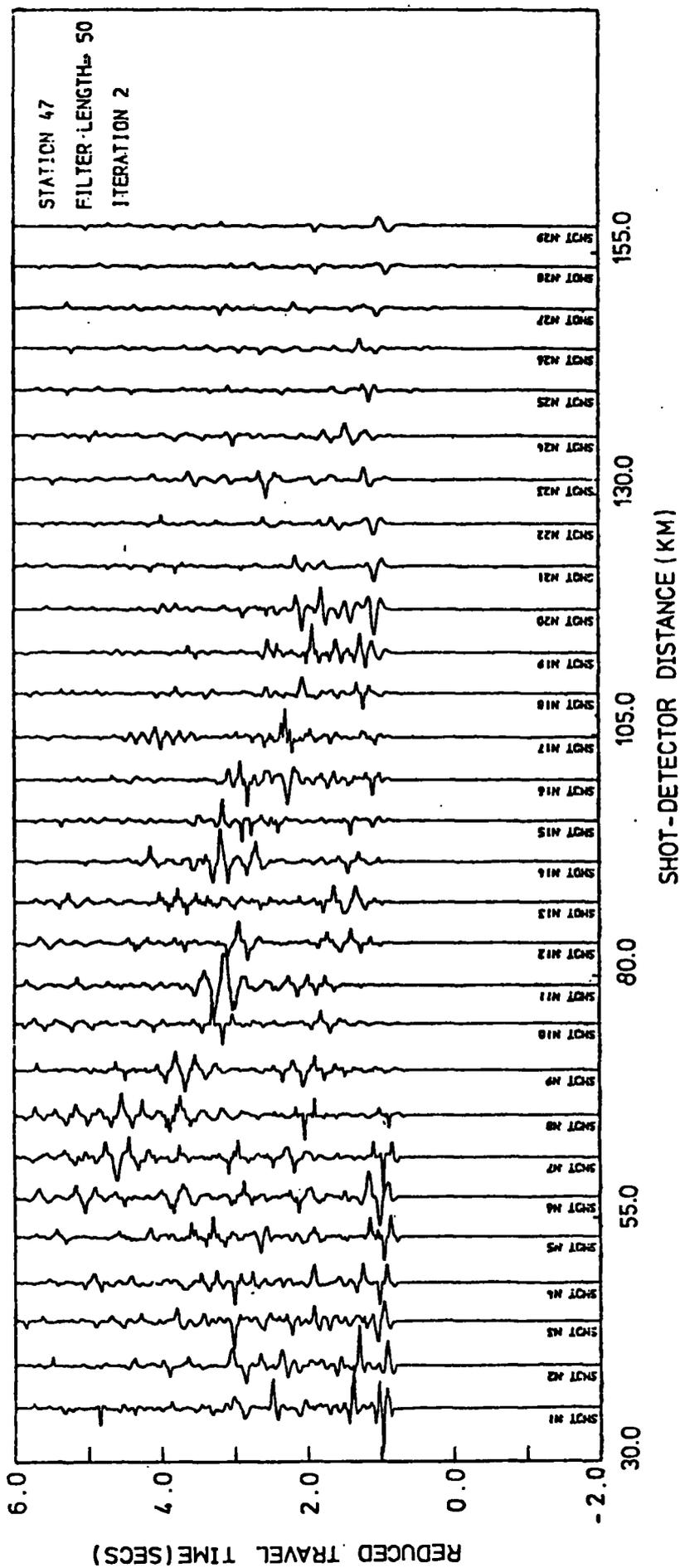
Fig. 5.1(f)

MED applied - effect of number of iterations.
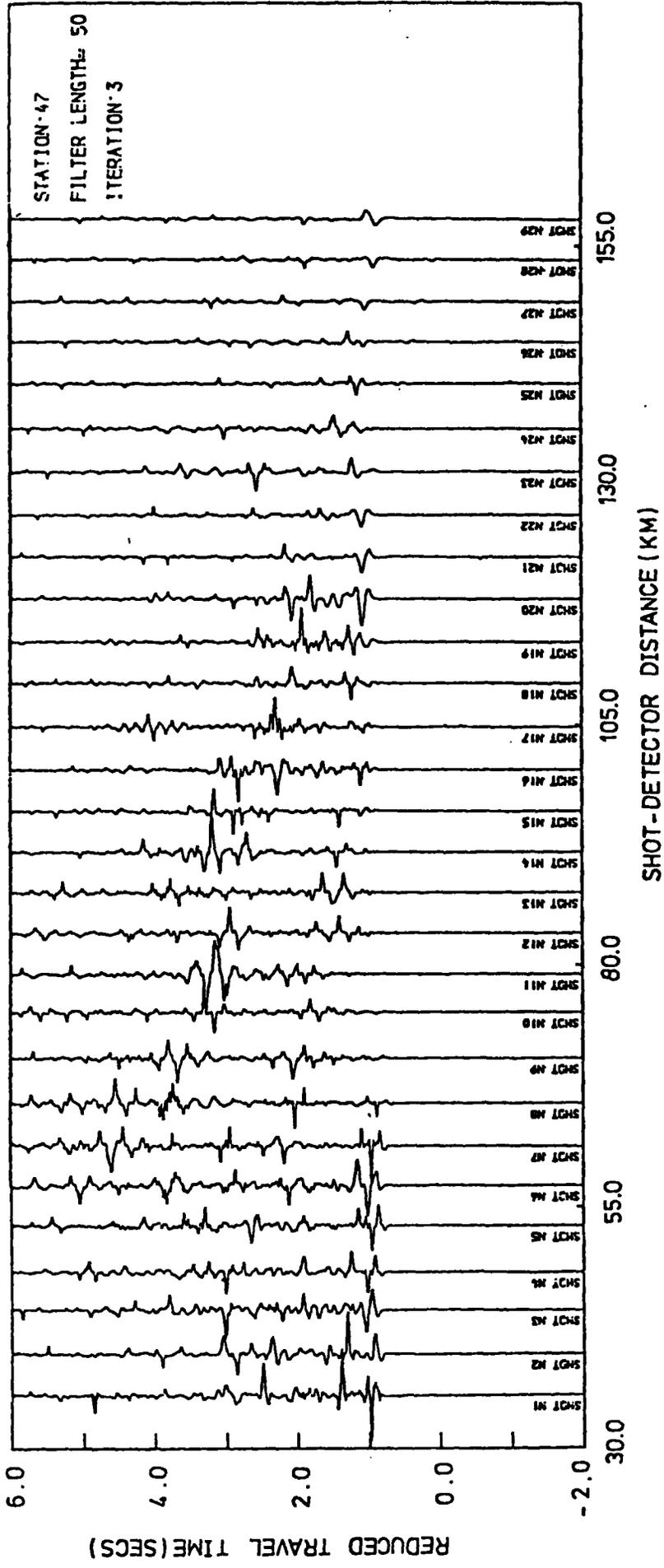
Fig. 5.1(g)

MED applied - effect of number of iterations.
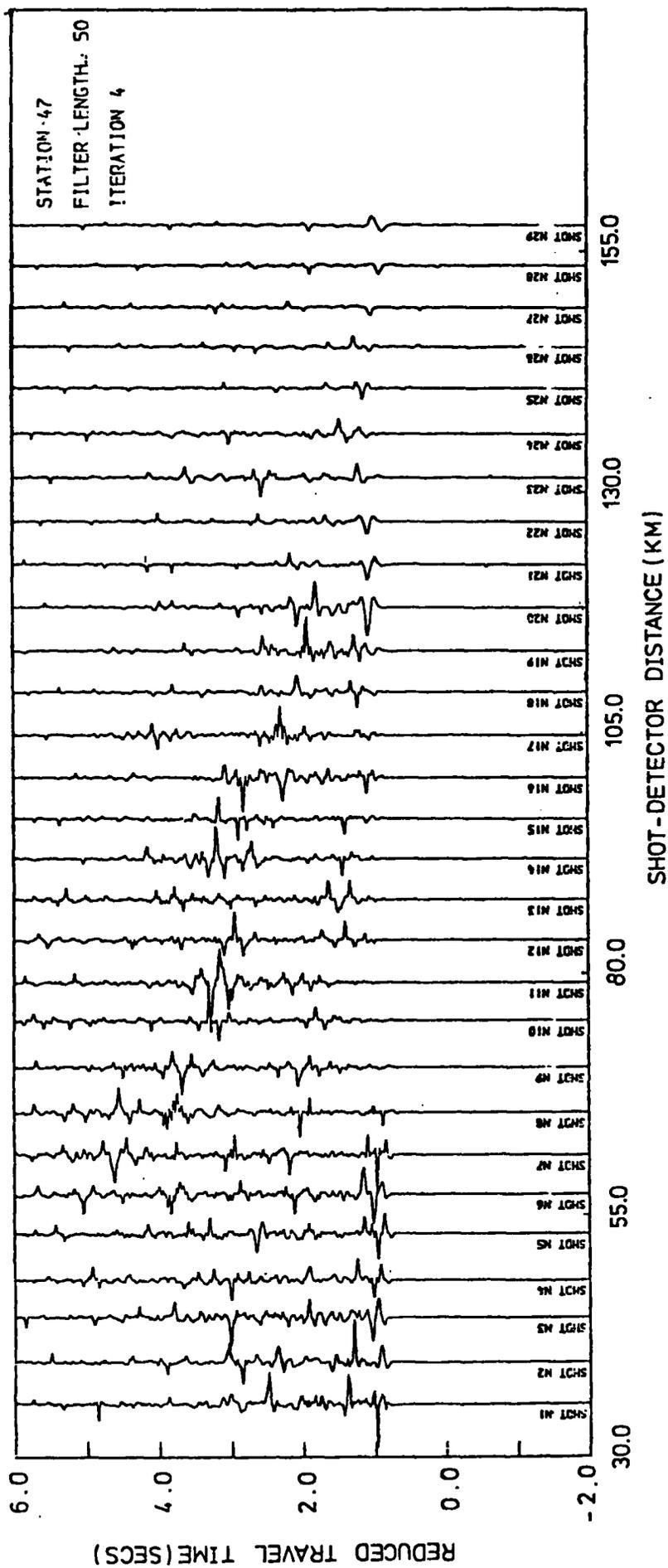
Fig. 5.1(h)

MED applied – effect of number of iterations.

Discussion of results

A comparison of 5.1(f), 5.1(g), 5.1(h) and 5.1(e) illustrates a number of points. Firstly, as the number of iterations is increased, so there is an increase in the amount of high frequency noise between spikes. This is a general property of all such deconvolution techniques and is inevitable, since making the spikes sharper necessarily requires the introduction of higher frequencies into the data (Anstey, 1981). Secondly, as the iterative process is continued it is seen that the output becomes 'cleaner' i.e. the number of spikes is reduced and those that remain are sharper.

There is little to choose between iterations 4 and 5 and indeed this is a good indication that the output is converging at this stage in the iterative process. Consequently there is no point in iterating further since to do so will either yield negligible change in the output or produce a reduction in amplitude of some of the spikes as discussed above. We conclude that 4 or 5 iterations with a filter length of 50 points yields the best results for this data.

5.6 Position of '1' in Initial Filter

In all the results discussed so far the choice of initial filter has been (1,0,0,...,0). It was concluded in Section 4.5 that this was the best choice for CSSP data. For completeness, the effects of using an initial filter with the '1' in the middle location are shown in Figures 5.1(i), 5.1(j) and 5.1(k), where the iterative process has been continued for 2, 3, and 5 iterations respectively.
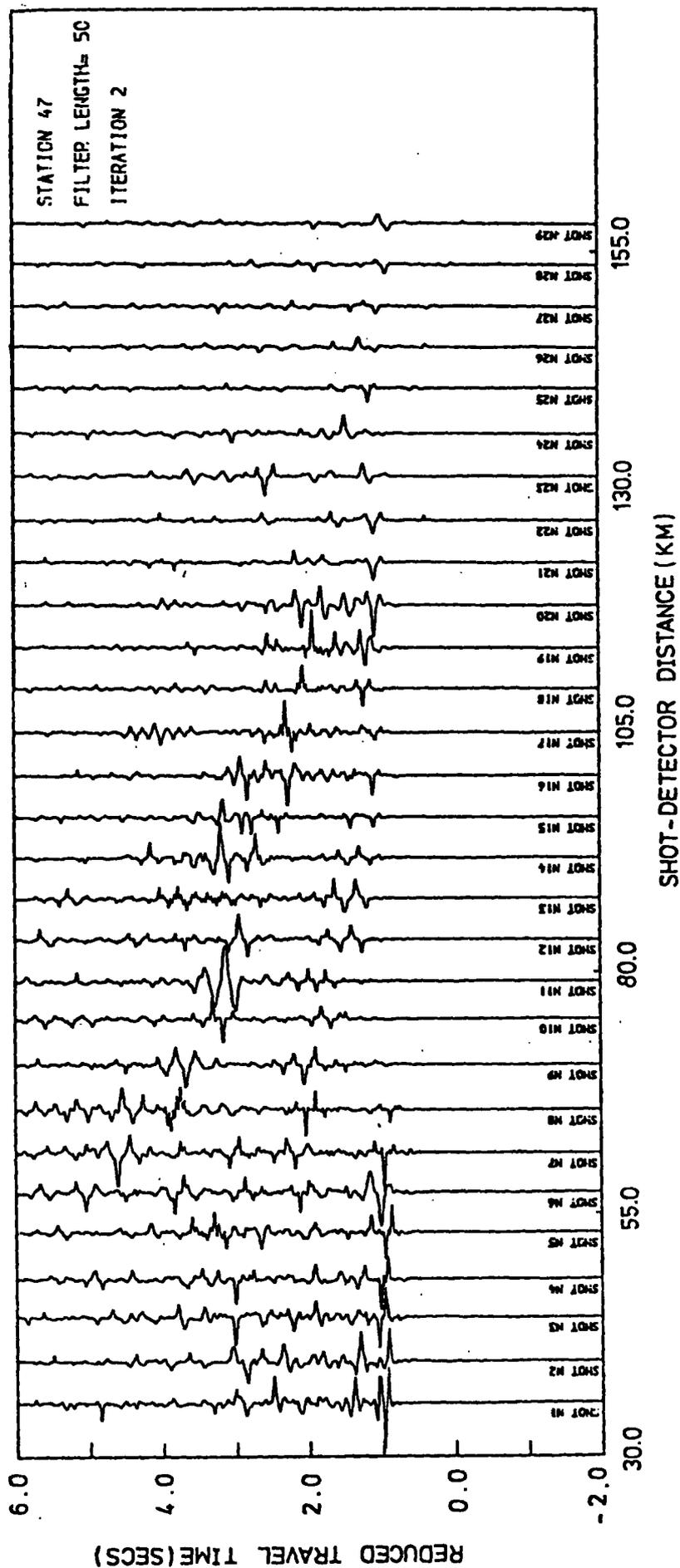
Fig. 5.1(i)

MED applied - effect of position of 'l' in initial filter.
'l' in middle location.
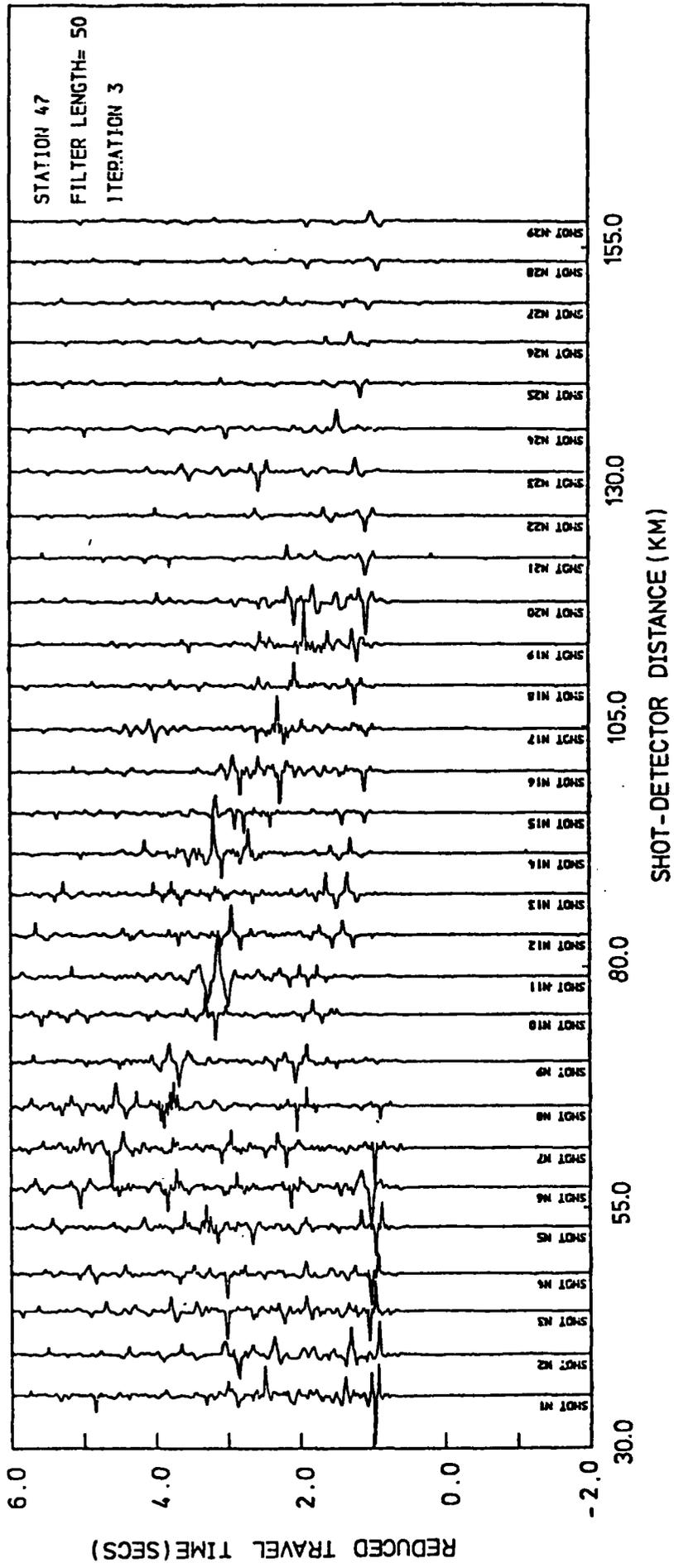
Fig. 5.1(j)

MED applied - effect of position of 'l' in initial filter.
'l' in middle location.
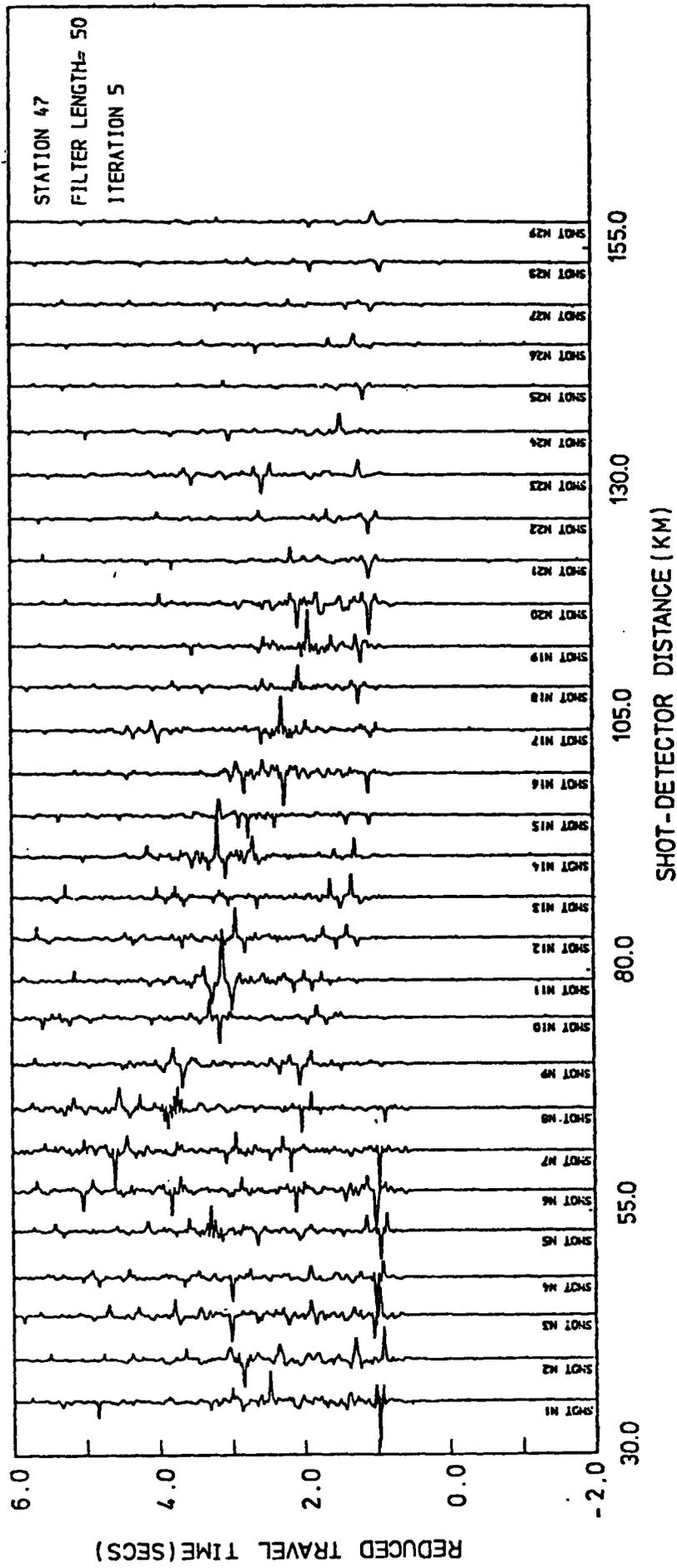
Fig. 5.1(k)

MED applied - effect of position of '1' in initial filter.
'1' in middle location.

Discussion of results

Comparison of the output from each iteration with the output from the same number of iterations using initial filter (1,0,0,...,0) shows that the choice of initial filter (0,...1,...,0) does a poorer job of spiking up some arrivals (especially the first two interfering arrivals of shots N1 to N7), whilst it does a better job of spiking up others. Comparing specifically the results from iteration 5 (Figures 5.1(k) and 5.1(e)) it is generally true that a filter with the '1' in the middle location produces sharper spikes from those arrivals lying towards the tail end of each trace (e.g. T = 5 sec., SHOT N6), and from those arrivals at greater shot-detector distances (e.g.SHOT N20). This is presumably due to the fact that the long distances travelled by such arrivals have invalidated the assumption that they are approximately minimum delay.

However, such improvement in spike quality is somewhat marginal, and in any case does not really make it any easier to pick arrivals. More importantly, the so-called 'transition zone' discussed in Section 4.2 has manifested itself in terms of lost arrivals. This is particularly evident in the first two arrivals of shots N1 to N7 in Figure 5.1(k) ; after 5 iterations, the second arrival has been effectively lost in all but SHOT N2. This is a very good reason for not choosing (0,...1,...,0) as the initial filter.

5.7 The Effect of Bandpassing

In Section 2.6 it was suggested that applying a bandpass filter to real data before application of MED would be undesirable if the signal-to-noise ratio were already good. The effect of bandpassing in practice is demonstrated using data from Station 38.

Station 38 is shown unprocessed in Figure 5.2(a). The quality of the data is comparable to that of Station 47. Figure 5.2(b) shows this data after application of MED, using a filter of length 50 and 5 iterations, with the window geometry as before. Again the results are good, particularly with respect to the resolving of the first two arrivals in shots N1 to N7. The output from applying MED to bandpassed data is illustrated in Figure 5.2(c).

At first sight the difference between Figures 5.2(b) and 5.2(c) may seem to be fairly negligible, and in practice in this case it is. However, closer examination reveals that the spikes produced from the bandpassed data are less sharp than those from the unbandpassed data. Frequently a bandpass filter is applied to refraction data on the assumption that it will always improve the sections. It has been shown here that far from improving the data, the application of a bandpass filter prior to MED reduces the efficiency of the MED process.

57
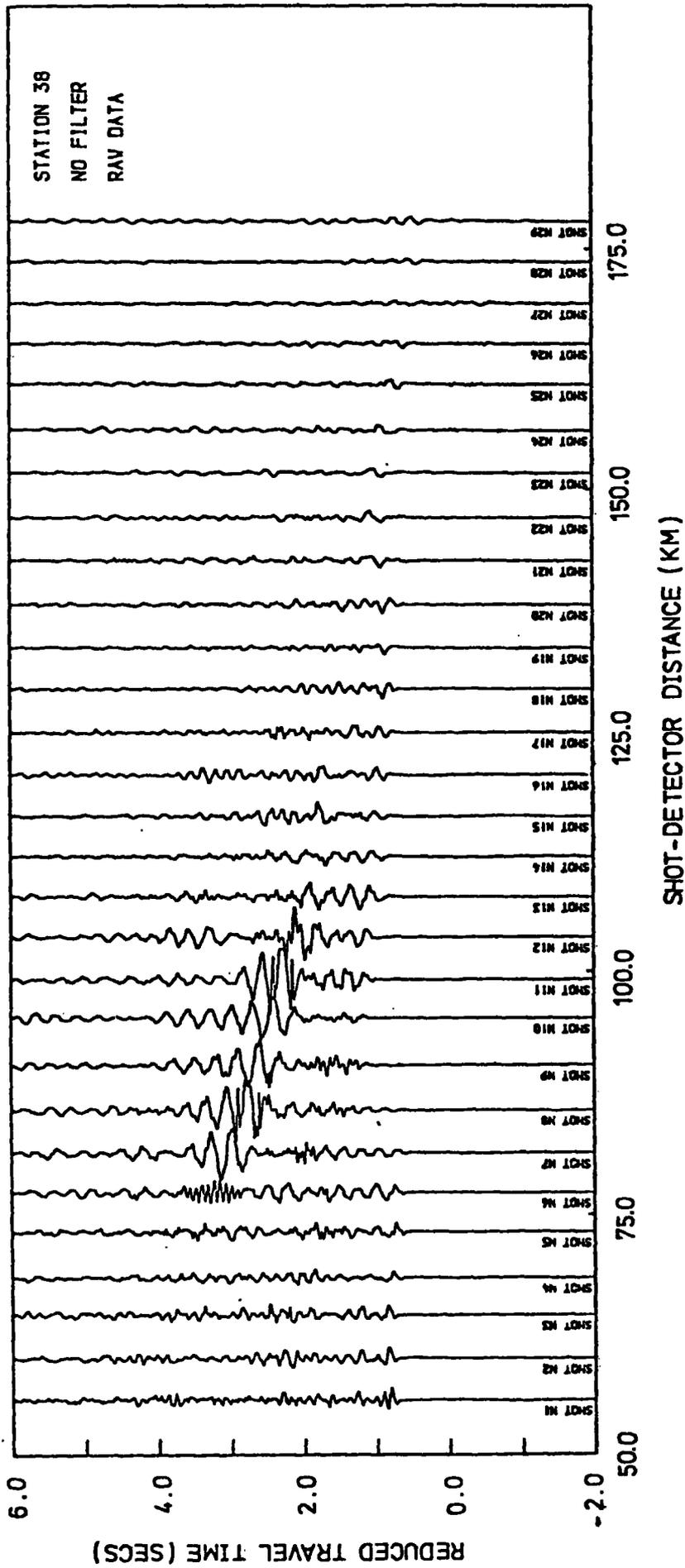
Fig. 5.2(a)

Unprocessed data from CSSP.
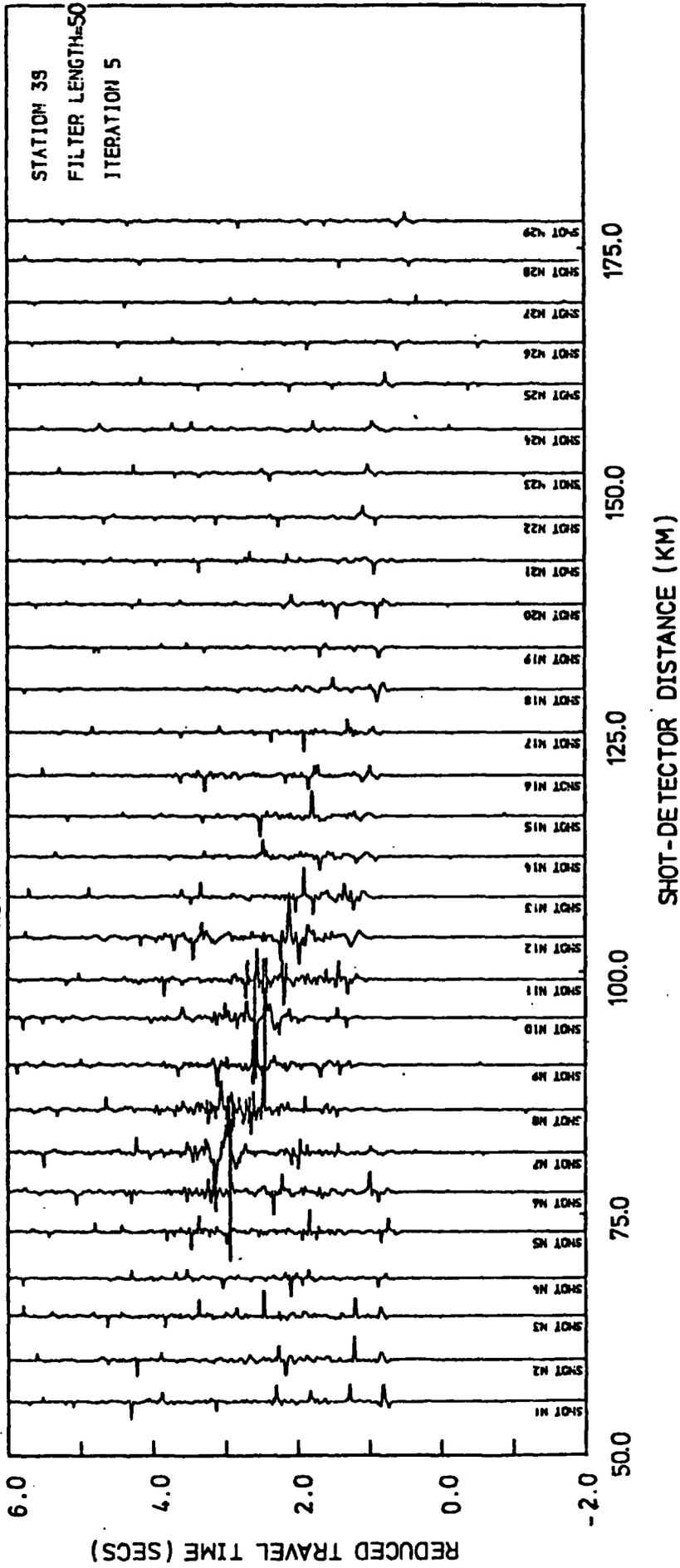
Fig. 5.2(b)

MED applied to unbandpassed data.
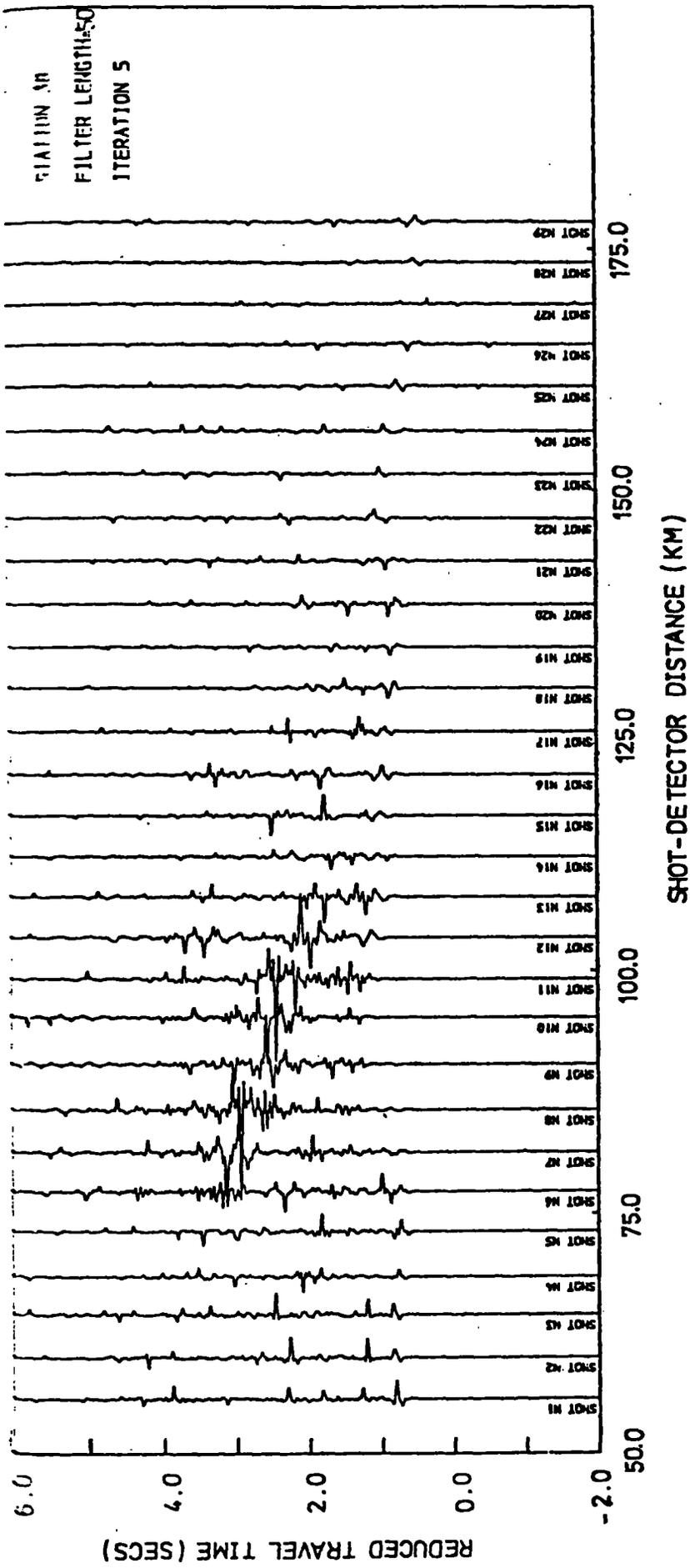
Fig. 5.2(c)

MED applied to bandpassed data.

5.8 Application to Noisy Data

Figure 5.3(a) (Event KIRK1) shows the very poor quality data obtained from the Kirkwhelpington land shot of the CSSP. Equalization has been carried out and a bandpass filter applied in the range 3.0 to 50.0 Hz in order to remove low frequency 'roll' discussed in Section 2.6. It is difficult, on this data, to follow any arrival other than the first.

The effect of MED is shown in Figure 5.3(b), and it is seen that the process has done little to improve the situation. The problem is that so many spikes have been produced that one could draw a line from trace to trace virtually anywhere. This is a further consequence of windowing ; if the maximum amplitude of the signal is less than the maximum amplitude of noise within the window, the process will spike up not the arrival but just that portion of the noise with the largest amplitude. The same is true if no signal is present at all.

One might wonder, then, why such spikes were not produced in Stations 38 or 47 when the window contained no signal - the fact is that in these cases the amplitude of the noise was so low that the cubing process inherent in MED had little effect. Nevertheless, looking back to Figure 5.1(e), small spikes have been produced from what is presumably just noise ; this is particularly evident after T = 5 sec. The important point is that in Stations 38 and 47 the amplitude of the spikes in question are

negligibly small compared to the spikes produced from arrivals.

The conclusion is that MED will only work effectively when the signal-to-noise ratio is high. It is conceivable that the application of a suitably chosen bandpass filter to remove some of the high frequency noise in the Kirkwhelpington data might lead to improved results, despite the associated reduction in sharpness of spikes which would result. Lack of time prevented the investigation of such a possibility. In any case, the point of the example is to show that the application of MED to poor quality data is ineffective ; there is no doubt that any process which might improve the signal-to-noise ratio of the input data will increase the possibility of MED producing a useful output.
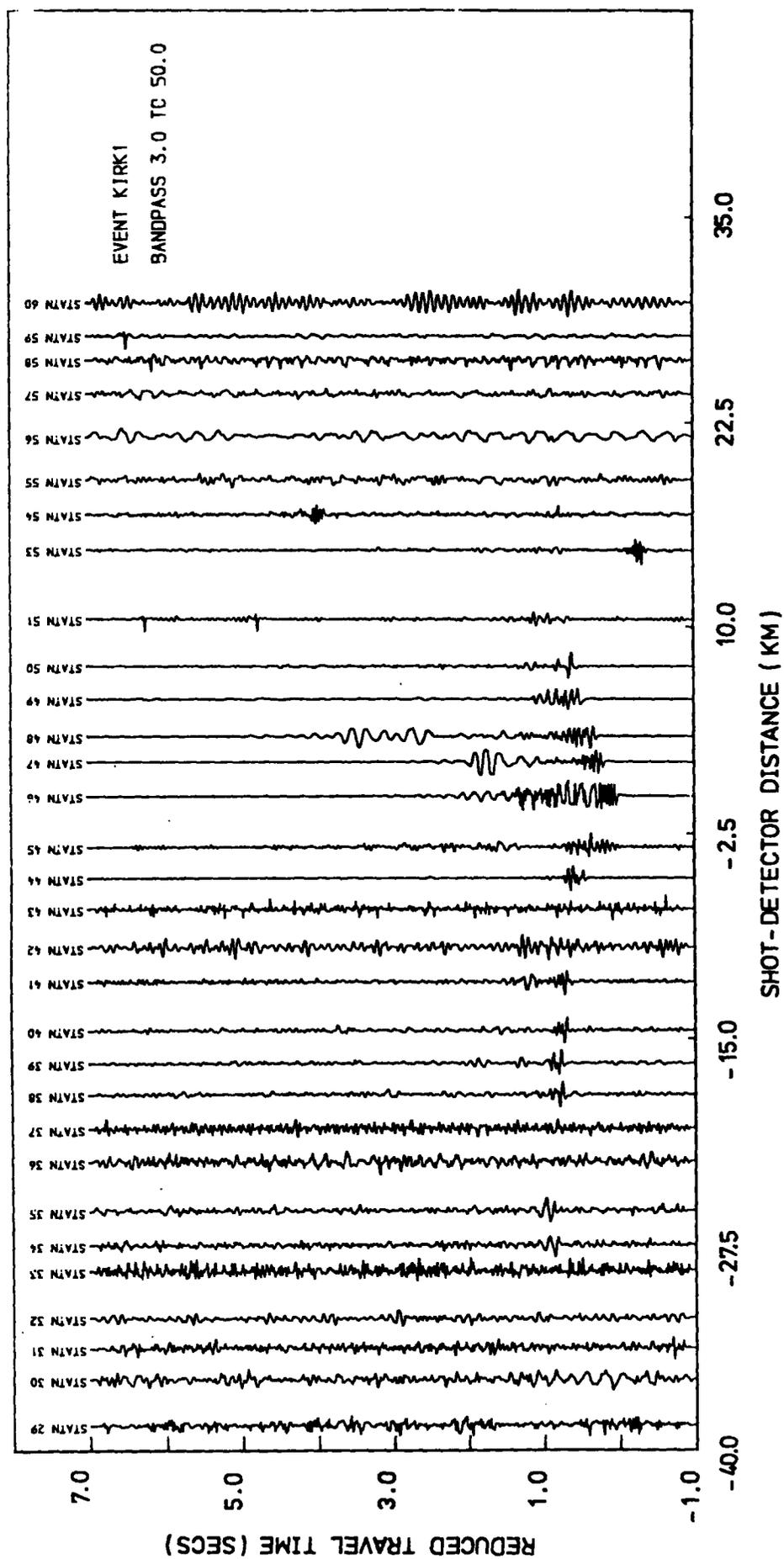
Fig. 5.3(a)

Noisy unprocessed data from CSSP – Kirkwhelpington shot.
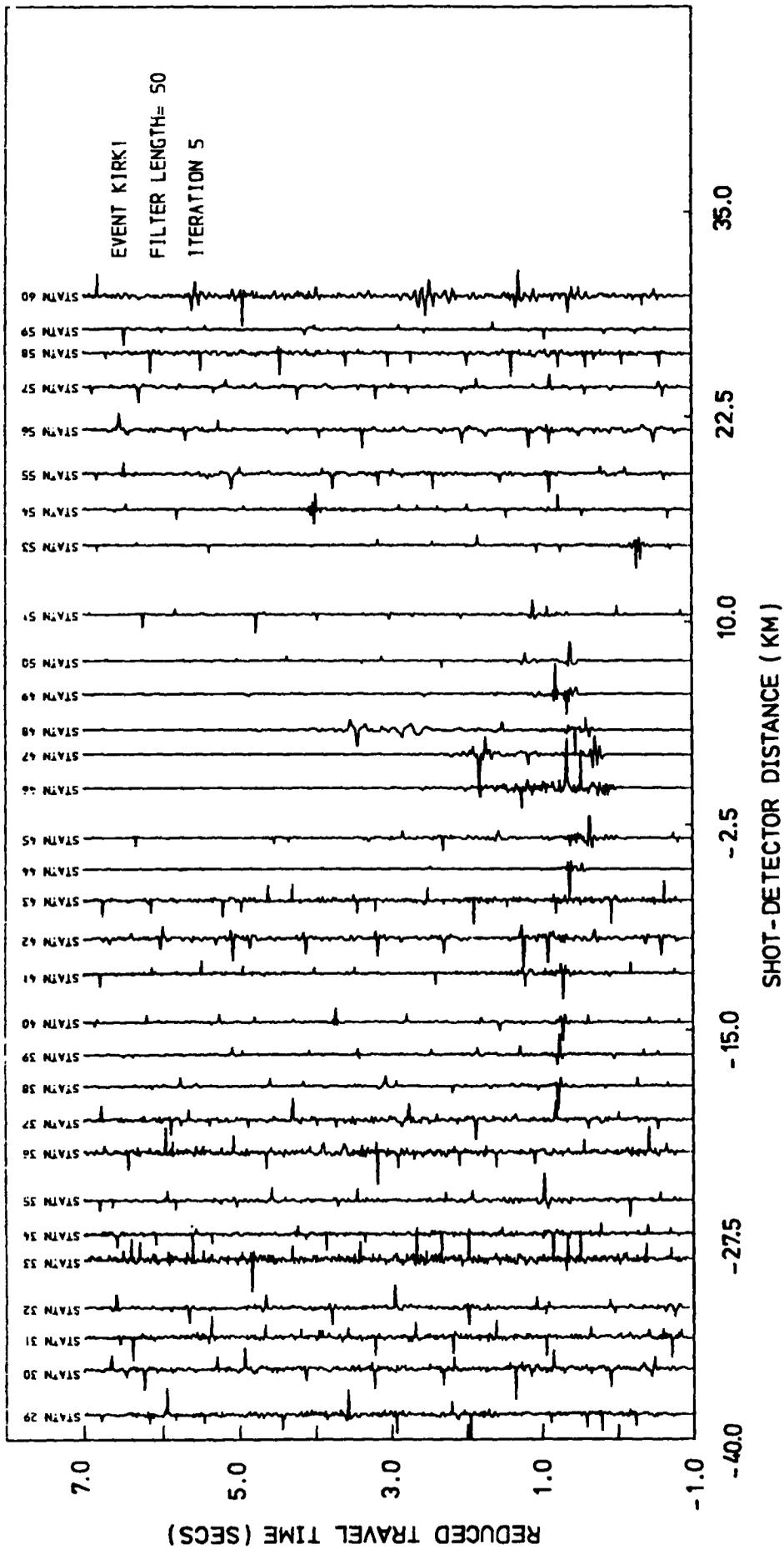
Fig. 5.3(b)

Effect of MED on noisy data.

Fig. 5.4(a)

Results of MED – interpretation.
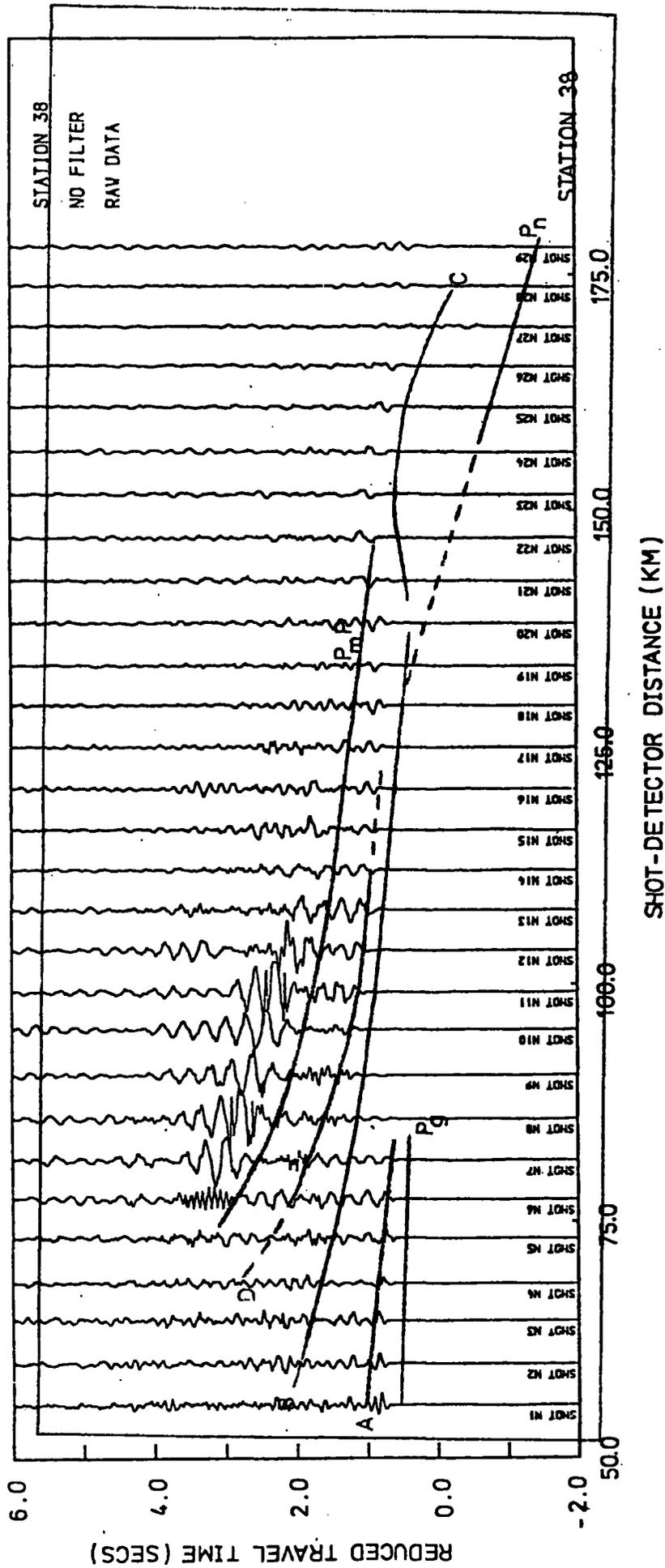
Fig. 5.4(b)

Unprocessed data from CSSP - interpretation.
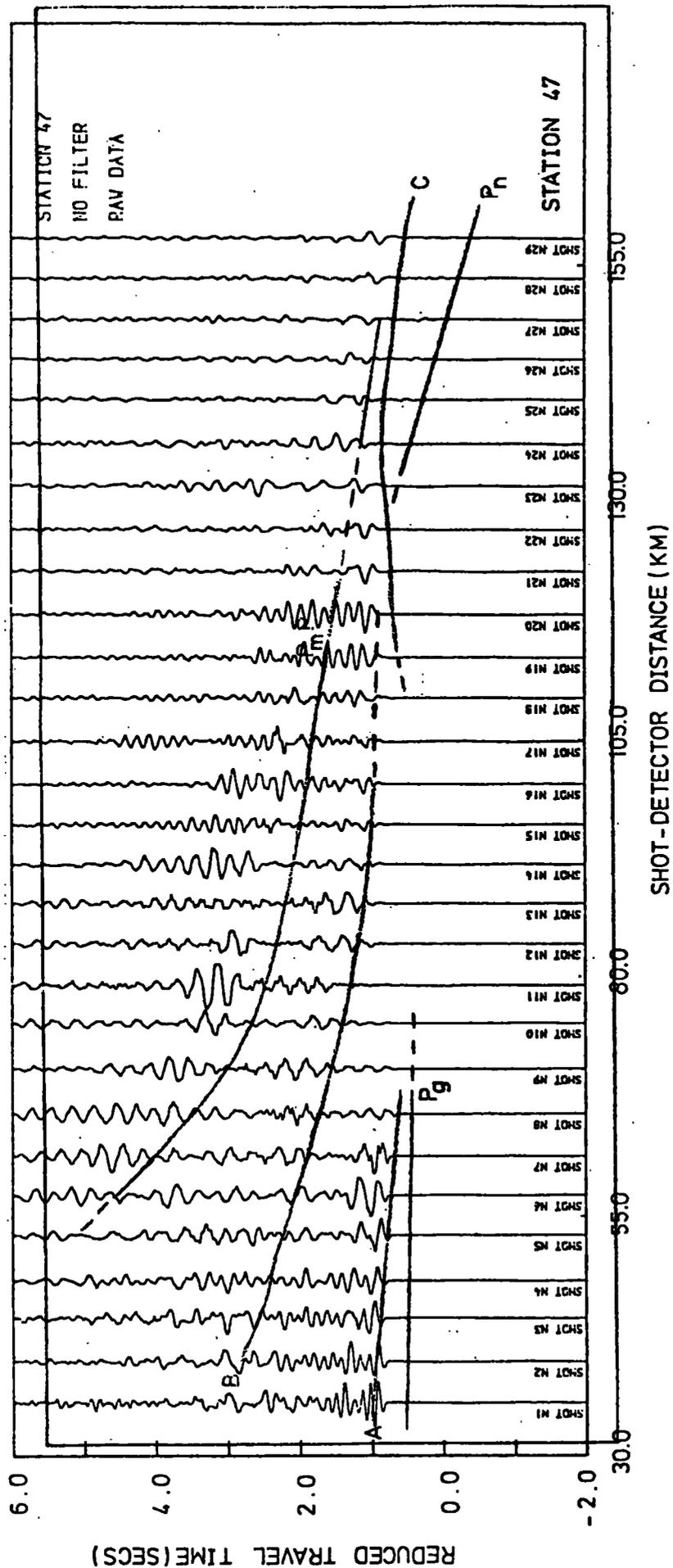
Fig. 5.4(c)

Unprocessed data from CSSP – interpretation.
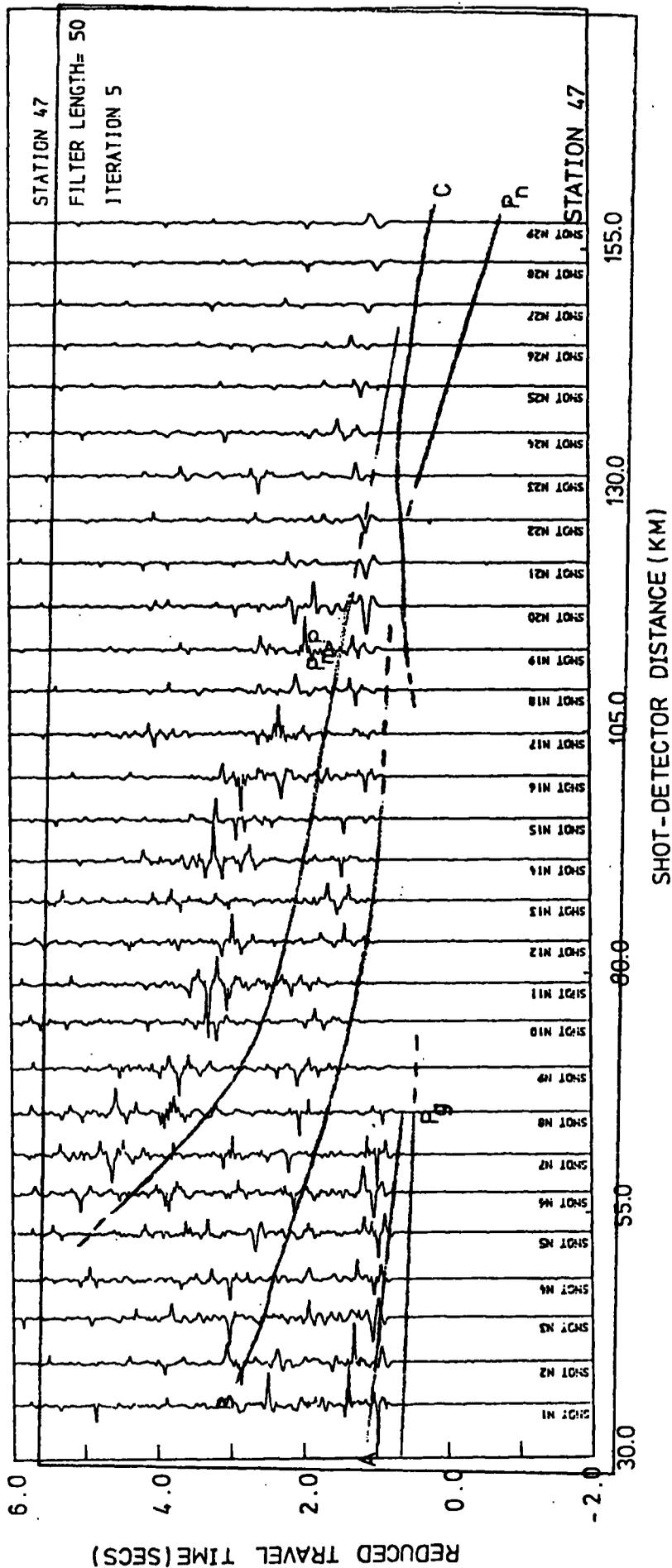
Fig. 5.4(d)

Results of MED – interpretation.

in the direction of increasing shot-detector distance.

In the processed sections arrival B is well defined and seems to be a mid-crustal reflection phase. Examination of the unprocessed sections shows that such an arrival is far from obvious there, but becomes discernible as the processed and unprocessed sections are compared. The picking of this arrival from the raw data, assuming it had been noticed in the first place, would have been very difficult using the unprocessed section alone.

Arrival C is obvious in all figures, but the arrivals are much sharper in the processed sections. It is suggested that this arrival is some lower crustal phase - possibly a diving wave.

Arrival D of Station 38 is drawn tentatively. It appears to be a reflection coming in between the mid-crustal reflection B and the Moho reflection P_mP. Again, the arrival can be seen in the unprocessed section when a comparison is made, although it is unlikely to have been noticed without the help of MED. This illustrates the value of MED as an aid to interpretation, in that it makes it easy to pick out possible arrivals. Comparison with the raw data can then be used to either confirm or reject the possibilities.

CHAPTER 6

DISCUSSION AND CONCLUSIONS

When picking arrivals manually from a raw section, arrivals other than the first are generally picked from the largest amplitude within the arrival wavelet. Appreciating that the MED process essentially chooses to spike up the largest amplitudes within a seismic trace, it is evident that the spikes are produced at precisely those locations where arrivals would most likely be picked prior to processing. By processing each trace independently, correlation between spikes from one trace to the next greatly increases the likelihood that such spikes correspond to genuine arrivals. In statistical terms simplification of the data in this way enhances the confidence with which the picking of arrivals may be carried out.

Experimentation has shown that a good choice of filter for the data used is one of length 50 with the '1' in the leading position, and that 4 or 5 iterations should be performed.

It is important to appreciate that such a choice is inevitably one of compromise, which is necessary if the process is to be applied to the complete section in a completely automatic way. Better results could be obtained, but at the expense of spending more time on

applying the process. Examination of each unprocessed trace would allow windows to be chosen in a more discerning manner, preventing truncation or splitting of arrivals at window junctions. For each window chosen, different filter parameters might be used, depending on the length of the arrival(s) in the window, the number of arrivals, and the distance or time travelled by the arrival(s).

One might be led to believe that, if it is possible to deduce all the above points from the unprocessed data, there is little point in applying the MED process. This is true, but the point is that it is often impossible for such deductions to be made. This is precisely why it was decided to try to develop MED as a completely automatic process, despite the fact that there are many obvious cases where the fixed window length of 100 points and fixed filter length of 50 are not ideal (e.g. wide angle reflection arrivals). The final results obtained are reasonable but by no means optimum ; it is a matter of opinion whether spending more time on each trace would yield sufficient improvement to merit the extra time taken.

From the results and discussions presented in this thesis it is possible to make a number of general conclusions relating to the practical application of MED to refraction data :

1) MED has the ability to resolve interfering arrivals in complex seismograms enabling arrivals to be picked with greater ease.

2) The process is useful only when applied to data in which the amplitude of signals exceeds the amplitude of noise.

3) The process may only be applied to short windows of data, otherwise arrivals will be lost. The windowing procedure developed here is likely to result in amplitude distortion.

4) Experience gained here suggests that the filter length may be extended to 2 or 3 times the length of the shortest signal resulting from interference of arrivals. This is true for five iterations, but loss of arrivals may result if the iterative process is continued further.

5) The longer the filter used, the cleaner the output and the sharper the spikes. Associated with this is the increased sensitivity to number of iterations performed and the increased danger of losing arrivals.

6) The initial filter should be chosen with the '1' in the leading position.

7) For input data consisting only of noise a spike will still be produced, assuming the noise is not white.

8) The process allows no control over which part of the arrival wavelet is spiked up, so that in general the results of MED can be used only as a qualitative aid to interpretation. In certain cases, however, (e.g. arrivals

A and C in Figures 5.4(b) and 5.4(d)) it is clear from comparison of the processed data with raw data that the same part of the wavelet is spiked up on each trace. In such cases, quantitative measurements may be made.

9) The spiking process destroys the character of the individual arrivals, so that in this respect information is lost.

10) The results of MED processing should not be interpreted alone, but should be used in conjunction with the unprocessed sections as an aid to picking arrivals.

Acknowledgements

References

AL-SADI, H. N., 1980, Seismic Exploration : Technique and Processing, Birkhauser, Stuttgart.

ANSTEY, N. A., 1981, Seismic Prospecting Instruments Vol.1 : Signal Characteristics and Instrument Specifications, Geoexploration Monographs.

ARFKEN, G., 1970, Mathematical Methods for Physicists, Academic Press.

BOTT, M. H. P. & LONG, R. E., Report on the Caledonian Suture Seismic Project.

CARROLL, J. B., 1953, An analytical solution for approximating simple structure in factor analysis, Psychometrika 18, 23-38.

CLAERBOUT, J. F., 1976, Fundamentals of Geophysical Data Processing, McGraw-Hill.

CLAERBOUT, J. F. & ROBINSON, E. A., 1963, The error in least-squares inverse filtering, Geophysics 29, No.1, 118-120.

FERGUSON, G. A., 1954, The concept of parsimony in factor analysis, Psychometrika 19, 281-290.

HAGEDOORN, J. G., 1959, The plus-minus method of interpreting seismic refraction sections, Geophysical Prospecting 7, 158-183.

KAISER, H. F., 1958, The Varimax criterion for analytic rotation in factor analysis, Psychometrika 23, 187-200.

KREYSZIG, E., 1979, Advanced Engineering Mathematics, Wiley.

OOE, M. & ULRYCH, T. J., 1979, Minimum entropy deconvolution with an exponential transformation, Geophysical Prospecting 27, 458-473.

PEACOCK, K. L. & TREITEL, S., 1969, Predictive deconvolution : theory and practice, Geophysics 34, 155-169.

ROBINSON, E. A. & TREITEL, S., 1967, Principles of digital Wiener filtering, Geophysical Prospecting 15, 311-333

STOFFA, P. L., BUHL, P. & BRYAN, G. M., 1974, The application of homomorphic deconvolution to shallow-water marine seismology, 1. Models, Geophysics 39, 401-416.

SUMMERS, T. P., 1982, A seismic study of crustal structure in the region of the Western Isles of Scotland, Ph.D. thesis.

WARREN, R. D., 1981, Signal processing techniques applied to seismic refraction data, M.Sc. thesis.

WIGGINS, R. A., 1978, Minimum entropy deconvolution, Geoexploration 16, 21-35.

APPENDIX

Computer Programs

```
C
C******************** PROGRAM TAPEREAD ********************
C
C     TAPEREAD PERFORMS BYTE-SWOPPING OPERATION ON INTEGER*2
C     VALUES READ FROM TAPE.
C     INPUT ON CHANNEL 7:DATA FILE CONTAINING 1 OR 2 COMPLETE
C     BLOCKS OF DATA.
C     UNFORMATTED OUTPUT ON CHANNEL 8: 800 DATA VALUES
C     STARTING AT SPECIFIED SAMPLE.
C
C     THE PROGRAM PROMPTS FOR NUMBER OF BLOCKS IN THE DATA
C     FILE, AND FOR DESIRED STARTING SAMPLE.
C
C
      INTEGER*2 X
      LOGICAL*1 IN(2048),TEMP
      EQUIVALENCE(IN(1),X(1))
C
C****** BLOCKS ARE 2048 BYTES LONG,2 BYTES PER INTEGER******
C
      DIMENSION X(2048)
      WRITE(6,200)
      READ(5,*) M
      WRITE(6,300)
      READ(5,*) ISTART
      READ(7)(X(I),I=1,1024)
      IF(M.EQ.1) GO TO 10
      READ(7) (X(I),I=1025,2048)
   10 CONTINUE
C
C****** BYTE SWOPPING COMMENCES ******
C
      DO 20 I=1,800
      X(I)=X(I+ISTART-1)
      TEMP=IN(2*(I-1)+1)
      IN(2*(I-1)+1)=IN(2*(I-1)+2)
      IN(2*(I-1)+2)=TEMP
   20 CONTINUE
C
C
      WRITE(8)(X(I),I=1,800)
  200 FORMAT(/'HOW MANY BLOCKS OF DATA IN INPUT FILE?')
  300 FORMAT('STARTING SAMPLE?'/)
      STOP
      END
```

```
C********************** PROGRAM READ **********************
C
C  PROGRAM READ CONVERTS BINARY INPUT TO READABLE REAL*8.
C
C     INPUT ON CHANNEL 7 : 800 BINARY DATA VALUES ON ONE LINE
C
C     OUTPUT ON CHANNEL 8 : 800 DATA VALUES IN F FORMAT
C
C*********************************************************
C
      REAL*8 X(800)
      READ(7) (X(I),I=1,800)
      WRITE(8,100) (X(I),I=1,800)
  100 FORMAT(F12.4)
      STOP
      END
```

```
C********************* PROGRAM AV *********************
C
C     AV CALCULATES AVERAGE VALUE FOR TRACE AND SUBTRACTS THIS
C     FROM EACH SAMPLE. AT THIS STAGE THE DATA VALUES ARE
C     CONVERTED FROM INTEGER*2 TO REAL*8.
C     INPUT ON CHANNEL 7: DATA FILE CONTAINING OUTPUT FROM
C     TAPEREAD.
C     DOUBLE PRECISION OUTPUT ON CHANNEL 8.
C
C
C     ARRAYS :      X = INPUT DATA VALUES
C               XMINAV = DATA VALUES MINUS AVERAGE VALUE
C
C     VARIABLES : XTOT = SUM OF INPUT DATA VALUES
C                  XAV = AVERAGE OF INPUT DATA VALUES
C
C**********************************************************
C
      INTEGER*2 X
      INTEGER XTOT
      REAL*8 XMINAV,XAV,AX
      DIMENSION X(800),XMINAV(800)
      XTOT=0
C
C****** READ DATA VALUES
C
      READ(7) (X(I),I=1,800)
C
C****** CALCULATE AVERAGE AND SUBTRACT FROM DATA VALUES:
C
      DO 10 I=1,800
      XTOT=XTOT+X(I)
  10  CONTINUE
      XAV=DFLOAT(XTOT)/800.0D0
      DO 20 I=1,800
      AX=X(I)
      XMINAV(I)=AX-XAV
  20  CONTINUE
C
C****** COSINE BELL TAPER APPLIED TO DATA ENDS
C
      PI=3.1415927
      DO 30 I=1,20
      XMINAV(I)=XMINAV(I)*(1+COS(PI*(I-20)/20))/2.0D0
  30  CONTINUE
      DO 40 I=781,800
      XMINAV(I)=XMINAV(I)*(1+COS(PI*(I-781)/20))/2.0D0
  40  CONTINUE
C
C****** WRITE OUTPUT TO CHANNEL 8
C
      WRITE(8) (XMINAV(I),I=1,800)
C
      STOP
      END
```

```
C******************** PROGRAM EQUALIZE ********************
C
C   FOR USE ON EVENT KIRK1.
C   PROGRAM EQUALIZES BY DIVIDING  EACH TRACE SAMPLE BY
C   THE MAXIMUM VALUE IN THAT TRACE. THE OUTPUT IS
C   COMPATIBLE WITH PROGRAMS MED AND PLOTTER.
C
C   INPUT ON CHANNEL 7 : DATA FILE WITH 7 LINES PER TRACE
C   THE CONTENTS OF LINES 1 TO 6 SHOULD BE THOSE REQUIRED BY
C   PROGRAM MED. LINE 7 IS THE INPUT DATA IN BINARY FORM.
C
C   OUTPUT ON CHANNEL 8 : AS INPUT BUT LINE 7 NOW CONTAINS
C   EQUALIZED DATA.
C
C      ARRAYS : X=INPUT DATA (REAL*8)
C   VARIABLES: XMAX=MAXIMUM VALUE IN TRACE
C               IF=NO.OF SAMPLES PER TRACE
C
C********************************************************************
C
      REAL*8 X(800),XMAX
      DIMENSION A(5),B(5),C(5),D(5)
C
C***** READ TITLES AND DATA
C
      DO 20 I=1,31
      READ(7,100) A
      READ(7,100) B
      READ(7,100) C
      READ(7,100) D
      READ(7,200) E
      READ(7,300) IF
      READ(7) (X(J),J=1,IF)
C
C***** EQUALIZATION BEGINS
C
      XMAX=0.0D0
      DO 30 L=1,IF
      IF(DABS(X(L)).LT.XMAX) GO TO 40
      XMAX=DABS(X(L))
   40 CONTINUE
   30 CONTINUE
      DO 10 K=1,IF
   10 X(K)=X(K)/XMAX
C
C***** WRITE TITLES AND EQUALIZED DATA
C
      WRITE(8,100) A
      WRITE(8,100) B
      WRITE(8,100) C
      WRITE(8,100) D
      WRITE(8,200) E
      WRITE(8,300) IF
      WRITE(8) (X(J),J=1,IF)
   20 CONTINUE
C
  100 FORMAT(5A4)
  200 FORMAT(F7.3)
  300 FORMAT(I6)
C
      STOP
      END
```

```
C********************** PROGRAM BANBOX **********************
C
C                  MODIFIED FROM R. WARREN (1981)
C
C  PROGRAM BANBOX PRODUCES TIME DOMAIN BANDPASS FILTER. A BOX-CAR
C  FUNCTION WITH TAPERED ENDS IS APPLIED TO YIELD A FINITE
C  REALISATION OF THE FILTER. THE PROGRAM PROMPTS FOR LENGTH OF
C  FILTER REQUIRED (MUST BE ODD), THE SAMPLING INTERVAL OF THE
C  TIME SERIES TO WHICH THE FILTER IS TO BE APPLIED, AND THE LOWER
C  AND UPPER PASS FREQUENCIES. ALL REPLIES IN FREE FORMAT.
C  THE FILTER IS APPLIED USING PROGRAM BANKONV. THE OUTPUT FROM
C  APPLICATION OF THE FILTER MUST BE SHIFTED TO LEFT BY (LF-1)/2
C  POINTS.
C
C     OUTPUT ON CHANNEL 8 : THREE LINES
C                   LINE 1 : TITLE
C                        2 : FILTER LENGTH (I6)
C                        3 : FILTER COEFFTS. (BINARY)
C
C     ARRAYS : FILT = FILTER COEFFTS.
C              W = MULTIPLICATION FACTOR FOR TAPERING
C
C  VARIABLES : LF = FILTER LENGTH (INTEGER UP TO 100)
C              DT = SAMPLING INTERVAL OF TIME SERIES (REAL)
C              FL = LOWER FREQ. OF BANDPASS (REAL)
C              FH = UPPER FREQ. OF BANDPASS (REAL)
C
C**************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION FILT(100),W(50)
C
C***** PROMPT FOR VARIABLES
C
      WRITE(6,100)
      READ(5,*) LF
      WRITE(6,200)
      READ(5,*) DT
      WRITE(6,300)
      READ(5,*) FL
      WRITE(6,400)
      READ(5,*) FH
C
C***** CALCULATE FILTER COEFFTS.
C
      M=(LF+1)/2
      FM=M
      IF((FH-FL)-1.0/(DT*FM))1,2,2
    1 FC=(FH+FL)/2.0D0
      WL=6.2831853*(FC*DT-0.5/FM)
      WH=6.2831853*(FC*DT+0.5/FM)
      GO TO 3
    2 WL=FL*DT*6.2831853
      WH=FH*DT*6.2831853
    3 FILT(1)=WH-WL
      DO 4 I=2,M
      FI=I-1
    4 FILT(I)=(DSIN(WH*FI)-DSIN(WL*FI))/FI
C
C***** COSINE BELL TAPER APPLIED TO ENDS OF FILTER
C
      M10=M-9
      DO 6 I=M10,M
      W(I)=(1+COS(3.14159265*(I-M10)/10))/2.0D0
    6 FILT(I)=FILT(I)*W(I)
C
C***** CALCULATION OF FILTER COEFFTS. CONTINUES
C
      DO 7 I=1,M
    7 FILT(I)=FILT(I)/3.14159265
      DO 8 I=1,M
    8 FILT(LF-I+1)=FILT(M-I+1)
      MM=M+M
      DO 9 I=1,M
      J=MM-I
    9 FILT(I)=FILT(J)
C
C***** WRITE TITLES AND FILTER COEFFTS. TO CHANNEL 8
C
      WRITE(8,500) FL,FH
      WRITE(8,600) LF
      WRITE(8) (FILT(I),I=1,LF)
C
  100 FORMAT(//'SPECIFY LENGTH OF FILTER(MUST BE ODD):')
  200 FORMAT(//'SPECIFY SAMPLING INTERVAL OF TIME SERIES:')
  300 FORMAT(//'SPECIFY LOWER FREQ. OF BANDPASS:')
  400 FORMAT(//'SPECIFY UPPER FREQ. OF BANDPASS:')
  500 FORMAT('BANDPASS ',F3.1,' TO ',F4.1,' HZ')
  600 FORMAT(I6)
C
      STOP
      END
```

```
C******************* PROGRAM BANKONV ********************
C
C  BANKONV CONVOLVES BANDPASS FILTER FROM BANBOX WITH TRACES
C  CONTAINING 800 SAMPLES (BINARY). THE PROGRAM PROMPTS FOR
C  THE NUMBER OF TRACES ON WHICH TO OPERATE. THE OUTPUT IS
C  SHIFTED BACK BY (LF-1)/2 POINTS AS REQUIRED.
C
C     INPUT ON CHANNEL 7 : DATA FILE CONTAINING UNPROCESSED
C     TRACES AS FOR INPUT TO MED.
C
C     INPUT ON CHANNEL 10 : THE BANDPASS FILTER - OUTPUT FROM
C     BANBOX.
C
C     OUTPUT ON CHANNEL 8 : BANDPASSED RESULTS (BINARY) WITH
C     TITLES COMPATIBLE WITH INPUT REQUIREMENTS OF MED AND
C     PLOTTER.
C
C     ARRAYS : X = INPUT DATA (BINARY; OPERATED AS REAL*8)
C              F = FILTER COEFFTS. (BINARY; REAL*8)
C           CONV = RESULTS OF CONVOLUTION (REAL*8)
C  A,B,C,D,FILT = TITLES
C
C  VARIABLES : LF = LENGTH OF FILTER
C               M = NO. OF TRACES
C               N = NO. OF SAMPLES PER TRACE
C
C*********************************************************
C
      REAL*8 X(800),F(100),CONV(900)
      DIMENSION A(5),B(5),C(5),D(5),FILT(6)
C
C***** READ FILTER
C
      READ(10,400) FILT
      READ(10,100) LF
      READ(10) (F(J),J=1,LF)
C
C***** PROMPT FOR NO. OF TRACES
C
      WRITE(6,200)
      READ(5,*) M
C
C***** MAIN LOOP BEGINS
C
      DO 70 IN=1,M
      READ(7,400) A
      READ(7,400) B
      READ(7,400) C
      READ(7,400) D
      READ(7,600) E
      READ(7,100) N
      READ(7) (X(I),I=1,800)
C
C***** WRITE TITLES TO OUTPUT FILE
C
      WRITE(8,400) A
      WRITE(8,400) B
      WRITE(8,500) FILT
      WRITE(8,600) E
      K=N+LF-1
      K1=N+(LF-1)/2
      WRITE(8,100) N
C
C***** CONVOLUTION BEGINS
C
      DO 10 I=1,K
      CONV(I)=0.0
      DO 20 J=1,I
      L=I-J+1
      IF(L.GT.N) GO TO 20
      IF(J.GT.LF) GO TO 10
      CONV(I)=CONV(I)+F(J)*X(L)
 20   CONTINUE
 10   CONTINUE
C
C***** SHIFT OUTPUT TO LEFT BY (LF-1)/2
C
      DO 60 I=1,K1
      CONV(I)=CONV(I+(LF-1)/2)
 60   CONTINUE
C
C***** WRITE RESULTS TO OUTPUT FILE
C
      WRITE(8) (CONV(I),I=1,N)
      WRITE(6,700)IN
 70   CONTINUE
C
100   FORMAT(I6)
 200  FORMAT('HOW MANY TRACES THIS TIME,O GREAT ONE:'//)
400   FORMAT(6A4)
500   FORMAT(6A4/)
 600  FORMAT(F7.3)
 700  FORMAT('TRACE',I3,'********')
C
      STOP
      END
```

```
C
C********************* PROGRAM MED **********************
C
C                   GARVEY M.ANDERSON (1983)
C
C                      M.Sc. PROJECT
C
C************** MINIMUM ENTROPY DECONVOLUTION ************
C
C     PROGRM MED APPLIES MINIMUM ENTROPY DECONVOLUTION TO A
C     VARIABLE NUMBER OF TRACES,WITH 800 SAMPLES PER TRACE.
C     WINDOWS ARE APPLIED AUTOMATICALLY TO EACH TRACE,AND A
C     FILTER DEVELOPED AND APPLIED FOR EACH WINDOW. THE WINDOW
C     LENGTH IS 120 INCLUDING A TAPER OF 10 AT EACH END.IGNORING
C     THE TAPERED ENDS,THE WINDOW GEOMETRY IS 0-100,50-150,100-
C     200 ETC.,SO THAT THERE IS COMPLETE OVERLAP OF WINDOWS.
C
C
C     THE PROGRAM PROMPTS FOR : FILTER LENGTH (UP TO 50 POINTS)
C                             : POSITION OF 1 IN INITIAL FILTER
C                             : NO. OF TRACES IN INPUT FILE
C                             : NO. OF ITERATIONS
C        ALL REPLIES IN FREE FORMAT
C
C     INPUT ON CHANNEL 7 : INPUT FILE CONSISTS OF FILE FOR EACH
C     TRACE CONCATENATED INTO ONE FILE.EACH TRACE FILE MUST
C     CONTAIN SEVEN LINES :
C              1         TITLE FOR COMPLETE SEISMOGRAM (5A4)
C              2         TITLE FOR TRACE (5A4)
C              3         BLANK LINE
C              4         BLANK LINE
C              5         SHOT-DETECTOR DISTANCE (F7.3)
C              6         NO. OF DATA SAMPLES PER TRACE (I6)
C              7         UNFORMATTED DATA VALUES FOR TRACE (BINARY)
C
C     INPUT ON CHANNEL 11 : AN INITIAL FILTER OTHER THAN
C     (0,0,...1,...0,0) MAY BE DEFINED IF DESIRED.THE PROGRAM
C     PROMPTS FOR VERIFICATION OF WHETHER OR NOT SUCH A FILTER HAS
C     BEEN DEFINED. LINE 1 OF INPUT FILE SHOULD SPECIFY THE LENGTH
C     OF THE FILTER (I6),FOLLOWED BY THE FILTER COEFFICIENTS WITH
C     ONE COEFFICIENT PER LINE (F7.3).
C
C     OUTPUT ON CHANNEL 8 : THE OUTPUT FILE CONTAINS THE RESULTS
C     OF MED IN UNFORMATTED (BINARY) FORM. EACH TRACE TAKES
C     UP SEVEN LINES :
C              1         TITLE FOR COMPLETE SEISMOGRAM (5A4)
C              2         TITLE FOR TRACE (5A4)
C              3         TITLE SPECIFYING LENGTH OF FILTER USED (5A4)
C              4         TITLE SPECIFYING NO. OF ITERATIONS
C                        PERFORMED (5A4)
C              5         SHOT-DETECTOR DISTANCE (F7.3)
C              6         NO. OF DATA SAMPLES PER TRACE (I6)
C              7         DATA VALUES OF DECONVOLVED TRACE (BINARY)
C
C     ARRAYS : X1 = INPUT DATA
C               X = DATA WITHIN WINDOW
C               F = INITIAL FILTER COEFFICIENTS
C               Y = RESULTS OF FILTER APPLIED TO DATA
C              Z1 = OUTPUT FROM FIRST WINDOW SET
C               Z = OUTPUT FROM SECOND WINDOW SET
C               G = RIGHT-HAND SIDE OF MED NORMAL EQUATIONS
C            FILT = FILTER COEFFTS.
C             STN = TITLE FOR SEISMOGRAM
C           TRACE = TITLE FOR EACH TRACE
C          FILTER = FILTER LENGTH TITLE
C          ITERTN = ITERATION NO. TITLE
C
C     VARIABLES : M = LENGTH OF FILTER (INTEGER)
C                 J = POSITION OF '1' IN INITIAL FILTER
C            ANSWER = NO. OF ITERATIONS TO BE PERFORMED (INTEGER)
C              NOTR = NO. OF TRACES (INTEGER)
C              DIST = SHOT-DETECTOR DISTANCE (F7.3)
C                 N = NO. OF SAMPLES PER TRACE
C            ITAPER = LENGTH OF WINDOW TAPER
C               NOW = NO. OF WINDOWS PER TRACE
C               NWP = COUNTER FOR STARTING SAMPLE OF EACH WINDOW
C                NW = LENGTH OF WINDOW EXCLUDING TAPERED ENDS
C              ICOF = LENGTH OF WINDOW INCLUDING TAPERED ENDS
C                 K = LENGTH OF CONVOLVED DATA
C
C********************************************************************
C
      INTEGER ANSWER
      REAL*8 X(200), F(50), Y(300), Z(800), Z1(800)
      REAL*8 X1(800), G(50), FILT(50)
      DIMENSION STN(5), TRACE(5), FILTER(5), ITERTN(5)
```

```fortran
C
C***** FILTER INITIATION
C
      WRITE (6,340)
      READ (5,*) IANS
      IF (IANS .LE. 1) GO TO 10
      WRITE (6,350)
      READ (5,*) IANS1
      IF (IANS1 .GT. 1) GO TO 30
   10 CONTINUE
      READ (11,420) M
      DO 20 I = 1, M
      READ (11,440) F(I)
   20 CONTINUE
      IF (IANS1 .EQ. 1) GO TO 70
      IF (IANS .EQ. 1) GO TO 70
   30 WRITE (6,310)
      READ (5,*) M
      WRITE (6,320)
      READ (5,*) J
      L = J - 1
      IF (L .LT. 1) GO TO 50
      DO 40 I = 1, L
      F(I) = 0.0D0
   40 CONTINUE
   50 F(J) = 1.0D0
      L1 = J + 1
      DO 60 I = L1, M
      F(I) = 0.0D0
   60 CONTINUE
   70 CONTINUE
C
C***** PROMPT FOR NO. OF ITERATIONS
C
      WRITE (6,330)
      READ (5,*) ANSWER
C
C***** READ TITLES
C
      WRITE (6,370)
      READ (5,*) NOTR
      WRITE (6,380) M
      WRITE (6,390) ANSWER
C
C***** LOOP FOR EACH TRACE BEGINS
C
      DO 300 INDEX = 1, NOTR
      WRITE (6,360) INDEX
      READ (7,430) STN
      READ (7,430) TRACE
      READ (7,430) FILTER
      READ (7,430) ITERTN
      READ (7,440) DIST
C
C***** READ INPUT DATA LENGTH N,AND DATA X(I),FROM CHANNEL 7.
C
      READ (7,420) N
      READ (7) (X1(I),I=1,N)
C
C***** START WINDOWING PROCEDURE
C
      DO 280 IODEX = 1, 2
      ITAPER = 10
      NOW = 8
      NOW = NOW + (IODEX - 1)
      NWP = 0
      DO 260 IW = 1, NOW
      IF (IODEX .GT. 1) GO TO 80
      NW = 100
      GO TO 90
   80 NW = 100
      IF (IW .EQ. 1) NW = 50
      IF (IW .EQ. 9) NW = 50
   90 CONTINUE
C
C***** FOR FIRST WINDOW TAPER IS ADDED ONLY TO WINDOW END
C
      IF (IW .GT. 1) GO TO 110
  100 CONTINUE
      ICO = 0
      ICO3 = 0
      ICOF = NW + ITAPER
      GO TO 130
  110 CONTINUE
      ICO = NWP - ITAPER
      ICO3 = ICO + ITAPER
      ICOF = NW + (2*ITAPER)
C
C***** FOR LAST WINDOW TAPER ADDED ONLY TO START OF WINDOW
C
      IF (IW .LT. NOW) GO TO 120
      ICOF = NW + ITAPER
  120 CONTINUE
  130 CONTINUE
```

```
C
C***** DATA READ IN FOR WINDOW
C
      DO 140 I = 1, ICOF
      X(I) = X1(I + ICO)
  140 CONTINUE
      PI = 3.1415927
      IF (IW .GT. 1) GO TO 160
C
C***** COSINE BELL TAPER APPLIED TO DATA FOR FIRST WINDOW
C
      IUT = NW
      IUT1 = NW + ITAPER
      DO 150 IT = IUT, IUT1
      X(IT) = X(IT) * (1 + COS(PI*(IT - IUT)/ITAPER)) / 2.0D0
  150 CONTINUE
      GO TO 190
  160 CONTINUE
      IF (IW .LT. NOW) GO TO 180
C
C***** TAPER APPLIED TO DATA FOR LAST WINDOW
C
      ILT = ITAPER
      DO 170 I = 1, ILT
      X(I) = X(I) * (1 + COS(PI*(I - ITAPER)/ITAPER)) / 2.0D0
  170 CONTINUE
      GO TO 190
  180 CONTINUE
C
C***** TAPER APPLIED TO DATA FOR INTERMEDIATE WINDOWS
C
      ILT = ITAPER
      IUT = NW + ITAPER
      CALL WINDOW(X, ICOF, ILT, IUT, ITAPER)
  190 CONTINUE
C
C********** MED BEGINS **********
C
      J1 = M ** 2
C
C***** CALCULATE AUTOCORRELATION FUNCTION COEFFTS.
C
      CALL AUTCOR(X, ICOF, M)
C
C***** CALCULATE AUTOCORRELATION MATRIX
C
      CALL MATRIX(M)
C
C***** CALCULATE Y BY CONVOLVING INITIAL FILTER WITH X
C
      CALL CONVO(X, F, ICOF, M, Y, K)
C
C***** ITERATIVE PROCESS BEGINS
C
      ITER = 0
  200 CONTINUE
C
C***** CALCULATE V AND U
C
      CALL VU(Y, K, U, V)
C
C***** CALCULATE CROSSCORRELATION MATRIX
C
      CALL CRSCOR(X, Y, ICOF, K, M, G)
C
C***** CALCULATE FILTER COEFFTS.
C
      CALL MWIENR(J1, M, M, V, U, G, FILT)
C
      ITER = ITER + 1
C
C***** APPLY FILTER TO DATA
C
      CALL CONVO(X, FILT, ICOF, M, Y, K)
C
      IF (ANSWER .GT. ITER) GO TO 200
C
C********** MED ENDS **********
C
C
C***** WRITE TITLES TO OUTPUT FILE,CHANNEL 8
C
      IF (IODEX .GT. 1) GO TO 210
      IF (IW .GT. 1) GO TO 210
      WRITE (8,430) STN
      WRITE (8,430) TRACE
      WRITE (8,450) M
      WRITE (8,460) ITER
      WRITE (8,440) DIST
      WRITE (8,420) N
  210 CONTINUE
      IF (IW .GT. 1) GO TO 230
```

```
C
C***** COPY FILTERED DATA TO ARRAY Z
C
      DO 220 I = 1, NW
      Z(ICO3 + I) = Y(I + J - 1)
  220 CONTINUE
      GO TO 250
  230 CONTINUE
      DO 240 I = 1, NW
      Z(ICO3 + I) = Y(I + ITAPER + J - 1)
  240 CONTINUE
  250 CONTINUE
      NWP = NW + NWP
  260 CONTINUE
      IF (IODEX .EQ. 2) GO TO 280
C
C***** COPY RESULTS OF FIRST WINDOW SET TO ARRAY Z1
C
      DO 270 I = 1, 800
  270 Z1(I) = Z(I)
  280 CONTINUE
C
C***** OUTPUT FROM TWO WINDOW SETS COMBINED
C
      DO 290 I = 1, 800
  290 Z(I) = (Z1(I) + Z(I)) / 2.0D0
      WRITE (8) (Z(I),I=1,800)
  300 CONTINUE
      STOP
  310 FORMAT ('FILTER LENGTH?')
  320 FORMAT ('POSITION OF 1?')
  330 FORMAT (/'HOW MANY ITERATIONS?')
  340 FORMAT (/'HAVE YOU DEFINED FILTER ON UNIT 11?(1=YES,2=NO)')
  350 FORMAT (/'DO YOU WANT TO?(1=YES,2=NO)')
  360 FORMAT ('TRACE ', I4, ' *************'//)
  370 FORMAT (/'HOW MANY TRACES?')
  380 FORMAT (//'FILTER LENGTH= ', I3)
  390 FORMAT (//'NO. OF ITERATIONS= ', I3)
  400 FORMAT (F8.2)
  410 FORMAT ('SAMPLE LENGTH=', I3)
  420 FORMAT (I6)
  430 FORMAT (5A4)
  440 FORMAT (F7.3)
  450 FORMAT ('FILTER LENGTH=', I3)
  460 FORMAT ('ITERATION ', I1)
      END
C
C*****MAIN PROGRAM ENDS
C
C
      SUBROUTINE WINDOW(X, ICOF, ILT, IUT, ITAPER)
C
C***** APPLIES COSINE BELL TAPER TO ENDS OF WINDOW DATA
C
C        X = INPUT DATA (REAL*8)
C     ICOF = LENGTH OF WINDOW
C      ILT = SAMPLE NO. UP TO WHICH LEFT-HAND TAPER APPLIED
C      IUT = SAMPLE NO. FROM WHICH RIGHT-HAND TAPER APPLIED
C   ITAPER = LENGTH OF TAPER
C
C            DOUBLE PRECISION USED THROUGHOUT
C
      REAL*8 X(ICOF), R, RT, RU
C
      PI = 3.1415927
      RT = DFLOAT(ITAPER)
C
      DO 10 I = 1, ILT
      R = DFLOAT(I)
      X(I) = X(I) * (1 + DCOS(PI*(R - RT)/RT)) / 2.0D0
   10 CONTINUE
C
      RU = DFLOAT(IUT)
      DO 20 I = IUT, ICOF
      R = DFLOAT(I)
      X(I) = X(I) * (1 + DCOS(PI*(R - RU)/RT)) / 2.0D0
   20 CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE CONVO(X, F, N, M, CONV, K)
C
C***** CONVOLVES TWO TIME SERIES F AND X
C
C        X = INPUT DATA - UP TO 200 VALUES
C        F = FILTER COEFFTS. (REAL*8)
C        N = NO. OF DATA VALUES.
C        M = NO. OF FILTER COEFFTS.
C     CONV = RESULTS OF CONVOLUTION (REAL*8)
C        K = NO. OF OUTPUT VALUES
C
C            DOUBLE PRECISION THROUGHOUT
C
C*****************************************
C
      REAL*8 X(200), F(M), CONV(300)
C
      K = N + M - 1
      DO 20 I = 1, K
      CONV(I) = 0.0D0
      DO 10 J = 1, I
      L = I - J + 1
      IF (L .GT. N) GO TO 10
      IF (J .GT. M) GO TO 20
      CONV(I) = CONV(I) + F(J) * X(L)
   10 CONTINUE
   20 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE VU(Y, K, U, V)
C
C***** CALCULATES VARIMAX NORM V,AND SUM OF SQUARES OF DATA U
C
C        Y = RESULTS OF PREVIOUS ITERATION (REAL*8)
C        K = NO. OF DATA VALUES
C     WSUM = SUM OF (DATA VALUES TO POWER 4)
C
C            DOUBLE PRECISION THROUGHOUT
C
C********************************************************************
C
      REAL*8 Y(K), YFOUR(300), YTWO(300)
      REAL*8 WSUM, U, V, W
C
      DO 10 I = 1, K
      YFOUR(I) = Y(I) ** 4
   10 YTWO(I) = Y(I) ** 2
C
      U = 0.0D0
      WSUM = 0.0D0
      DO 20 I = 1, K
      W = YFOUR(I)
      WSUM = WSUM + W
      U = YTWO(I) + U
   20 CONTINUE
C
      V = WSUM / (U**2)
C
      RETURN
      END
C
C
      SUBROUTINE AUTCOR(X, N, M)
C
C***** CALCULATES AUTOCORRELATION FUNCTION COEFFTS.
C
C        X = INPUT DATA (REAL*8)
C        N = NO. OF DATA VALUES.
C        M = NO. OF AUTOCORRELATION COEFFTS.
C      AUT = AUTOCORRELATION COEFFTS. FOR X
C
C            DOUBLE PRECISION THROUGHOUT
C
C******************************************************
C
      COMMON /AUTCR/ AUT
      REAL*8 X(200), AUT(50)
C
      DO 30 I = 1, M
      NR = N + 1 - I
      AUT(I) = 0.0D0
      IF (I .GT. N) GO TO 20
      DO 10 J = 1, NR
   10 AUT(I) = AUT(I) + X(J) * X(J + I - 1)
   20 CONTINUE
   30 CONTINUE
C
      RETURN
      END
```

A/2

```
      SUBROUTINE CRSCOR(X, Y, N, K, M, G)
C
C***** CALCULATES CROSS-CORRELATION FUNCTION COEFFTS.
C
C         X = INPUT DATA (REAL*8)
C         Y = RESULTS OF PREVIOUS ITERATION
C         N = NO. OF DATA VALUES X
C         K = NO. OF DATA VALUES Y
C         M = NO. OF CROSS-CORRELATION COEFFTS.
C         G = CROSS-CORRELATION COEFFTS.
C
C             DOUBLE PRECISION THROUGHOUT
C
C************************************************************
C
      REAL*8 X(200), YCUBE(200), G(M), Y(K)
C
      DO 10 I = 1, K
   10 YCUBE(I) = Y(I) ** 3
C
      DO 30 I = 1, M
      G(I) = 0.0D0
      DO 20 J = 1, N
   20 G(I) = G(I) + X(J) * YCUBE(J + I - 1)
   30 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE MATRIX(M)
C
C***** ARRANGES AUTOCORRELATION COEFFTS. INTO A TWO-DIMENSIONAL
C         ARRAY, AUTMX. AUTMX(I,J) IS Jth ELEMENT OF THE Ith ROW OF
C         AUTOCORRELATION MATRIX. OUTPUTS THE ELEMENTS OF AUTMX IN
C         SEQUENCE REQUIRED BY SUBROUTINE MWIENR.
C
C         M = NO. OF AUTOCORRELATION COEFFTS.
C       ACF = AUTOCORRELATION COEFFTS.
C     AUTMX = AUTOCORRELATION MATRIX
C       AUT = AUTOCORRELATION COEFFTS. ARRANGED IN ORDER FOR MWIENR
C
C             DOUBLE PRECISION THROUGHOUT
C
C************************************************************
C
      COMMON /AUTCR/ ACF
      COMMON /MATRX/ AUT
      REAL*8 ACF(50), AUTMX(50,50), AUT(2500)
      INTEGER P
C
C***** AUTMX SET UP
C
      DO 30 P = 1, M
      DO 20 J = 1, M
      I = J + P - 1
      IF (I .GT. M) GO TO 10
      AUTMX(I,J) = ACF(P)
   10 CONTINUE
   20 CONTINUE
   30 CONTINUE
C
      DO 60 P = 1, M
      DO 50 I = 1, M
      J = I + P - 1
      IF (J .GT. M) GO TO 40
      AUTMX(I,J) = ACF(P)
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
C
C***** AUT READ FROM AUTMX
C
      K = 0
      DO 100 I = 1, M
      DO 90 J = 1, M
      K = K + 1
      IF (J .EQ. I) GO TO 70
      AUT(K) = AUTMX(I,J)
      GO TO 80
   70 CONTINUE
C
C***** DIAGONAL TERMS OF MATRIX INCREASED BY 0.5% TO REDUCE
C         ILL-CONDITIONING
C
      AUT(K) = AUTMX(I,J) + (5.0D0*AUTMX(I,J)/1000.0D0)
   80 CONTINUE
   90 CONTINUE
  100 CONTINUE
C
      RETURN
      END
```

```fortran
C
C
      SUBROUTINE MWIENR(J1, M, LR, V, U, G, FILT)
C
C***** SOLVES NORMAL EQUATIONS USING LEVINSON RECURSION
C
C         J1 = SQUARE OF FILTER LENGTH
C          M = FILTER LENGTH
C          V = VARIMAX NORM
C          U = SUM OF SQUARES OF RESULTS OF PREVIOUS ITERATION
C          G = CROSS-CORRELATION COEFFTS.
C       FILT = FILTER COEFFTS.
C        AUT = AUTOCORRELATION COEFFTS.
C      AUTVU = AUT WEIGHTED BY V/U
C       CROS = G WEIGHTED BY 1/(U SQUARED)
C
C              DOUBLE PRECISION THROUGHOUT
C
C*****************************************************************
C
      COMMON /MATRX/ AUT
      REAL*8 AUT(2500), CROS(50), PE(50), FILT(M), G(M)
      REAL*8 AUTVU(2500), V, U, R1, R2, D, F1, Q, RLP, AL, HOLD, FL
C
      DO 10 I = 1, J1
      AUTVU(I) = AUT(I) * (V/U)
   10 CONTINUE
C
      DO 20 I = 1, M
      CROS(I) = G(I) / (U**2)
   20 CONTINUE
C
      ISP15 = 0
      R1 = AUTVU(1)
      R2 = AUTVU(2)
      D = R2
      PE(1) = 1.0D0
      F1 = CROS(1) / R1
      FILT(1) = F1
      Q = F1 * R2
      IF (LR .LE. 1) GO TO 90
      DO 80 L = 2, LR
      RLP = MINO(1 + L,LR)
      IF (R1 .GT. 0.0D0) GO TO 30
      ISP15 = L - 1
C
      GO TO 90
   30 AL = -D / R1
      PE(L) = AL
      FILT(L) = R1
      R1 = R1 + AL * D
      D = AUTVU(RLP) + AL * R2
      L2 = L / 2
      IF (L .LE. 3) GO TO 50
      L2 = L / 2
C
      DO 40 J = 2, L2
      K = L - J + 1
      HOLD = PE(J)
      PE(J) = PE(J) + AL * PE(K)
      D = D + PE(J) * AUTVU(K + 1)
      PE(K) = PE(K) + AL * HOLD
   40 D = D + PE(K) * AUTVU(J + 1)
C
   50 IF (L2*2 .EQ. L) GO TO 60
      J = L2 + 1
      PE(J) = PE(J) + AL * PE(J)
      D = D + PE(J) * AUTVU(J + 1)
   60 CONTINUE
C
      FL = (CROS(L) - Q) / R1
      FILT(L) = FL
      Q = FL * R2
C
      L1 = L - 1
      DO 70 J = 1, L1
      K = L - J + 1
      FILT(J) = FILT(J) + FL * PE(K)
   70 Q = Q + FILT(J) * AUTVU(K + 1)
   80 CONTINUE
   90 RETURN
      END
```

```
C******************* PROGRAM PLOTTER *******************
C
C     PROGRAM PLOTTER PRODUCES REDUCED TRAVEL-TIME PLOTS. IT
C     PROMPTS FOR NUMBER OF TRACES TO PLOT. PLOT PRODUCED IS
C     A4 SIZE. PLOTTER ACCEPTS OUTPUT FROM PROGRAM MED.
C
C        INPUT ON CHANNEL 7 : DATA FILE WITH 7 LINES PER TRACE
C
C             LINE 1 : TITLE FOR PLOT(4A4)
C                  2 : TITLE FOR TRACE(4A4)
C                  3 : FILTER SPECIFICATION(5A4)
C                  4 : ITERATION SPECIFICATION(4A4)
C                  5 : SHOT-DETECTOR DISTANCE(F7.3)
C                  6 : NO. OF SAMPLES IN TRACE(I6)
C                  7 : INPUT DATA (BINARY)
C
C        OUTPUT ON CHANNEL 9
C
C     THE PROGRAM CALLS *GHOST SUBROUTINES
C
C        ARRAYS : Z=INPUT DATA (REAL*8)
C                 X=DATA SCALED FOR REDUCED PLOT
C                 Y=SAMPLE COUNTER
C
C     VARIABLES : M1=NO. OF TRACES
C                 N=NO. OF SAMPLES PER TRACE
C****************************************************************
C
      REAL*8 Z(900)
      DIMENSION STN(4), TRACE(4), FILT(5), ITER(4)
      DIMENSION X(800), Y(800)
C
C***** PROMPT FOR NO. OF TRACES
C
      WRITE (6,60)
      READ (5,*) M1
C
C***** INITIATE PLOTTING PROCEDURE
C
      CALL PAPER(1)
      CALL PSPACE(0.1, 1.0, 0.2, 0.55)
      CALL CSPACE(0.05, 1.0, 0.0, 0.60)
      CALL MAP(0.0, 30.0, 0.0, 800.0)
      CALL AXES
C
C***** READ INPUT
C
      DO 20 J = 1, M1
      READ (7,50) STN
      READ (7,50) TRACE
      READ (7,50) FILT
      READ (7,50) ITER
      READ (7,40) DIST
      READ (7,30) N
      READ (7) (Z(I),I=1,N)
C
C***** CONVERT DATA TO SINGLE PRECISION AND APPLY SCALING
C      FACTOR FOR REDUCED PLOT
C
      DO 10 I = 1, N
      X(I) = Z(I) / 3000.0D0+(DIST - 30.0) / 5.0D0
      Y(I) = FLOAT(I)
   10 CONTINUE
C
C***** WRITE TITLE FOR TRACE AND PLOT TRACE
C
      CALL BLKPEN
      CALL CTRMAG(5)
      XP = X(1) - 0.2
      CALL CTRORI(1.0)
      CALL PLOTCS(XP, 10.0, TRACE, 8)
      CALL CTRORI(0.0)
      CALL PTPLOT(X, Y, 1, N, -2)
   20 CONTINUE
C
C***** WRITE PLOT TITLES
C
      CALL CTRMAG(8)
      CALL PLOTCS(26.0, 750.0, STN, 10)
      CALL PLOTCS(26.0, 700.0, FILT, 17)
      CALL PLOTCS(26.0, 650.0, ITER, 11)
      CALL BORDER
      CALL GREND
C
   30 FORMAT (I6)
   40 FORMAT (F7.3)
   50 FORMAT (5A4)
   60 FORMAT (//'HOW MANY TRACES HAVE YOU GOT?')
C
      STOP
      END
```

```
C********************* PROGRAM SYNMED *********************
C
C
C
C                  GARVEY M. ANDERSON (1983)
C
C                       M.Sc. PROJECT
C
C
C**************** MINIMUM ENTROPY DECONVOLUTION ****************
C
C
C     PROGRAM SYNMED PERFORMS MINIMUM ENTROPY DECONVOLUTION ON
C     SYNTHETIC DATA. WINDOWS ARE NOT APPLIED.
C
C     THE PROGRAM PROMPTS FOR : FILTER LENGTH (UP TO 100 POINTS)
C                             : POSITION OF 1 IN INITIAL FILTER
C                             : NO. OF ITERATIONS
C
C     INPUT ON CHANNEL 7 : SYNTHETIC DATA X(I) (UP TO 100 VALUES),
C     IN F10.4 FORMAT. EACH DATA VALUE ON A NEW LINE. FIRST FOUR
C     LINES OF INPUT FILE ARE :
C               LINE 1          TITLE FOR SEISMOGRAM (5A4)
C                    2          TITLE FOR FILTER LENGTH (5A4), OF IF
C                               DATA IS UNPROCESSED PUT 'NO FILTER'.
C                    3          TITLE FOR NO. OF ITERATIONS (5A4), OR IF
C                               UNPROCESSED PUT 'RAW DATA'.
C                    4          NO. OF DATA VALUES (I6)
C
C     INPUT ON CHANNEL 11 : AS FOR PROGRAM MED :- ALLOWS INITIAL
C     FILTER OTHER THAN (0,0,...1,....0) TO BE SPECIFIED.
C
C     OUTPUT ON CHANNEL 8 : RESULTS OF EACH ITERATION IN F10.4,
C     WITH ONE VALUE PER LINE. FIRST FOUR LINES OF OUTPUT FROM
C     EACH ITERATION ARE :
C               LINE 1          TITLE FOR SEISMOGRAM (5A4)
C                    2          TITLE SPECIFYING FILTER LENGTH USED
C                    3          TITLE SPECIFYING ITERATION NO.
C                    4          NO. OF DATA VALUES
C
C     OUTPUT COMPATIBLE WITH PROGRAM SYNPLOTTER.
C
C     OUTPUT ON CHANNEL 10 : THE FILTER COEFFTS. PRODUCED AT EACH
C     STAGE IN THE ITERATIVE PROCESS, WITH TITLES.
C
C     ARRAYS : X = INPUT DATA
C              F = INITIAL FILTER COEFFTS.
C           FILT = MED FILTER COEFFTS.
C              G = RIGHT-HAND SIDE OF NORMAL EQUATIONS
C              Y = RESULTS OF MED
C
C     VARIABLES : N = NO. OF DATA VALUES IN SYNTHETIC INPUT
C                 M = NO. OF FILTER COEFFTS.
C                 J = POSITION OF '1' IN INITIAL FILTER
C                 K = LENGTH OF DATA AFTER CONVOLUTION WITH F
C
C     DOUBLE PRECISION USED THROUGHOUT.
C
C     IN SUBROUTINE MATRIX 0.5% SHOULD NOT BE ADDED TO THE
C     DIAGONAL TERMS OF THE AUTOCORRELATION MATRIX FOR
C     SYNTHETIC DATA.
C
C************************************************************************
C
C
      INTEGER ANSWER
      REAL*8 X(100), F(100), FILT(100), G(100), Y(200)
      REAL*8 U, V
      DIMENSION TITLE(4), FILTER(5), ITERTN(4), X1(100), Y1(100)
C
C***** READ TITLES
C
      READ (7,230) TITLE
      READ (7,230) FILTER
      READ (7,230) ITERTN
C
C***** READ INPUT DATA LENGTH N, AND DATA X(I), FROM CHANNEL 7.
C
      READ (7,190) N
      WRITE (8,230) TITLE
      WRITE (8,230) FILTER
      WRITE (8,230) ITERTN
      WRITE (8,190) N
      DO 10 I = 1, N
      READ (7,200) X1(I)
      WRITE (8,200) X1(I)
      X(I) = DBLE(X1(I))
   10 CONTINUE
```

```
C----- FILTER INITIATION
C
      WRITE (6,150)
      READ (5,*) IANS
      IF (IANS .LE. 1) GO TO 20
      WRITE (6,160)
      READ (5,*) IANS1
      IF (IANS1 .GT. 1) GO TO 40
   20 CONTINUE
C
      READ (11,190) M
      DO 30 I = 1, M
      READ (11,200) F(I)
   30 CONTINUE
C
      IF (IANS1 .EQ. 1) GO TO 80
      IF (IANS .EQ. 1) GO TO 80
   40 WRITE (6,130)
C
      READ (5,*) M
      WRITE (6,140)
      READ (5,*) J
      L = J - 1
      IF (L .LT. 1) GO TO 60
      DO 50 I = 1, L
      F(I) = 0.0D0
   50 CONTINUE
   60 F(J) = 1.0D0
      L1 = J + 1
      DO 70 I = L1, M
      F(I) = 0.0D0
   70 CONTINUE
C
   80 CONTINUE
C
C
C********** MED BEGINS **********
C
C
C***** CALCULATE AUTOCORRELATION FUNCTION COEFFTS.
C
      J1 = M ** 2
      CALL AUTCOR(X, N, M)
C
C***** CALCULATE AUTOCORRELATION MATRIX
C
      CALL MATRIX(M)
C
C***** CALCULATE Y BY CONVOLVING INITIAL FILTER WITH X
C
      CALL CONVO(X, F, N, M, Y, K)
C
C***** PROMPT FOR NO. OF ITERATIONS
C
      ITER = 0
      WRITE (6,170)
      READ (5,*) ANSWER
   90 CONTINUE
C
C***** CALCULATE V AND U
C
      CALL VU(Y, K, U, V)
C
C***** CALCULATE CROSSCORRELATION MATRIX
C
      CALL CRSCOR(X, Y, N, K, M, G)
C
C***** CALCULATE FILTER COEFFTS.
C
      CALL MWIENR(J1, M, M, V, U, G, FILT)
C
C***** OUTPUT FILTER COEFFTS. ON CHANNEL 10
C
      ITER = ITER + 1
      WRITE (10,220) M
      WRITE (10,230) TITLE
      WRITE (10,210) ITER
      WRITE (10,180) N
      WRITE (10,190) M
      DO 100 I = 1, M
      WRITE (10,200) FILT(I)
  100 CONTINUE
```

```
C
C***** APPLY FILTER TO DATA
C
      CALL CONVO(X, FILT, N, M, Y, K)
C
C***** WRITE TITLES AND RESULTS TO OUTPUT FILE,CHANNEL 8
C
      WRITE (8,230) TITLE
      WRITE (8,220) M
      WRITE (8,210) ITER
      WRITE (8,190) K
      DO 110 I = 1, K
      Y1(I) = SNGL(Y(I))
      WRITE (8,200) Y1(I)
  110 CONTINUE
      IF (ANSWER .LE. ITER) GO TO 120
      GO TO 90
C
  120 STOP
  130 FORMAT ('FILTER LENGTH?')
  140 FORMAT ('POSITION OF 1?')
  150 FORMAT (/'HAVE YOU DEFINED FILTER ON UNIT 11?(1=YES,2=NO)')
  160 FORMAT (/'DO YOU WANT TO?(1=YES,2=NO)')
  170 FORMAT (/'HOW MANY ITERATIONS?')
  180 FORMAT ('SAMPLE LENGTH=', I3)
  190 FORMAT (I6)
  200 FORMAT (F10.4)
  210 FORMAT ('ITERATION ', I2)
  220 FORMAT ('FILTER LENGTH=', I3)
  230 FORMAT (5A4)
      END




C******************** PROGRAM SYNPLOTTER ********************
C
C   SYNPLOTTER PRODUCES PLOTS OF RESULTS FROM SYNMED, SHOWING THE
C   RESULTS OF EACH STEP IN THE ITERATIVE PROCESS. INPUT DATA
C   LENGTH IS RESTRICTED TO 200 POINTS.
C
C     INPUT ON CHANNEL 7 : OUTPUT FROM SYNMED
C
C     OUTPUT ON CHANNEL 9
C
C   THE PROGRAM PROMPTS FOR THE TOTAL NO. OF ITERATIONS TO PLOT,
C   UP TO A MAXIMUM OF 6. IT FURTHER PROMPTS FOR THE POSITION OF
C   THE '1' IN THE INITIAL FILTER USED BY SYNMED. A NUMBER OF
C   PARAMETERS RELATING TO THE PLOT ARE ALSO REQUESTED.
C
C   THE PROGRAM CALLS *GHOST SUBROUTINES.
C
C     ARRAYS : X = INPUT DATA (UP TO 200 F10.4 VALUES)
C              Y = SAMPLE COUNTER
C          YTICK = COUNTER FOR TICK SPACING ALONG TRACE
C
C   VARIABLES : M1 = NO. OF TRACES
C             IPOS1 = POSITION OF '1' IN INITIAL FILTER
C         XMIN,XMAX = LIMITS OF MATHEMATICAL SPACE
C                 N = NO. OF SAMPLES PER TRACE
C              YMAX = LENGTH OF PLOT IN Y-DIRN. (UP TO 200)
C            SPACER = TICK SPACING ALONG TRACE
C              GRAT = GRATICULE SPACING FOR PLOT
C
C*************************************************************
C
      DIMENSION TITLE(4), ITER(4), FILT(5)
      DIMENSION X(200), Y(200), YTICK(120), XTICK(120)
C
C***** PROMPT FOR PLOT PARAMETERS
C
      WRITE (6,120)
      READ (5,*) M1
      WRITE (6,130)
      READ (5,90) IPOS1
      WRITE (6,80)
      READ (5,*) XMIN, XMAX
      WRITE (6,70)
      READ (5,*) YMAX
      WRITE (6,110)
      READ (5,*) SPACER
      WRITE (6,100)
      READ (5,*) GRAT
      IYMAX = IFIX(YMAX)
      ISPACE = IFIX(SPACER)
      M = IYMAX / ISPACE
C
      DO 10 I = 1, M
      XTICK(I) = 0.0
      YTICK(I) = SPACER * I
   10 CONTINUE
C
      CALL CSPACE(0.0, 0.5, 0.0, 0.9)
```

```
C
C***** MAIN LOOP BEGINS...TRACES PLOTTED ONE AT A TIME
C
      DO 40 J = 1, M1
      READ (7,90) TITLE
      READ (7,90) FILT
      READ (7,90) ITER
      READ (7,50) N
      Y(1) = 1.0
      DO 20 I = 1, N
      Y(I + 1) = Y(I) + 1.0
      READ (7,60) X(I)
   20 CONTINUE
C
      IF (J .GT. 1) GO TO 30
      CALL PAPER(1)
      CALL BLKPEN
C
C***** WRITE MAIN TITLE ON PLOT
C
      CALL CTRMAG(5)
      CALL PLACE(62, 16)
      CALL TYPECS(TITLE, 10)
   30 CONTINUE
C
C***** PLOT EACH ITERATION WITH TITLES
C
      J1 = J - 1
      CALL PSPACE(0.1, 0.32, 0.8 - (0.05*J1), 0.8 - (0.05*J))
      CALL MAP(0.0, YMAX, XMIN, XMAX)
      CALL PTPLOT(Y, X, 1, N, -2)
      CALL CTRORI(1.0)
      CALL CTRMAG(3)
      CALL PTPLOT(YTICK, XTICK, 1, M, 44)
      CALL GRATSI(GRAT, XMAX)
      IF (J .LE. 1) GO TO 40
      CALL CTRMAG(5)
      CALL CTRORI(0.0)
      CALL PLACE(62, 15 + 7*(J - 1))
      CALL TYPECS(ITER, 12)
C
   40 CONTINUE
C
C***** WRITE TITLES AT BOTTOM OF PLOT
C
      CALL PLACE(55, 15 + 7*M1)
      CALL TYPECS(FILT, 17)
      CALL CRLNFD
      CALL SPACE(54)
      CALL TYPECS('POSITION OF 1=', 14)
      CALL TYPECS(IPOS1, 2)
C
      CALL PSPACE(0.1, 0.32, 0.8, 0.8 - (M1*0.05))
      CALL SCALSI(25.0, 500.0*M1)
      CALL BORDER
      CALL GREND
C
   50 FORMAT (I5)
   60 FORMAT (F10.4)
   70 FORMAT ('DESIRED PLOT LENGTH?(REAL):')
   80 FORMAT ('SPECIFY XMIN,XMAX OF MATHEMATICAL SPACE.(REAL):')
   90 FORMAT (5A4)
  100 FORMAT (/'SPECIFY GRATICULE SPACING ALONG CENTRE LINE:')
  110 FORMAT (/'SPECIFY TICK SPACINGS FOR CENTRE LINE')
  120 FORMAT (//'HOW MANY TRACES HAVE YOU GOT?')
  130 FORMAT (//'SPECIFY POSITION OF 1 IN FILTER USED:')
C
      STOP
      END
```