

Durham E-Theses

Cross-hole seismic reflection surveying in coal measures

Findlay, Michael John

How to cite:

Findlay, Michael John (1991) *Cross-hole seismic reflection surveying in coal measures*, Durham theses, Durham University. Available at Durham E-Theses Online: <http://etheses.dur.ac.uk/6163/>

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a [link](#) is made to the metadata record in Durham E-Theses
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the [full Durham E-Theses policy](#) for further details.

Cross-hole Seismic Reflection Surveying in Coal Measures

by

Michael John Findlay

**A thesis submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy**

**Department of Geological Sciences
The University of Durham
1991**

The copyright of this thesis rests with the author.
No quotation from it should be published without
his prior written consent and information derived
from it should be acknowledged.



18 AUG 1992

The research in this thesis has been carried out at the Department of Geological Sciences of the University of Durham between October 1987 and September 1990. It is the original work of the Author unless stated otherwise. None of this work has been submitted for any other degree.

The copyright of this thesis rests with the Author. No quotation from it may be published without his prior written consent, and information derived from it should be acknowledged.

Acknowledgements

There are a number of people whom I would like to mention for their contributions either directly or indirectly towards the work of this thesis.

Above all, I would like to thank Dr Neil Goulty for his supervision and encouragement throughout my research work.

I would also like to thank Ed Kragh for his comradeship and useful suggestions.

My thanks go to the other members of the borehole seismic research group who helped in fieldwork, i.e. Stefan Thatcher and Miles Leggett and also to the many British Coal Opencast geologists and drilling crews who were most helpful in providing access to boreholes and borehole logs.

I am grateful to the Department of Education for Northern Ireland for funding me.

Finally, I would like to thank my patient wife, Anita, for her encouragement over the past three years and especially so whilst this thesis was being written.

Abstract

This thesis is concerned with the development of the cross-hole seismic reflection surveying method with particular application to the shallow Coal Measures strata found in opencast coal mining prospects in the U.K.

A field acquisition technique developed for shallow boreholes utilising explosive sources and hydrophone receivers is described. Data have been acquired from several test sites in northern England.

Data-processing techniques including wavefield separation and waveshaping deconvolution have been developed for cross-hole data and the theories behind these techniques are discussed.

Methods of imaging cross-hole reflection data including the 'VSP-CDP' transformation and Generalised Kirchhoff migration are applied to computer-generated synthetic data and to real data in order to yield a depth section of the seismic reflectivity between the boreholes.

Finally, the data-processing and imaging techniques developed are applied to real data acquired at British Coal Opencast exploration sites in northern England between 1987 and 1990.

Table of Contents

Chapter I

Introduction	1
1.1 Synopsis.....	1
1.2 Opencast coal mining in the U.K.....	1
1.3 Opencast coal exploration	2
1.4 Seismic exploration for coal.....	4

Chapter II

The cross-hole seismic reflection method	6
2.1 Borehole seismic techniques	6
2.2 Cross-hole seismic techniques.....	7
2.3 Cross-hole seismic data acquisition	8
2.3.1 The source	8
2.3.2 The receivers	9
2.3.3 Recording equipment.....	10
2.3.4 Field equipment set-up	11
2.4 Data acquisition problems	12
2.4.1 Timing errors.....	12
2.4.2 Noise	12
2.4.3 Borehole blockages.....	13
2.5 Uphole shots and verticality.....	14
2.5.1 Uphole velocity estimates.....	14
2.5.2 Borehole verticality	15

Chapter III

Data processing	16
3.1 Introduction	16
3.2 Editing	16
3.2.1 Shot-matching deconvolution	17
3.2.2 First break estimation and direct arrival suppression	18

3.3 Wavefield separation.....	19
3.3.1 The Nyquist frequencies and aliasing	21
3.3.2 Two-dimensional filter application.....	21
3.3.3 Filter design	22
3.3.4 Tube waves and spatial aliasing.....	23
3.3.5 Examples of wavefield separation in the f-k domain	25
3.4 Waveshaping deconvolution	27
3.5 Velocity field estimation	28
3.6 The reflection point mapping scheme or VSP-CDP transform.....	29
3.7 Velocity analysis using the VSP-CDP transform.....	32
 Chapter IV	
Migration of cross-hole reflection data.....	37
4.1 Introduction	37
4.2 Finite-difference migration	38
4.3 Implementation of the finite-difference migration algorithm	39
4.3.1 The finite-difference approximation	40
4.3.2 Boundary conditions.....	42
4.3.3 Grid dispersion.....	44
4.3.4 Modelling seismic data with EXTRAP.....	45
4.3.5 Migrating seismic data with EXTRAPREV.....	46
4.4 Kirchhoff migration.....	46
4.4.1 Diffraction stack migration.....	46
4.4.2 The Kirchhoff operator.....	47
4.4.3 The Generalised Kirchhoff operator.....	49
4.5 Comparison of finite-difference and Kirchhoff migration.....	50
4.5.1 Point diffractor model	50
4.5.2 Horizontal reflecting interface model	51
 Chapter V	
Results from cross-hole surveys in Coal Measures strata.....	54
5.1 Introduction	54
5.2 Tinsley Park.....	54

5.3 Lowther South, West Yorkshire.....	57
5.3.1 Survey B, illustrating the combination of up- and down- going sections	58
5.3.2 Survey C; between boreholes II and III.....	60
5.3.3 Survey D; wider-spaced cross-hole reflection survey	61
5.3.4 Survey E; cross-hole survey through major fault zone.....	63
5.4 Lostrigg, Cumbria.....	66
5.4.1 Survey F; old mineworkings	66
 Chapter VI	
A comparison of different migration operators for cross-hole seismic data	68
6.1 Introduction	68
6.2 Migration operators	68
6.2.1 Suitability of impulse response functions for cross-hole data ..	70
 Chapter VII	
Conclusions and suggestions for future work.....	72
 References.....	75
 Appendix A Computer software.....	
Appendix A.1 Data processing software	II
Appendix A.2 The reflection point loci program REFLOC	III
Appendix A.3 The VSP-CDP transform program XHRMAP	IV
Appendix A.4 The raytracing program TRACER	V
Appendix A.5 The raytracing subroutine RAYTRA	VI
Appendix A.6 The finite-difference migration program EXTRAPREV	VII
Appendix A.7 The forward modelling program EXTRAP.....	VIII
Appendix A.8 The migration program KIRCHMIG.....	IX

Chapter I

Introduction

1.1 Synopsis

The purpose of this research work has been to develop the cross-hole reflection surveying method, and to consider its feasibility as an exploration technique for opencast coal prospects. The remainder of this chapter is concerned with opencast coal exploration in the U.K. in order to put the development of the cross-hole method into context. Chapter II deals with the acquisition of cross-hole seismic survey data and the problems associated with it. Chapter III is an overview of the data processing which is required for these surveys and chapter IV concentrates on the migration methods which may be used to image the data. Some trial surveys which were conducted at opencast prospects in the north of England are discussed in chapter V, and a comparison of different cross-hole migration operators is included in chapter VI. Conclusions of this work are drawn in chapter VII.

1.2 Opencast coal mining in the U.K.

In the modern era, opencast coal mining in the U.K. began in 1942, when it was introduced as a wartime measure to extract shallow coal reserves from the top few metres of the ground. Modern mines are now much deeper; a typical working site may be 100m deep. Pits are much smaller than in other parts of the world, simply because of the high population density in the U.K. Site boundaries are restricted by



population centres and the communication networks between them as well as by natural boundaries.

Current estimates of shallow coal reserves in the U.K. suggest that there are around 300 million tonnes of coal which could be extracted by opencast mining. Although the total annual output (~15 Mtonnes) from opencast mines is less than 20% of the total U.K. production, opencast coal is more economical to produce and is often of a higher quality than deep-mined coal. Deep-mined coal often has opencast coal blended with it to improve its calorific value and bring it up to the required standard for a marketable product.

The responsibility for opencast coal developments lies with British Coal Opencast (BCO), which is a subsidiary of the British Coal Corporation. BCO provides a detailed specification of a prospect to civil engineering companies who tender for the contract to mine the site. BCO also uses the specification in applying for planning permission. As part of their contract, the civil engineering companies must restore the pit area when the mining operation has been concluded.

1.3 Opencast coal exploration

Initial site assessments are based upon the currently available geological knowledge of the area. Additional information is also provided by the plans of old mineworkings (although these are usually too unreliable to help in site reserve estimates) as well as the data obtained from any nearby opencast sites. Once there is sufficient evidence to show that a proposed site may be profitable, the exploration stage begins.

BCO's present exploration strategy involves the drilling of a dense grid of shallow (typically with total depths less than 100m) boreholes. The boreholes have 14cm diameter steel casing from the surface down to rock head (typically 5-10 metres of casing), but below this depth they are left uncased. This drilling programme is a massive exploration effort with a total metreage of around 750,000m drilled each year. These are drilled using an air-flush system in which compressed air forces rock chippings from the drill bit to the surface, where they are logged by the driller. All the boreholes are subsequently geophysically logged by using natural gamma and density tools. The density log gives excellent responses to coal seams, whilst the natural gamma log identifies dirt bands and is used in correlations between boreholes. Approximately 20% of the boreholes are also cored. An ideal drilling strategy is initially to drill holes on a grid pattern with 120m spacing. This would then be reduced to 60m and then to 30m, or even less, in areas of particular interest where faulting or old mineworkings are believed to exist. However, permit difficulties and time limitations often dictate that the initial reconnaissance stage is omitted, and in some parts of the country only a single phase of drilling on, say, 50m centres is carried out. The interpreted logs from the boreholes are used to determine the overburden lithologies as well as the location and thicknesses of coal seams and the presence of fault zones which may lead to problems in the excavation of coal.

It is very important that BCO can provide as accurate an estimate as possible of the rippable coal reserves, since it will incur contractual penalties if the excavating civil engineering contractor finds less readily extractable coal in the site than was predicted, and also because any profits from excess coal reserves will chiefly benefit the contractor. The site must also be shown to be of primary economic importance to the appropriate planning authorities.

To date the drilling programme has proven to be an effective exploration strategy, but information coverage is disrupted where physical obstructions such as buildings and rivers are located. These gaps in the data might be filled by geophysical techniques which can provide information on the strata between the boreholes. It is barely possible to detect faults with throws of 2m or less from borehole levels, even if the boreholes were drilled only 2m apart. There is, therefore, great potential for a geophysical technique which can detect such small faults in critical areas such as site boundaries. In addition, suitable geophysical surveys combined with a 60m borehole spacing, might allow old workings and faults to be determined more accurately than would be possible by drilling boreholes with 30m separations. This could potentially eliminate the need for ~75% of the boreholes in these areas.

The pillar and stall type of old mineworkings are particularly prevalent in areas suitable for opencast mining because, naturally enough, the earliest old mineworkings were frequently located near outcrops. As many as five or six sets of old mineworkings may be present in different seams on the same site. Indeed, the greater part of the coal in an economically viable prospect may be in the form of pillars in the old mineworkings which were left behind to prevent collapse.

1.4 Seismic exploration for coal

Surface seismic reflection surveys have been used as an exploration tool for deep mines for almost twenty years. In these surveys target depths are of the order of hundreds of metres, and poor imaging is obtained in the upper 100m of the sections (see e.g. Ziolkowski, 1979). The surface reflection method has been applied

specifically to opencast targets by Brabham (1986), but the results were poor with long wavelength (~20m) reflections being recorded, the higher frequencies having been absorbed by the weathered surface layers. A great problem for shallow seismic reflection surveys is interference from refracted waves and ground roll; this is very dependent on the shallow geology (Bredewout and Goulty, 1986).

In-seam seismic methods have also been used successfully in deep mines (Buchanan, 1983; Jackson, 1985). Deep mine hole-to-hole seam wave transmission experiments were carried out by Jackson et al. (1989), who observed arrivals with the dispersive character which one would expect for a low-velocity SH channel wave between holes where the seam was continuous. When old mineworkings or faulting exist between the boreholes, no such wave should be detected. The method has been applied to shallow Coal Measures strata (Goulty et al., 1990), but without success because of the strong attenuation of high-frequency shear waves in shallow strata. Vertical seismic profiling (VSP) methods have been applied to deeper coal seams by Suprajitno and Greenhalgh (1985) and Jackson et al. (1989) and to shallow Coal Measures strata by Kragh (1990).

It is hoped that cross-hole seismic reflection surveys may be used in conjunction with VSPs and borehole logs to provide a flexible, high-resolution tool for opencast coal exploration. In particular, they could fill in information where boreholes cannot be drilled and accurately determine the location of small faults and old mineworkings.

Chapter II

The cross-hole seismic reflection method

2.1 Borehole seismic techniques

The check shot survey is a long-established borehole seismic technique in the oil exploration industry which is used to calibrate velocity measurements obtained from borehole sonic logs. A natural progression from this was the vertical seismic profile (VSP). The VSP involves repeatedly firing rig-source shots while a geophone is located at successive depth levels in the borehole; see Balch et al. (1982) for a review. The seismic data recorded can then be processed to obtain a vertical profile of the seismic reflection response of the strata at the borehole. VSPs are useful for correlating conventional surface seismic reflection sections with well-log information.

Further developments in VSP data acquisition include offset VSPs, where the source is fired at a constant offset from the rig (see e.g. Dillon and Thomson, 1984) and walkaway VSPs, where the source is fired at several different offsets while the geophone position is kept constant (see e.g. Kohler and Koenig, 1986). These illuminate the geological structure over narrow cross-sections through the borehole in the azimuthal direction of the source locations. The most general case of such profiles utilises multiple geophone positions and source offsets in the same survey.

When the source is fired within the borehole and surface receivers are used the survey is known as a hole-to-surface seismic profile (Kragh, 1990), a simpler terminology for what has also become known as the inverse multi-offset VSP

(IMOVSP) (e.g. Jackson et al., 1989). These types of survey provide increased resolution of seismic reflectors compared with conventional surface seismic surveys because the acoustic signal only passes through the near-surface layer once, whereas two passes are needed for surface seismic reflection surveys. Thus the higher temporal frequencies in the signal are less attenuated by near-surface effects.

2.2 Cross-hole seismic techniques

Cross-hole seismic methods involve firing shots in one borehole and recording the transmitted and scattered wavefields on receivers positioned in a neighbouring borehole. By positioning both sources and receivers in the sub-surface, the frequency content of the data is higher than that of VSPs. The first applications of this recording geometry were in velocity surveys where the travel times of direct arrivals are used by tomographic inversion algorithms to generate a tomogram or slice of the velocity field between the boreholes (see e.g. Worthington, 1984; Dines and Lytle, 1979). The uses of this technique are principally in locating local velocity anomalies caused by intrusions and voids (e.g. Cottin et al., 1986; Bishop and Styles, 1990), and also in monitoring the changes in a velocity anomaly with time which may occur during fluid-injection enhanced oil recovery projects (Macrides et al., 1988 and Justice et al., 1989). Kragh (1990) applied the technique in trying to locate shallow old mineworkings in Coal Measures strata by looking for velocity anomalies in the collapsed zones immediately above the old mineworkings, but concluded that any such velocity anomalies were too small to be detectable by this method.

The basic type of tomographic survey uses only the direct arrival traveltime data. Obviously additional information is also present in the later scattered arrivals. Techniques to utilise scattered arrivals have been developed by Devaney (1984) and Pratt and Worthington (1988), amongst others. The diffraction tomography method may be applied to cross-hole data to provide velocity and attenuation tomograms. An alternative to this approach is to consider a cross-hole reflection survey as being composed of several offset VSPs where the sources and receivers are all below ground level. Data processing of the survey is then similar to, but more complicated than, processing a set of offset VSPs. This processing route has previously been applied to synthetic data by Hu et al. (1988) and to ultrasonic scale-model data by Zhu and McMechan (1988) and Balch et al. (1990). This is the processing method which has been adopted in this work.

2.3 Cross-hole seismic data acquisition

As in any type of seismic survey, the choice and deployment of acquisition equipment should be considered carefully. Of particular importance are the types of source and receiver and the recording device used.

2.3.1 The source

An explosive source was used in all the trial surveys of Chapter V. The explosive charge used consisted of an electrical (no.8 type) detonator which was sometimes boosted by 25g of dynamite. The dynamite was used to improve the

signal-to-noise ratio in surveys where high background noise levels were observed, but it was generally avoided in order to minimise the risk of damage to the borehole walls. There was no noticeable change in the frequency content of data recorded using a dynamite source compared with that of data recorded using a detonator alone, suggesting that the charge scaling law proposed by Ziolkowski et al. (1980) does not hold for small-sized charges fired in boreholes.

The charge is positioned at the end of a 4cm diameter hollow steel tube which is 40cm long. The purpose of the tubing is to provide to provide sufficient weight to allow the source to be lowered and raised easily in the borehole and also to prevent the explosion from damaging the firing or triggering leads with which the source is connected to the surface. This source was found to give an adequate frequency range for the purposes of the trial surveys and also gave good shot repeatability.

2.3.2 The receivers

It was decided that, for reasons of practicality and speed, a string of receivers would be needed. The simplicity of utilising a ready-built hydrophone string greatly outweighed the complexity of constructing a string of (three-component) geophones. Hydrophones in a vertical fluid-filled borehole may be expected to detect SV body waves as well as P waves, but not SH waves (Schoenberg, 1986). However, the practical difficulties of accurately recording rock particle motion by a borehole geophone were recently demonstrated by Beattie (1990). She found that the geophone was much more sensitive to tube waves; a surprising result but explained by the effect of the pressure in the fluid due to the tube waves acting on the top and bottom of the geophone housing.

Two 12-channel hydrophone strings were used; the first (used in survey A of chapter V) had hydrophone spaced at 4m intervals and the second (used in the remainder of the trial surveys) had a 2m hydrophone spacing. The hydrophones were streamlined by enclosing each one in heat-shrink plastic to prevent snagging on the borehole wall. A weight is fastened to the bottom end of the string to facilitate lowering in the borehole. The 4m-spacing string was supported by a steel cable fed past each hydrophone and attached to the weight at the bottom. In the 2m-spacing prototype a nylon rope was used instead in order to reduce the overall weight of the string and make it easier to handle.

2.3.3 Recording equipment

Two recording devices were used in the field trials. The first was an EG&G Geometrics Nimbus 12-channel 10 bit enhancement seismograph with a dynamic range of 66dB. This was used in survey A of chapter V. In the remainder of the surveys an EG&G Geometrics ES-2401 24-channel enhancement seismograph was used. This equipment was a significant improvement on the earlier model with a much greater dynamic range and the additional benefit of storing data in SEG-DOS format on floppy disks instead of the tape reels (SEG-D format) used by the Nimbus. The advantage of 24-channel recording allowed the simultaneous acquisition of cross-hole and hole-to-surface surveys, with 12 channels allocated to the downhole hydrophones and the other 12 to a surface geophone spread. This led to a greatly improved rate of data acquisition, and reduced the risk of losing a survey due to explosive damage instigating collapses in the borehole wall.

2.3.4 Field equipment set-up

The typical equipment arrangement used in the field trials of chapter V is illustrated schematically in figure 2.1. The string of hydrophones is lowered into borehole B and the source is positioned in hole A. Shots are repeated at the same depth levels for different positions of the hydrophone string in order to extend the coverage of the receiver array. One hydrophone position is kept in common between the repeated shots to allow amplitude scaling and repeated shot deconvolution to be carried out (see section 3.2.1). In the trial surveys, the hydrophone string remained fixed while shots were fired successively at depth intervals of 2m (to make the coverage of reflectors between the boreholes as uniform as possible). The hydrophone string was then moved to a new position and the shot sequence was repeated.

A sampling interval of $200\mu\text{s}$ was used in all the trials and a data length of 1024 samples (204.8 ms) was recorded.

The deepest shot and receiver locations were usually limited by collapses in the boreholes, rather than the total depth actually drilled, and the shallowest shot and receiver positions were restricted by the level of the water table in each borehole. A hydrophone will simply not respond if it is not submerged, and there is insufficient coupling of the source energy to the surrounding rock unless the shot is also submerged. Shots fired 1-2m below the water table resulted in a noticeably lower frequency content in the recorded data; the dominant frequencies being approximately half those of the deeper shots. This was presumably due to an insufficient confining hydrostatic pressure to restrict the expansion of the exploding gas bubble.

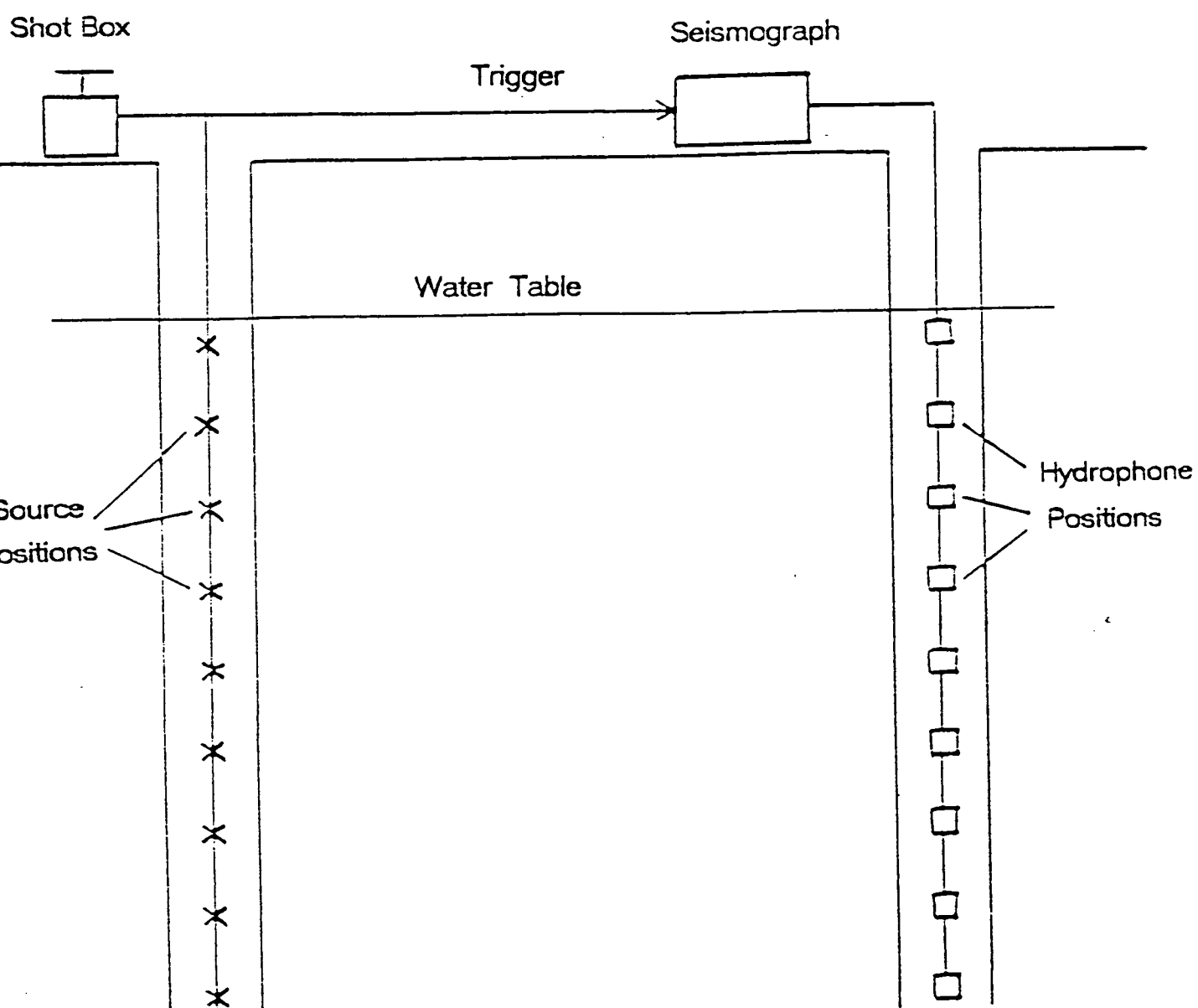


Figure 2.1 Typical cross-hole survey field set-up.

2.4 Data acquisition problems

Three distinct types of problem were encountered during data acquisition: timing errors, noise and borehole blockages.

2.4.1 Timing errors

Initial trials with the acquisition system used a Huntex shot box to fire the detonator and simultaneously send a trigger signal to the seismograph, but this was found to be insufficiently accurate, resulting in random timing errors of up to 1 ms in the recorded data. As an alternative to this, some wire was wrapped around the detonator and connected to a trigger lead above the steel tube of the source tool. The trigger lead ran up the borehole and was fed into the trigger input of the seismograph via a differentiating capacitor. The firing pulse was provided by a 12V car battery. When the detonator exploded, the wire around the detonator was broken causing a resistive change at the trigger input. This method was found to give very accurate timing breaks. Timing errors could be checked by inspecting the first break time of the common hydrophone channel between repeated shots. Occasionally random timing errors would still occur, and these were attributed to electrical pick-up between the firing and triggering lines.

2.4.2 Noise

There were two different categories of noise: shot-generated noise and background noise. The shot-generated noise consisted of tube-wave arrivals (equivalent to surface waves in conventional seismic surveys) which are low-velocity

events travelling along the borehole wall. These are discussed in section 3.3.4. No suitable acquisition technique could be used to suppress these arrivals.

Background noise was both man-made and natural, including drilling-rig vibrations (because acquisition took place on active exploration sites), and factory-generated noise at one site. Water dripping into the boreholes from above the water table caused noise events on shallow receivers and low-frequency (~ 10 Hz) pipe resonances were detected in some surveys. Pipe resonances could be removed by the use of high-pass filters, but the presence of drilling crews frequently led to long delays between successive shots, until a lull in drilling operations took place.

2.4.3 Borehole blockages

Borehole blockages were the most serious problem encountered in data acquisition, and led to several trial surveys being abandoned. After drilling was finished, the borehole would gradually fill with mud and debris which fell from the borehole walls. This obviously reduced the maximum source and receiver depths as time passed. It was therefore essential for surface casing to be left in the boreholes so that the soft near-surface drift material would not simply fall down the boreholes. Ideally, surveys would also be carried out as soon as possible after the boreholes were drilled and logged.

Unfortunately, where old mineworkings and faulting occur in the boreholes (i.e. those areas of particular interest), the borehole walls are particularly susceptible to collapse. It was frequently the case that a borehole would be blocked just above the level of old mineworkings. Surveys in available boreholes were not attempted

when the borehole walls were thought to be particularly unstable for fear of losing the hydrophone string.

2.5 Uphole shots and verticality

Two further sets of measurements were required to complete each survey: velocity readings within the boreholes and verticality readings. Unfortunately, in some cases these additional data were unobtainable due to borehole collapses.

2.5.1 Uphole velocity estimates

Interval velocity estimates are made in each borehole by firing a shot 2m below the hydrophone string. In this way approximate values of the vertical velocity field can be obtained. A sampling interval of 100 μ s is used and worst errors, Δv , in the interval velocities, v , can be estimated using the following equation (Stewart, 1984):

$$\Delta v = 2v \frac{\delta t}{\Delta t} \quad (2.1)$$

where Δt is the travel time between two receiver positions, and δt is the timing error. To avoid the risk of losing the hydrophone string, it was sometimes necessary to estimate the interval velocities by recording the traveltimes to a surface geophone

from several shot positions, but this resulted in less accurate first break picks and thus greater errors in the interval velocity field estimates.

2.5.2 Borehole verticality

The deviation of the borehole from the vertical and the azimuth of this deviation were measured by using a pendulum-type inclinometer. This tool could be run down the borehole within its own casing in two perpendicular directions, but had a depth limit of 60m. The dip readings were converted to borehole displacements using the program of Howson and Sides (1986). On the whole, borehole deviations were slight, particularly at shallow depths, and were rarely more than 3.0 metres at the bottom source and receiver depths. Neighbouring boreholes usually deviated in a similar manner.

Chapter III

Data processing

3.1 Introduction

The close resemblance of cross-hole seismic reflection common shot gathers to offset vertical seismic profiles (OVSPs) provides an obvious route for processing these surveys. Essentially, each common shot gather may be thought of as an individual OVSP, but with the additional complication of having some receiver positions at shallower depths than the level of the source. A typical data processing scheme for cross-hole data is shown in figure 3.1. The processing consists of several distinct stages: editing, wavefield separation, deconvolution, velocity field estimation and imaging. These are discussed below.

3.2 Editing

Following data acquisition, the field data are transferred from DOS-format floppy disks to the University of Durham Amdahl 470/V8 mainframe computer. Data from combined hole-to-surface and cross-hole surveys (see section 2.3.3) are then separated using utility software, and the cross-hole data are then sorted into common shot gathers.

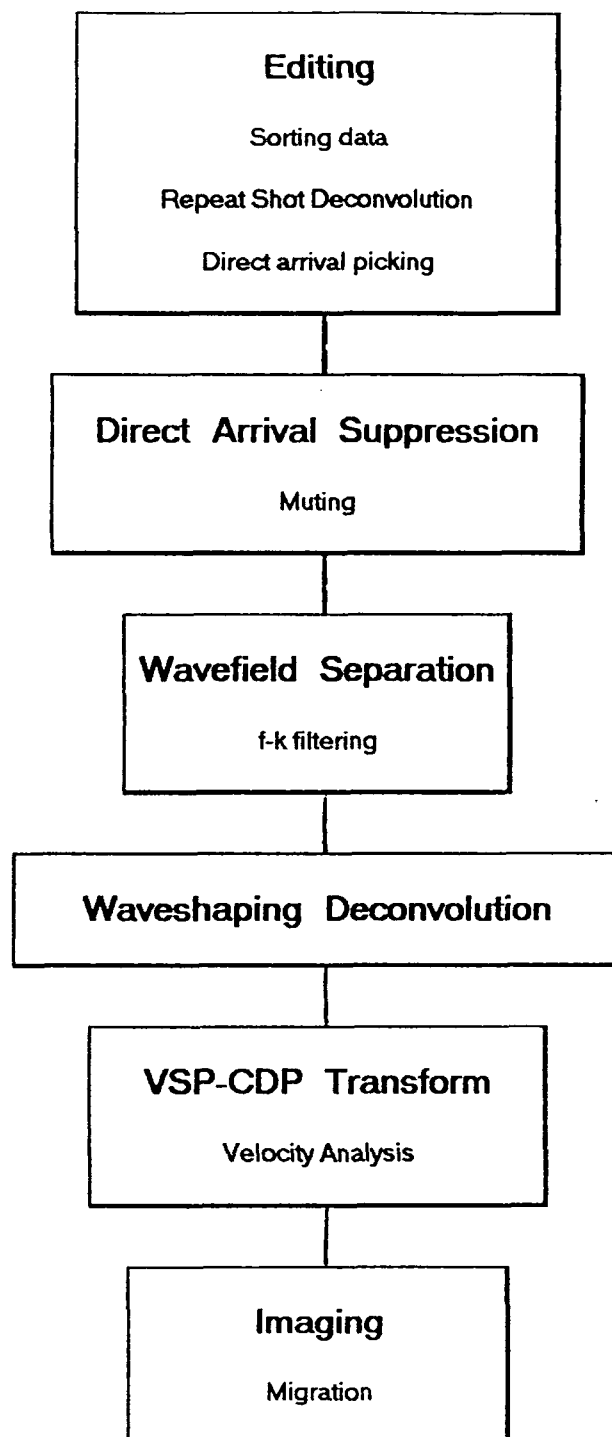


Figure 3.1 Outline of cross-hole reflection data-processing scheme.

3.2.1 Shot-matching deconvolution

Usually each common shot gather comprises two (or more) positions of the hydrophone string with at least one hydrophone position kept in common between adjacent string positions for the repeated shots. This is used as a precautionary measure to ensure that there are no time break errors in the survey and also to check for the repeatability of the source waveform. In cases where the common shot traces do not correspond, a waveshaping Wiener filter may be designed and applied to one set of hydrophone positions so that a match will occur on the common trace. An example of such a deconvolution is shown in figures 3.2-3.5 .

Figure 3.2 shows the raw recorded data for a particular common shot gather. In this case, data were recorded by firing a shot whilst the hydrophones were at depths (12-34 m) and then moving the hydrophones so that they lay at depths (34-56m) and firing a second shot at the same depth. Thus channels 12 and 13 correspond to the common channel for this common shot gather. There is a noticeable mismatch in the character of these traces and so a deconvolution operator was designed along the following lines. Firstly, a suitable time window (including the first arrivals and any dominant reflections) was chosen for the two traces and the segments of data within the window were tapered. A least-error-energy Wiener waveshaping filter was then calculated which, when convolved with the window from trace 13, would result in an output which approximated the window from trace 12. Figure 3.3 illustrates the windowed data, the deconvolution operator, and the filter output as time series. The respective amplitude spectra are plotted in figure 3.4. Figure 3.5 is a plot of the common shot gather following the application of the same waveshaping filter to traces 13-24.

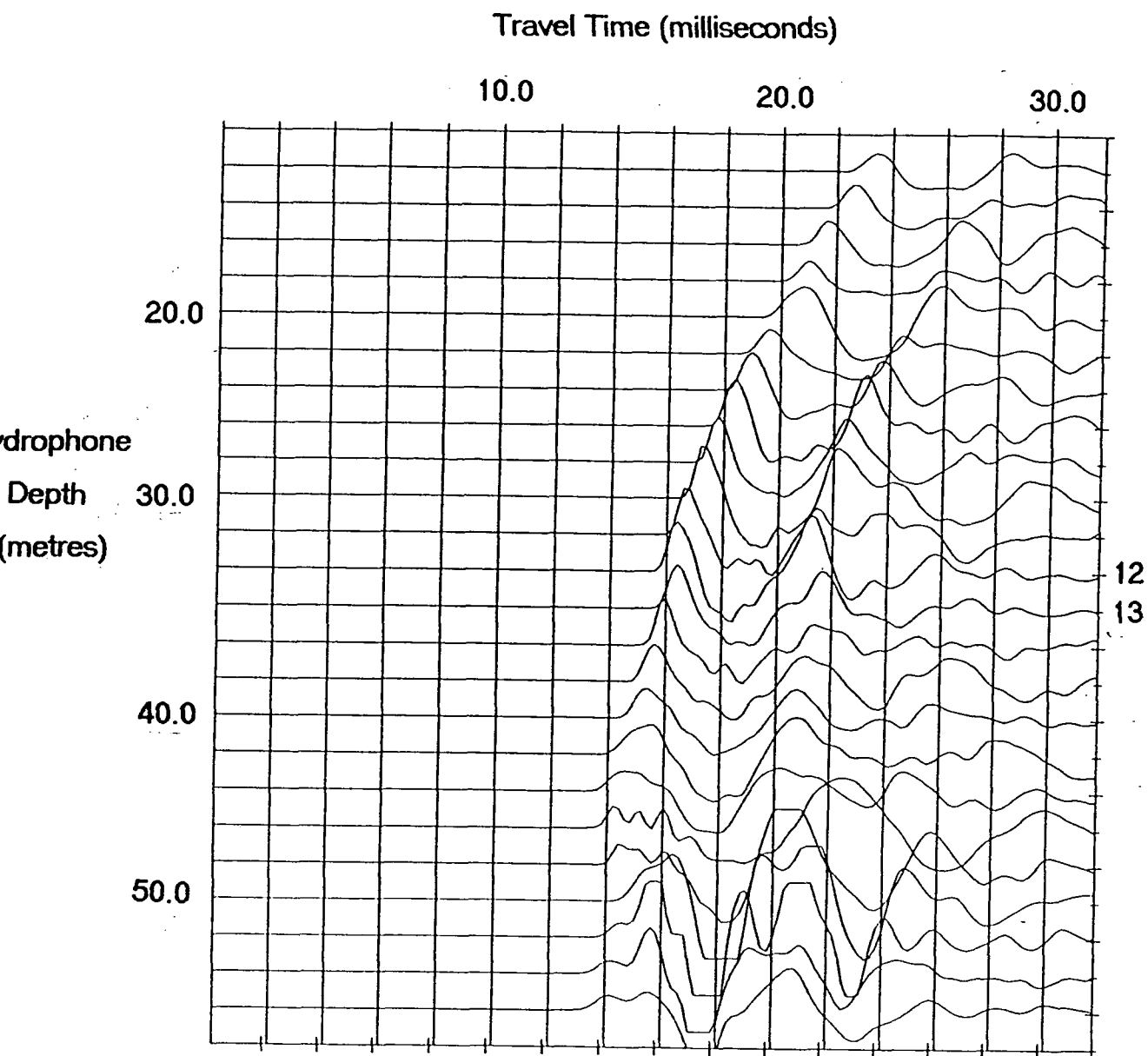


Figure 3.2 Unprocessed common shot gather. Channels 12 and 13 are repeat shot common receivers.

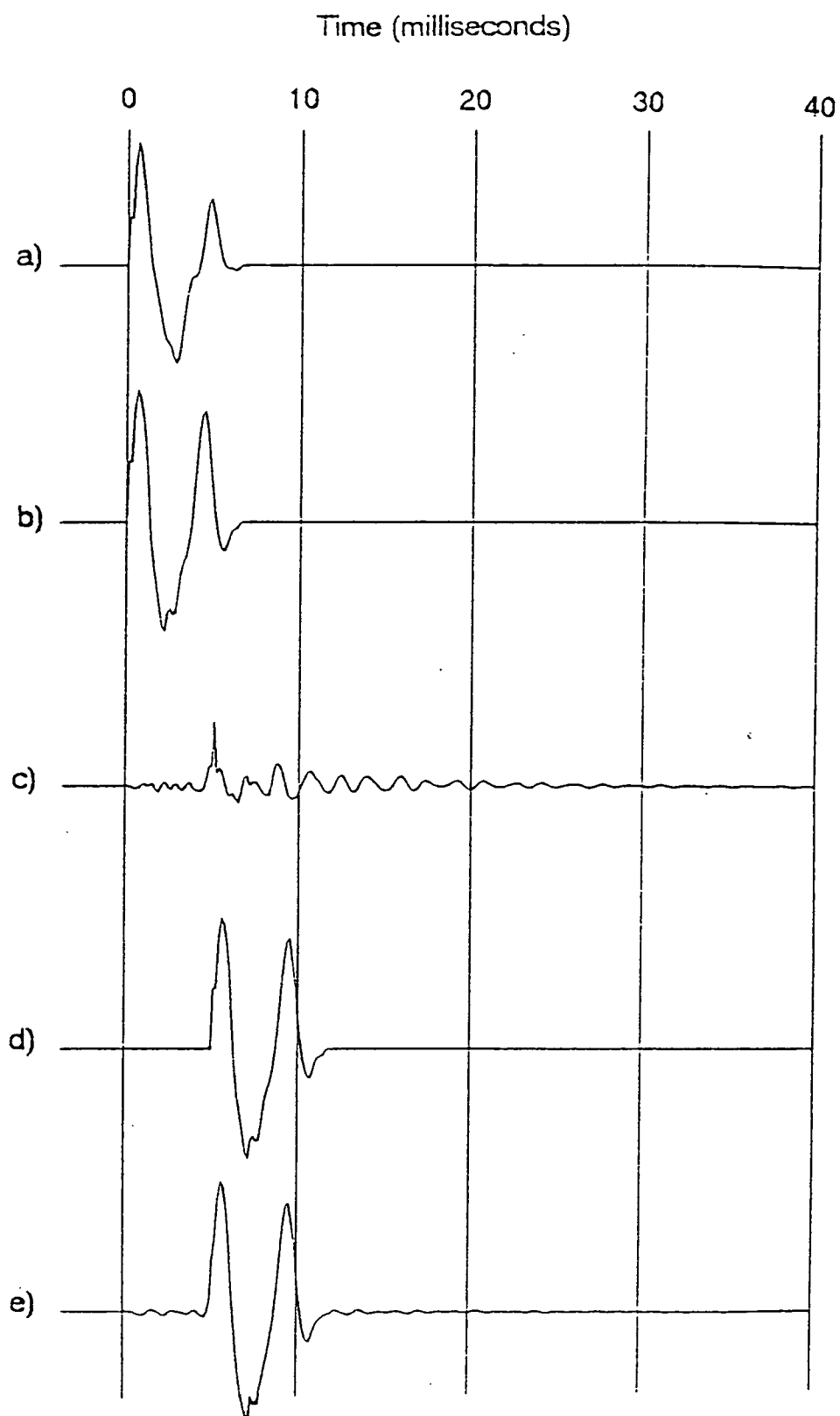


Figure 3.3 Repeat shot deconvolution: a) input signal, b) desired output signal, c) filter, d) lagged desired output signal, e) convolution of input signal and filter.

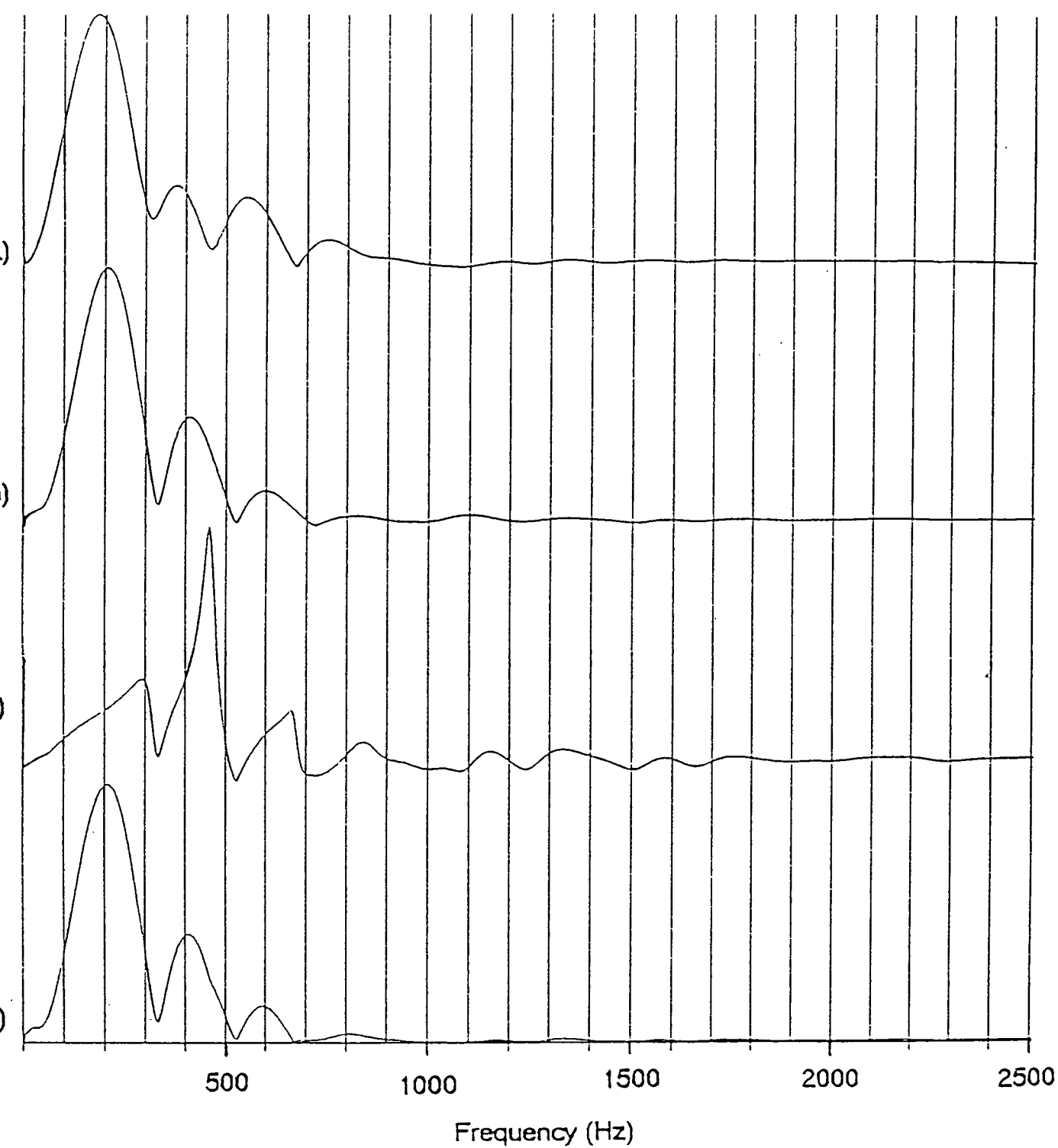


Figure 3.4 Repeat shot deconvolution - amplitude spectra: a) input signal, b) desired output, c) filter, d) filtered input.

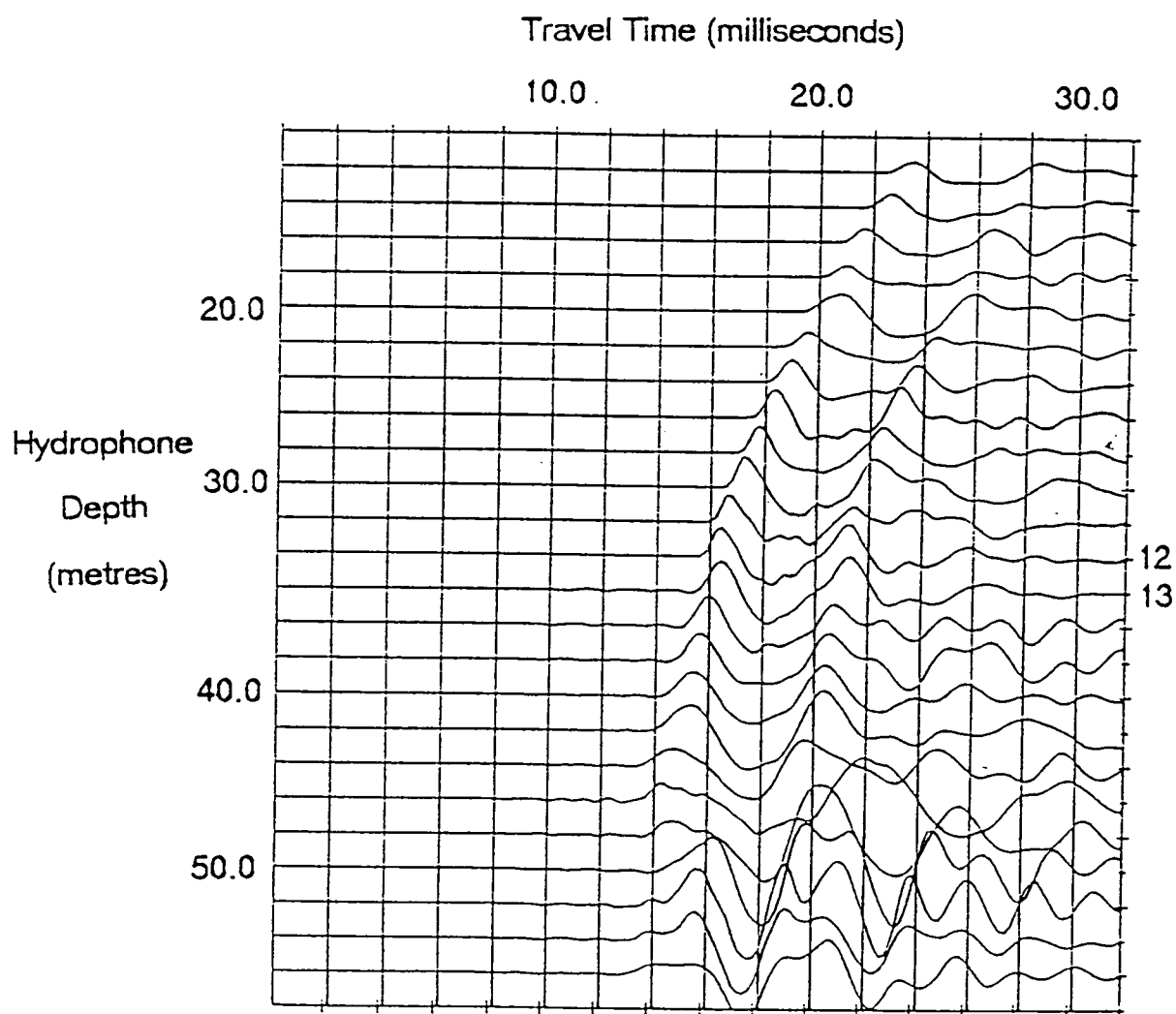


Figure 3.5 Common shot gather after repeat shot deconvolution.

After this preliminary deconvolution, an amplitude scaling factor is applied to equalise the shot energies for the two hydrophone positions. This is taken to be the ratio of the rms amplitudes of the common receiver traces.

3.2.2 First break estimation and direct arrival suppression

In order to mute out the large-amplitude direct arrivals, it is first necessary to estimate the direct arrival times at each hydrophone position. An automatic first-break picking program was found to give good results for cross-hole data provided the signal-to-noise ratio was sufficiently high (as is usually the case with cross-hole data). First breaks are picked on a statistical basis, by simply locating the time sample where the recorded signal amplitude rises sufficiently above the rms amplitude of the background noise. In addition, it was found to be necessary to check all first-break times manually to avoid the inclusion of "head wave" arrival picks.

Once direct arrival times have been picked from the data, it is a simple matter to mute the direct arrival energy by specifying a window for a cosine taper on each trace. Some reflected energy is undoubtedly lost by removing the direct arrivals in this way. Unfortunately this means that reflector coverage will be lost near the receiver borehole, but this method was found to be the most suitable for the field datasets discussed in chapter V.

An alternative scheme for direct arrival removal has been proposed by Pratt and Goulty (1991) who applied it to ultrasonic model data. This removes the direct arrival waveform of a particular trace by aligning the first breaks of those traces which correspond to nearest neighbour sources and receivers. These trace are then averaged to provide an estimate of the direct arrival waveform which may then be

subtracted from the original trace to yield the scattered wavefield. This method was found to be unsuitable for the cross-hole datasets discussed later, because the direct arrival waveform was not sufficiently consistent between adjacent traces. This is probably due to peg-leg multiples which will be present in data acquired from the real earth with its small-scale layering, as opposed to that acquired from an ultrasonic modelling system where the model consists of relatively few homogeneous layers.

Direct arrival muting may be carried out either before or after wavefield separation. Inclusion of the direct arrivals in the wavefield separation can lead to ringing problems when frequency-wavenumber filters are applied to the data (see section 3.3), but on traces where the direct arrivals are downgoing and the interesting reflections are upgoing, it is preferable to mute after separation in order to preserve as much of the scattered wavefield as possible. This is the final stage of the editing procedure, and the processing sequence then continues with the application of wavefield separation filters.

3.3 Wavefield separation

The scattered wavefield recorded for each common shot gather may be classified as consisting of upgoing and downgoing components. These refer to waves which travel upwards or downwards, respectively, across the receiver array. The upgoing and downgoing scattered energies must be imaged independently because of the net cancellation effect which would result in combining their respective reflectivity amplitudes.

Clearly, for a particular reflecting interface, the signals which are recorded from shots above the interface on receivers above the interface (i.e. upgoing reflection events) will be of opposite polarity to those which are recorded from shots below the interface on receivers also below the interface (i.e. downgoing reflected events). Since the apparent velocities of these upgoing and downgoing events are of opposite sign ($\frac{dz}{dt} > 0$ for downgoing events; $\frac{dz}{dt} < 0$ for upgoing events) these events are readily separable by means of velocity filters.

For zero-offset VSP surveys, separation may be carried out by median filters which rely upon the linear moveout of reflection events (Hardage, 1983). When the source is offset from the receiver array, however, this moveout is non-linear and so it is necessary to apply filters which can selectively pass or reject arrivals with a range of apparent velocities. One such filter type is the pie-slice filter (Embree et al., 1963), which is applied to data which have undergone a two-dimensional Fourier transform into the frequency-wavenumber (f - k) domain.

The two-dimensional Fourier transform is implemented on digitally recorded data by means of the Fast Fourier Transform or FFT (Cooley and Tukey, 1965). This algorithm requires that the data consist of 2^n samples (n being an integer) and assumes that the data have a periodicity equal to this sample size. To carry out a two-dimensional transform on a common shot gather the algorithm may be applied first in the time direction on all traces, and then in the spatial (depth or z) direction at all frequencies, although the sequence of these operations is immaterial.

In dealing with digitally sampled data, there are some important factors to be considered which do not occur with analogue data. In particular, in discrete Fourier space, there may be problems due to the data being undersampled and thus not uniquely defined.

3.3.1 The Nyquist frequencies and aliasing

If the recorded data have a time sampling interval Δt and the receivers are regularly spaced at intervals of Δz , then the temporal and spatial Nyquist frequencies are respectively defined as:

$$f_{\text{NYQ}} = \frac{1}{2\Delta t} ; k_{\text{NYQ}} = \frac{1}{2\Delta z} \quad (3.1)$$

Only if all (f,k) components in the recorded data lie within the range governed by the positive and negative Nyquist values will the data be completely determined by the recorded samples. If (f,k) components outside the Nyquist range are present then they will be aliased when they are recorded. This means that they will wrap around in the discrete Fourier space because of the periodic nature of the transform and contaminate the unaliased (f,k) components. March and Bailey (1983) give an informative account of the f - k transform.

3.3.2 Two-dimensional filter application

Filter application in the f - k domain involves complex number multiplications of each component in f - k space with the corresponding component of the filter. This operation is equivalent to a convolution in the z - t domain of the original data with the impulse response of the filter.

One characteristic of the Fourier transform is that discontinuities or steep slopes which are present in a function in one domain lead to the Gibb's phenomenon, also known as ringing, in the transformed data. With cross-hole seismic data it is

common to find that the data taper smoothly towards zero amplitude in the temporal direction but, because of the restriction imposed by the finite recording aperture of the receiver array, there are sharp discontinuities at the edges of the data in the spatial direction. Normally these would result in ringing in the wavenumber domain. It is thus necessary to apply a taper across the edge traces in the spatial direction, and a simple cosine taper over four traces is found to be adequate for this purpose.

Another problem with the discrete Fourier transform is that the data, although recorded over a finite range of times and depths, are assumed to be periodic outside the recorded ranges. Hence, a multiplication in the f - k domain is equivalent to a cyclic convolution, and not a transient convolution. Convolution in the time domain results in an output which has a length equal to the length of the data plus the length of the filter. When the filter is applied in the f - k domain, the extra output length will become wrapped around "noise" on the outer traces after the reverse transform is applied. This filter leakage can be avoided if the data are padded out with blank traces before applying the FFT. The total number of traces is made up to the next power of two, since the FFT algorithm requires that the number of samples to be processed is a power of two.

3.3.3 Filter design

In the case of pie-slice filters which are applied to common shot gathers of cross-hole seismic data, it is usually necessary to design the f - k filter with steep slopes in order to separate the upgoing and downgoing events, whilst rejecting as little of the appropriate P-wave reflection energy as possible. In order to accomplish this, while at the same time keeping the ringing effects induced by the steep slopes to a

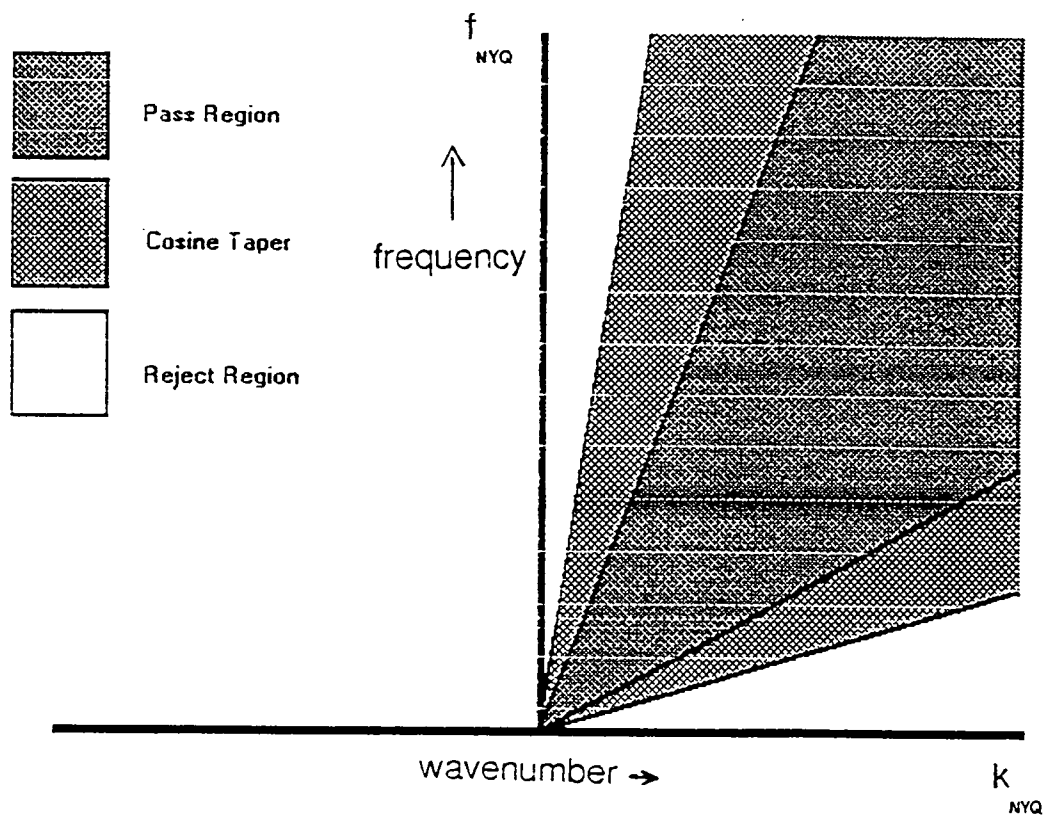


Figure 3.6 Two-dimensional f - k filter implementation

minimum, the edges of the velocity filters are selected so that they lie along low amplitude channels in the data; an approach advocated by Christie et al. (1983) and similar to the contour-slice filtering proposed by Suprajitno and Greenhalgh (1985). Filter edges are tapered by a smooth cosine taper which runs perpendicular to the bisector of two lines in f - k space which determine the tapering region (figure 3.6).

Other considerations of filter design include the rejection of events other than P-wave reflections. Although the hydrophone receivers used in the cross-hole surveys discussed later are pressure sensitive devices, they will also record SV energy when they are located in a borehole (Schoenberg, 1986). The apparent velocity of SV arrivals ranges from infinity (wavefronts parallel to the borehole axis) down to the SV wave velocity in the rock medium (raypaths parallel to the borehole axis). Thus direct wave SV arrivals are difficult to filter out but SV arrivals with apparent velocities less than the P-wave velocity in the rock may be readily removed. Generally, it would only be mode-converted P $\rightarrow$ SV reflections which would arrive within the time window of interest in the surveys reported here.

3.3.4 Tube waves and spatial aliasing

Another type of coherent noise event, which is more significant in borehole seismic data, is the tube wave. The tube waves, which have been observed in the boreholes discussed in chapter V, typically have a larger amplitude and lower frequency than body waves and can be thought of as a type of surface wave (the Stoneley wave) which travels along the interface between the borehole wall and the borehole fluid with an elliptical particle motion. These waves sometimes emanate from the top (or water table) and bottom of the receiver borehole, and frequently

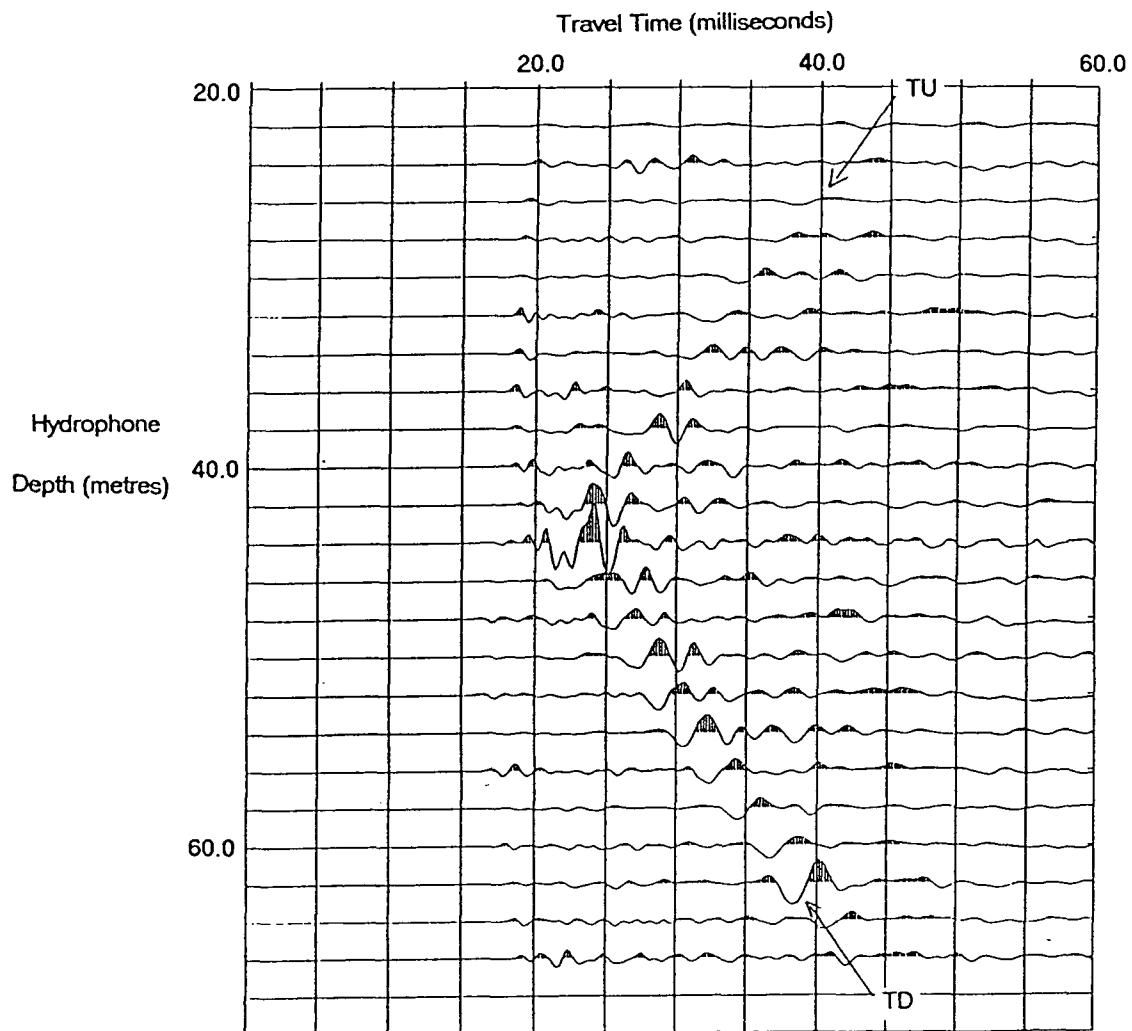


Figure 3.7 Tube waves example: upgoing and downgoing tube wave events are indicated by TU and TD respectively.

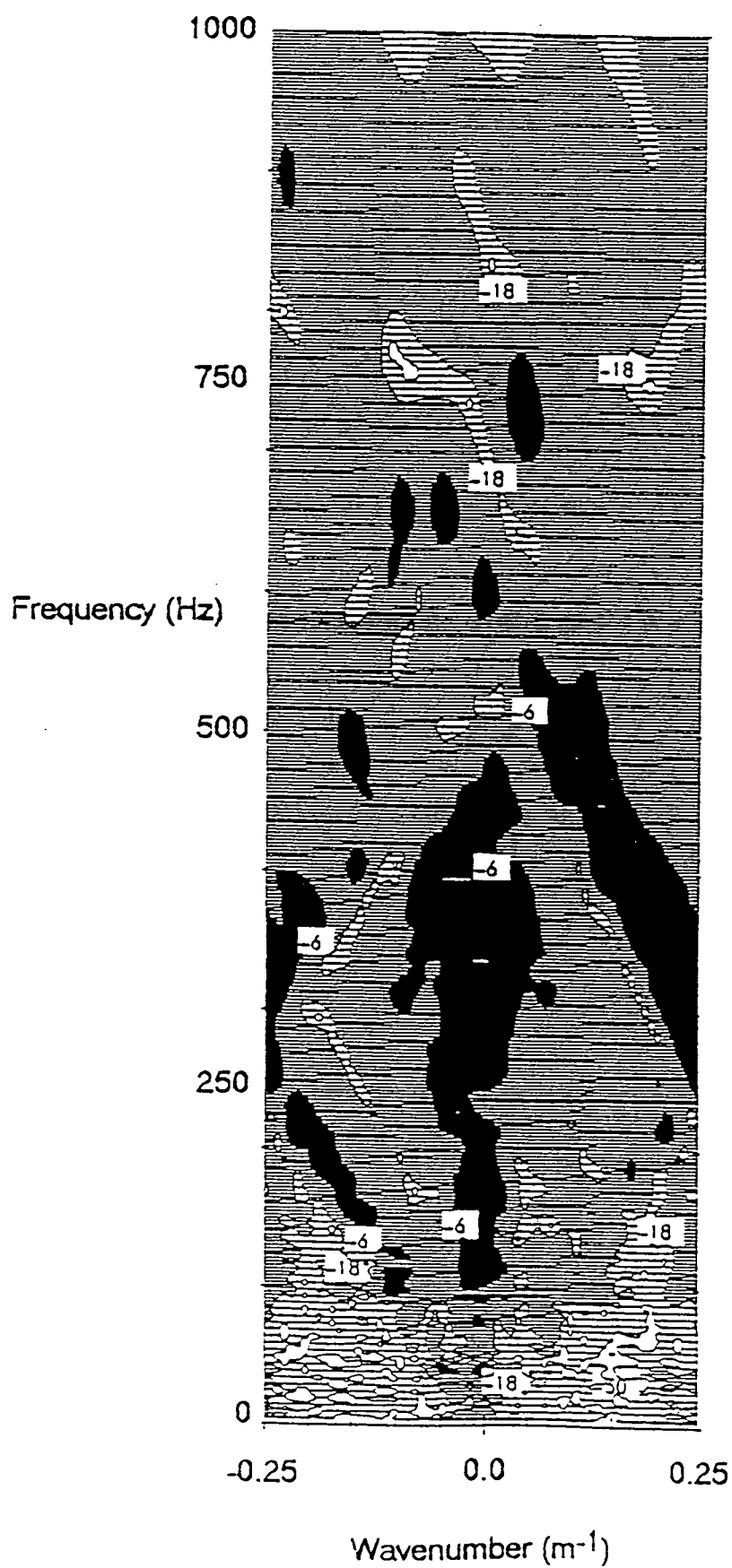


Figure 3.8 Tube wave amplitude spectrum. Contour levels are at 6, 18 and 30 dB below peak amplitude.

originate at seam level depths in shallow Coal Measures strata. One such tube wave is shown in figure 3.7 with its f - k spectrum in figure 3.8.

Examination of the moveout per trace of the tube wave arrival in figure 3.7 reveals that this is greater than half the dominant period of the arrival and so it is spatially aliased. An event with an apparent velocity V_{APP} will be spatially aliased for temporal frequencies greater than $f_{ALIAS} = V_{APP} / k_{NYQ}$. A spatially aliased event wraps round in the f - k domain (see figure 3.8) due to the periodicity of the discrete Fourier transform. Therefore, in order to remove spatially aliased noise from the data, it is necessary to apply a rejection filter which itself will wrap around in the f - k domain. Unfortunately, this type of filter will also remove reflected P-wave energy from the quadrant in f - k space into which it wraps round, thus reducing the information content of the data. In most cases where this was done it did not have a severe effect on the data.

An alternative method of removing low velocity spatially aliased noise is to apply a linear moveout to the data before applying the two dimensional FFT and thereby alter the moveouts of all events so that the spatially aliased events are no longer aliased and can be removed with ease in f - k space. The artificial moveout is then removed following a transformation back to the z - t domain. This method can lead to events, which were not previously aliased, becoming aliased, and so the method is highly data-dependent and must be applied with care. In most cross-hole surveys where tube waves occurred they were both upgoing and downgoing which meant that this method could not be applied (de-aliasing of one tube wave type would mean even more severe aliasing of the other).

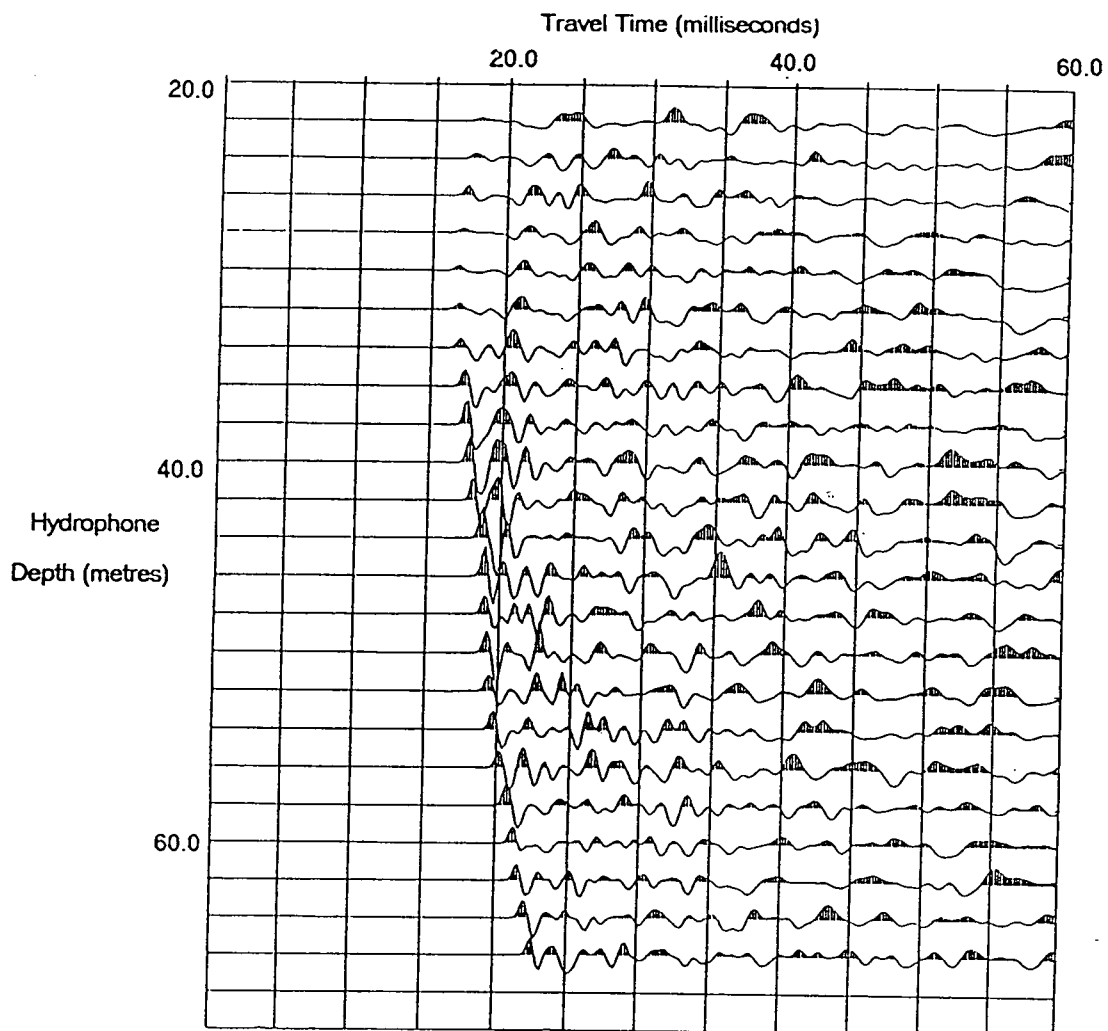


Figure3.9 A common shot gather from survey A of chapter V.

The data processing software (Appendix A.1) allows for several different filter design options, including the wrap around type of filter and trapezoidal-shaped filters as well as pie slice filters. All filters can be specified as pass or reject types.

3.3.5 Examples of wavefield separation in the f - k domain

A common shot gather from one particular survey (survey A in Chapter V) is illustrated in figure 3.9. In this case the shot was positioned at a depth of 30m and the hydrophones were positioned at 2m intervals over a depth range of 22-66m. The receiver borehole was at an offset of 56m from the source. The data consist of both upgoing and downgoing reflections with direct arrivals also being present.

The data were transformed into f - k space to yield the amplitude spectrum of figure 3.10. This spectrum is plotted on a dB scale with the 6dB contour representing component amplitudes which are half the maximum amplitude. The principal features of this spectrum include: large direct arrival amplitudes around the $k=0$ axis and to the left of it; upgoing scattered arrivals to the right of this axis; and downgoing arrivals to the left of this axis. A pie slice filter designed to pass only the upgoing energy is specified by the lines OA and OB. This filter will pass all upgoing arrivals which travel with an apparent velocity of between 2000 m/s and 8000 m/s across the receiver array. Following application of the filter, the data were transformed back into z - t space (figure 3.11). The filter has successfully removed downgoing energy and enhanced the appearance of the upgoing reflected energy. Some of the direct arrival energy (on the upper traces) is also upgoing and so remains after the filter has been applied, but this may be removed by the application of a mute.

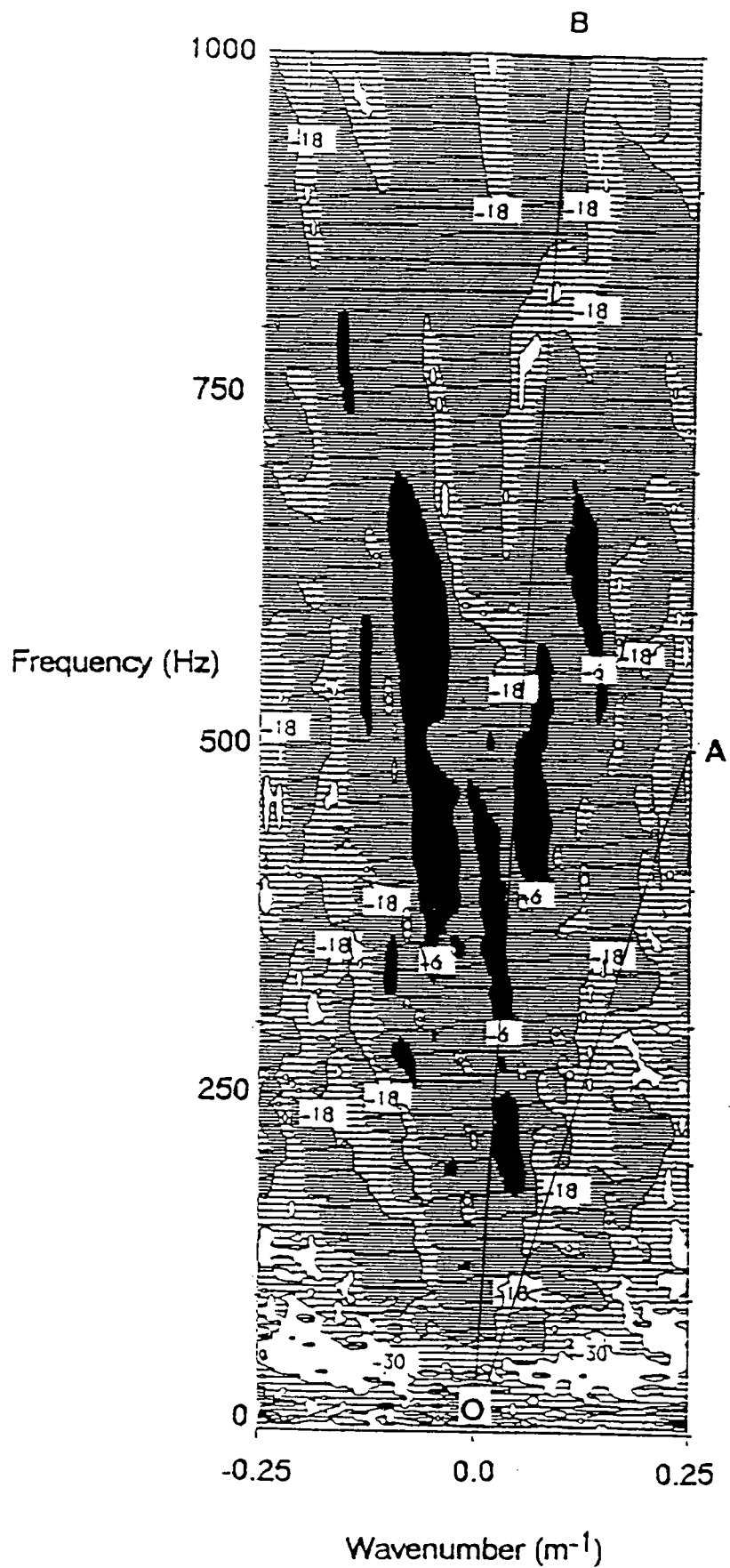


Figure 3.10 f-k amplitude spectrum of the common shot gather shown in figure 3.9. Edges of a filter to extract the upgoing wavefield are indicated by the lines OA and OB.

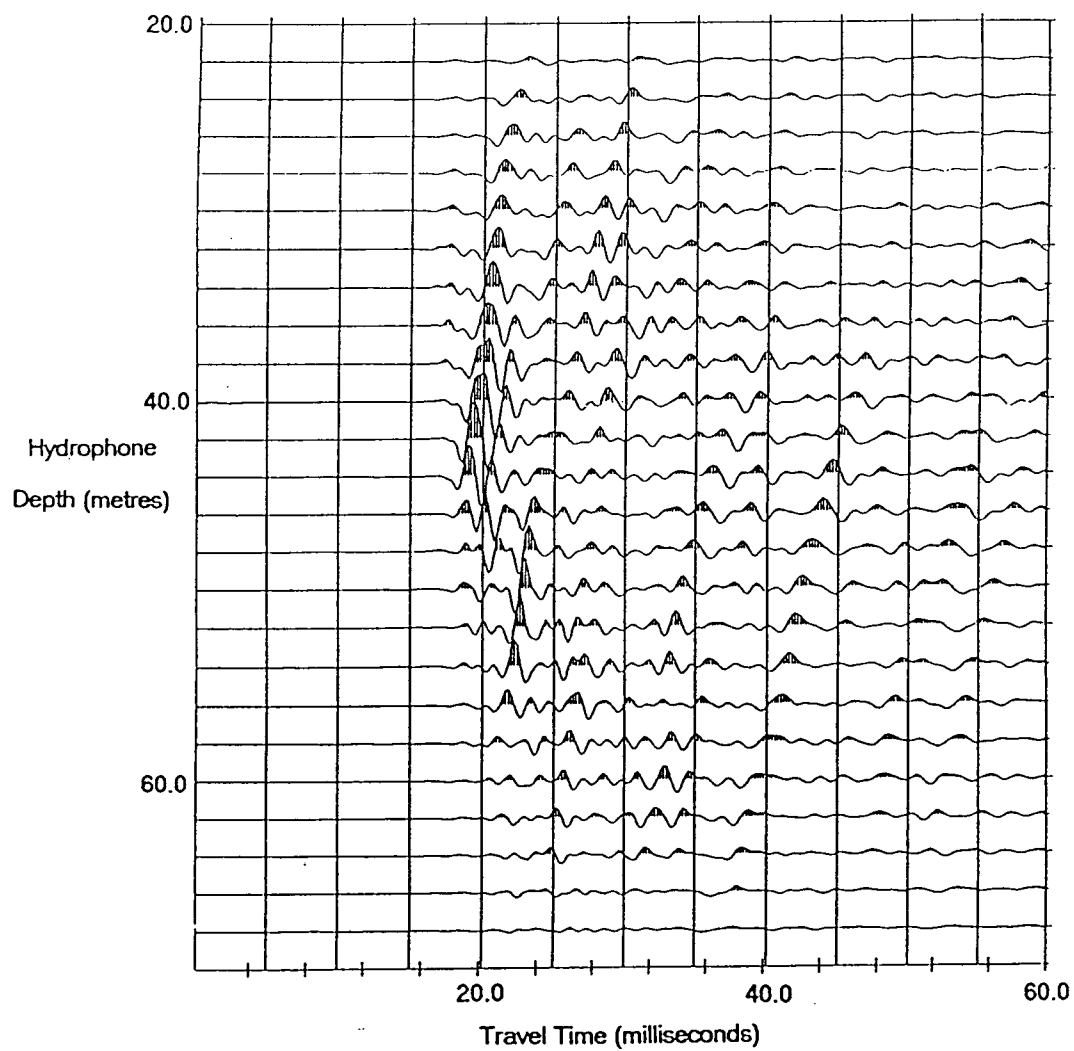


Figure 3.11 Upgoing wavefield after the application of a wavefield separation filter to the data of figure 3.9.

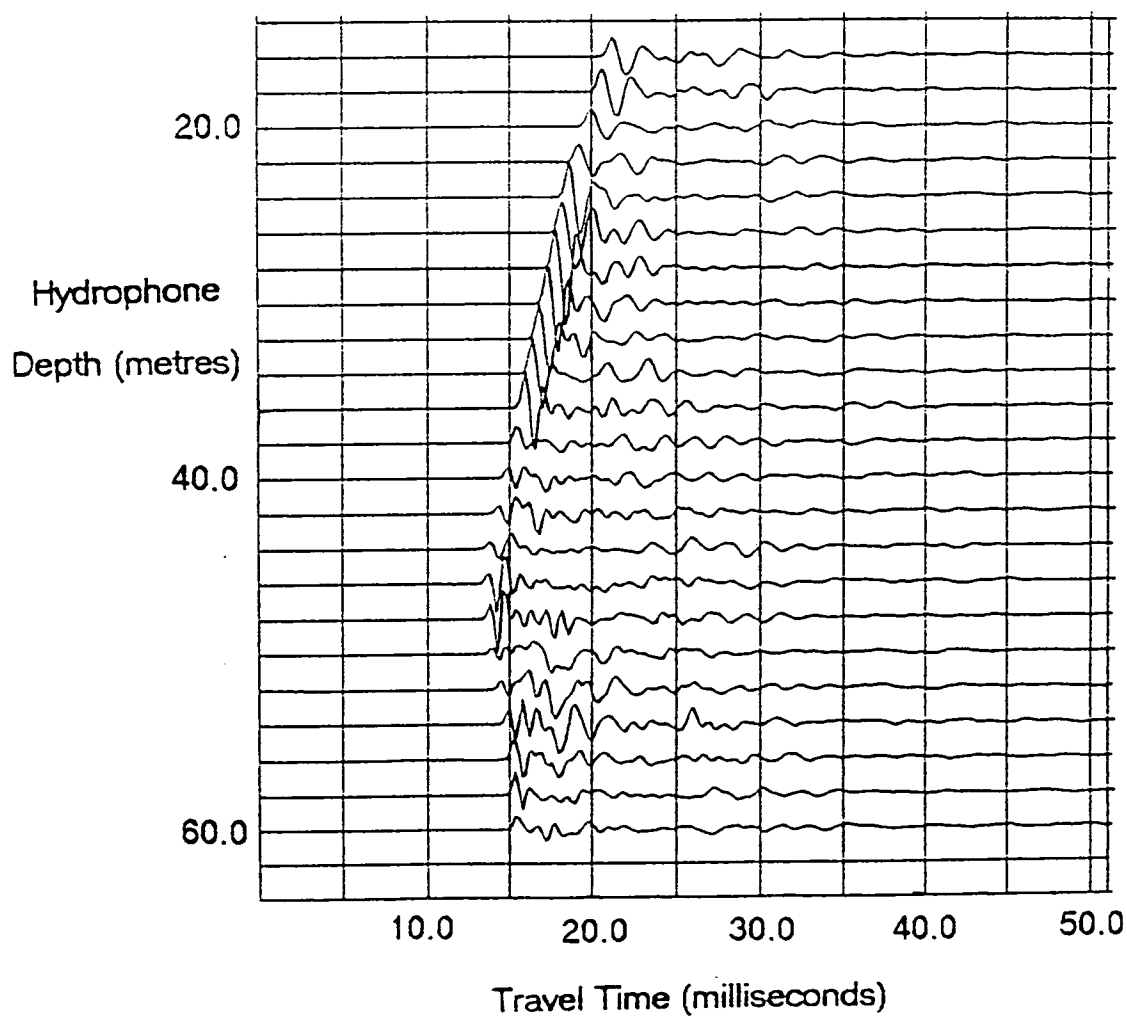


Figure 3.12 Common shot gather from survey B of chapter V.

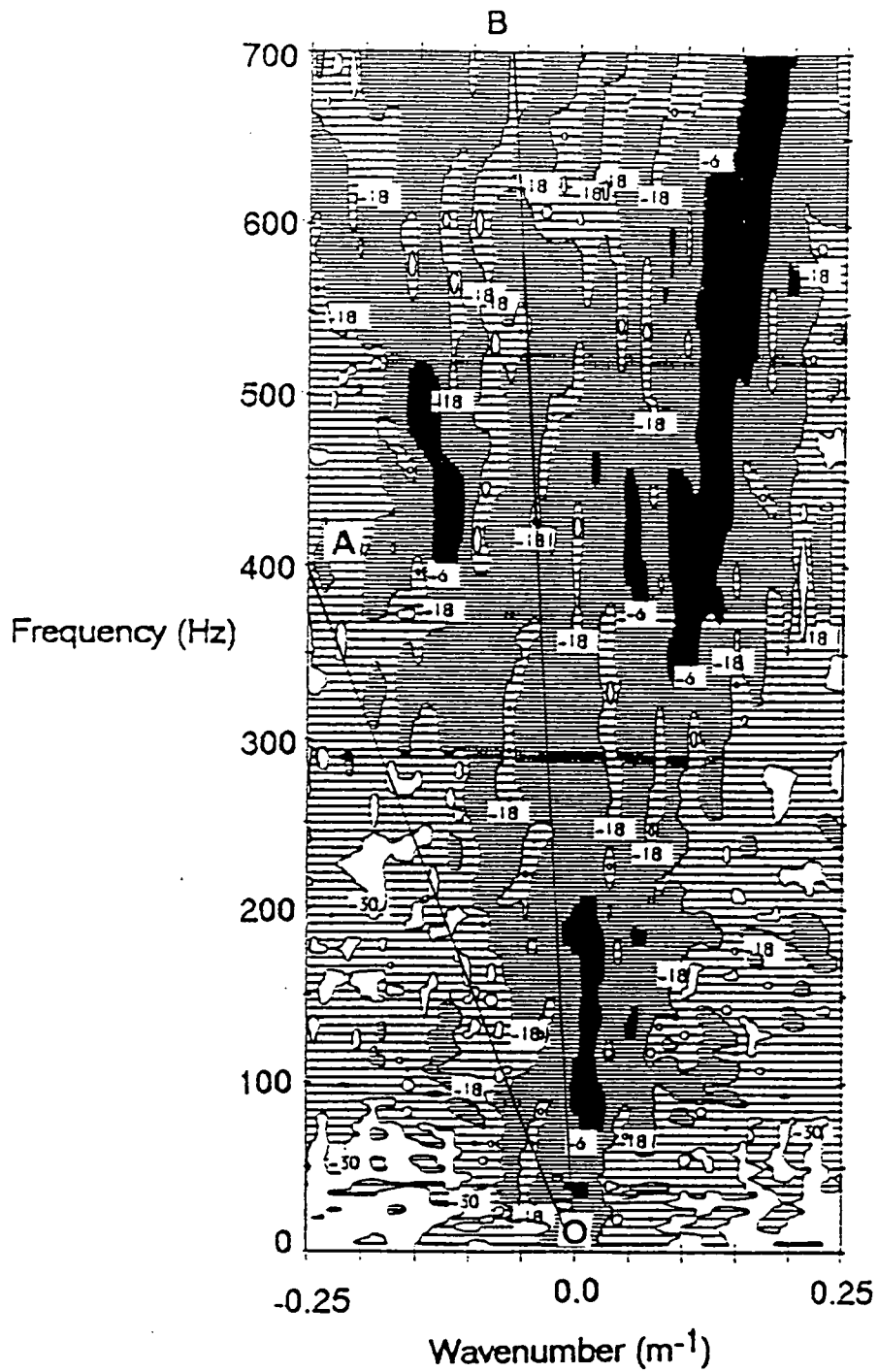


Figure 3.13 f-k amplitude spectrum of the common shot gather shown in figure 3.12. Edges of a filter to extract the downgoing wavefield are indicated by the lines OA and OB.

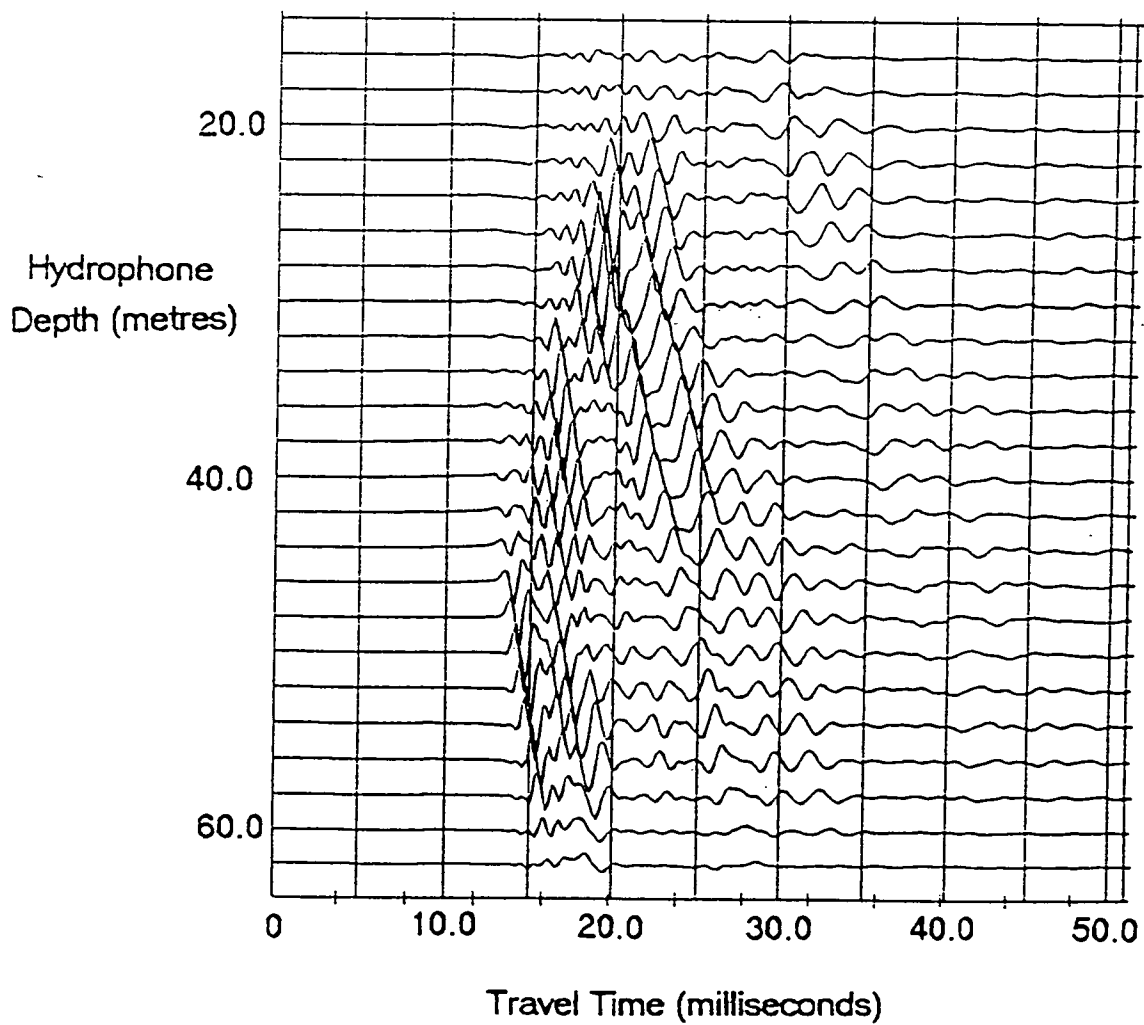


Figure 3.14 Downgoing wavefield after the application of a wavefield separation filter to the data of figure 3.12.

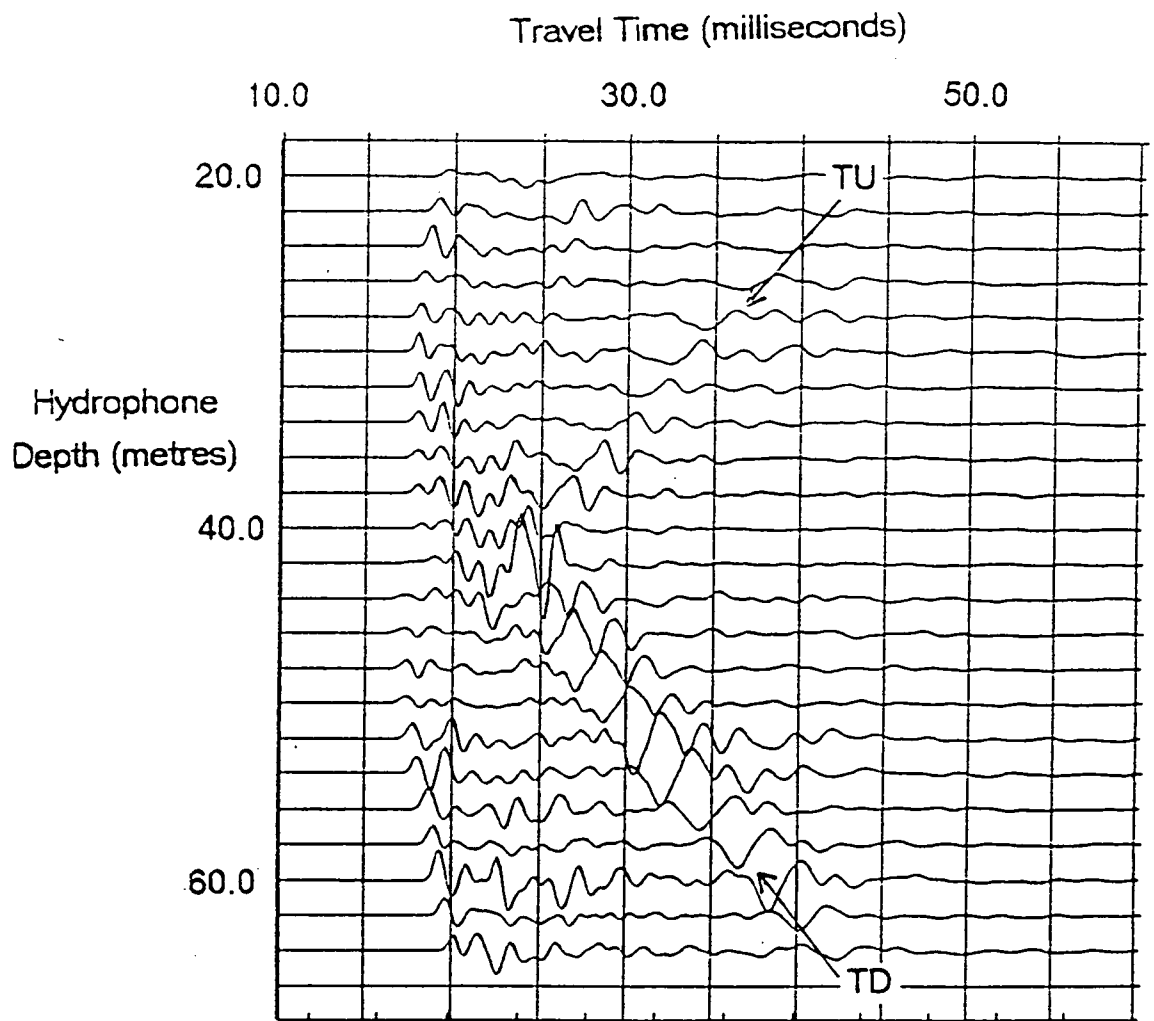


Figure 3.15 A common shot gather with tube waves indicated by TU (upgoing tube wave) and TD (downgoing tube wave).

Figure 3.12 is another common shot gather acquired on survey B of chapter V. In this case the shot depth is 48m and the receiver borehole offset is 40.9m, with 23 hydrophone positions covering a depth range of 16-60m. The two-dimensional amplitude spectrum is shown in figure 3.13. Again a filter has been defined by the lines OA and OB, in this case to pass velocities of between 1600 and 10000 m/s in the downgoing wavefield. Figure 3.14 shows the filtered common shot gather in z - t space. The downgoing reflected arrivals are now apparent, and the remaining direct arrival energy can again be muted out.

An example of a low-frequency high-amplitude tube wave with an apparent velocity of 1100 m/s is presented in figure 3.15. The events (both upgoing and downgoing) originate from a depth of 44m which corresponds to a coal seam in the receiver borehole. The shot was fired at the seam level depth. Presumably energy from P-wave first arrivals has been focused as "diving rays" by the low velocity of the coal relative to its surrounding bedrock. This would result in a large amplitude arrival at the receiver borehole which has in some way triggered off a tube wave, perhaps because the receiver borehole is caved at the seam level. The wave is obviously spatially aliased (i.e. undersampled by the receiver array) with the moveout per trace being greater than half the dominant period of the wave.

When the data are transformed into the f - k domain the amplitude spectrum (figure 3.16a) also shows the aliasing effect. The dominant downgoing tube wave wraps around in the k -direction from $-k_{\text{NYQ}}$ to $+k_{\text{NYQ}}$. In order to remove the tube wave noise from the data, it is necessary to apply a reject filter with parallel edges so that as much of the P-wave arrival energy is preserved as possible. The filter used is indicated by the lines AB, CD, EF and GH. On returning to z - t space (see figure 3.16b), we can see that the downward-travelling aliased event has been suppressed,

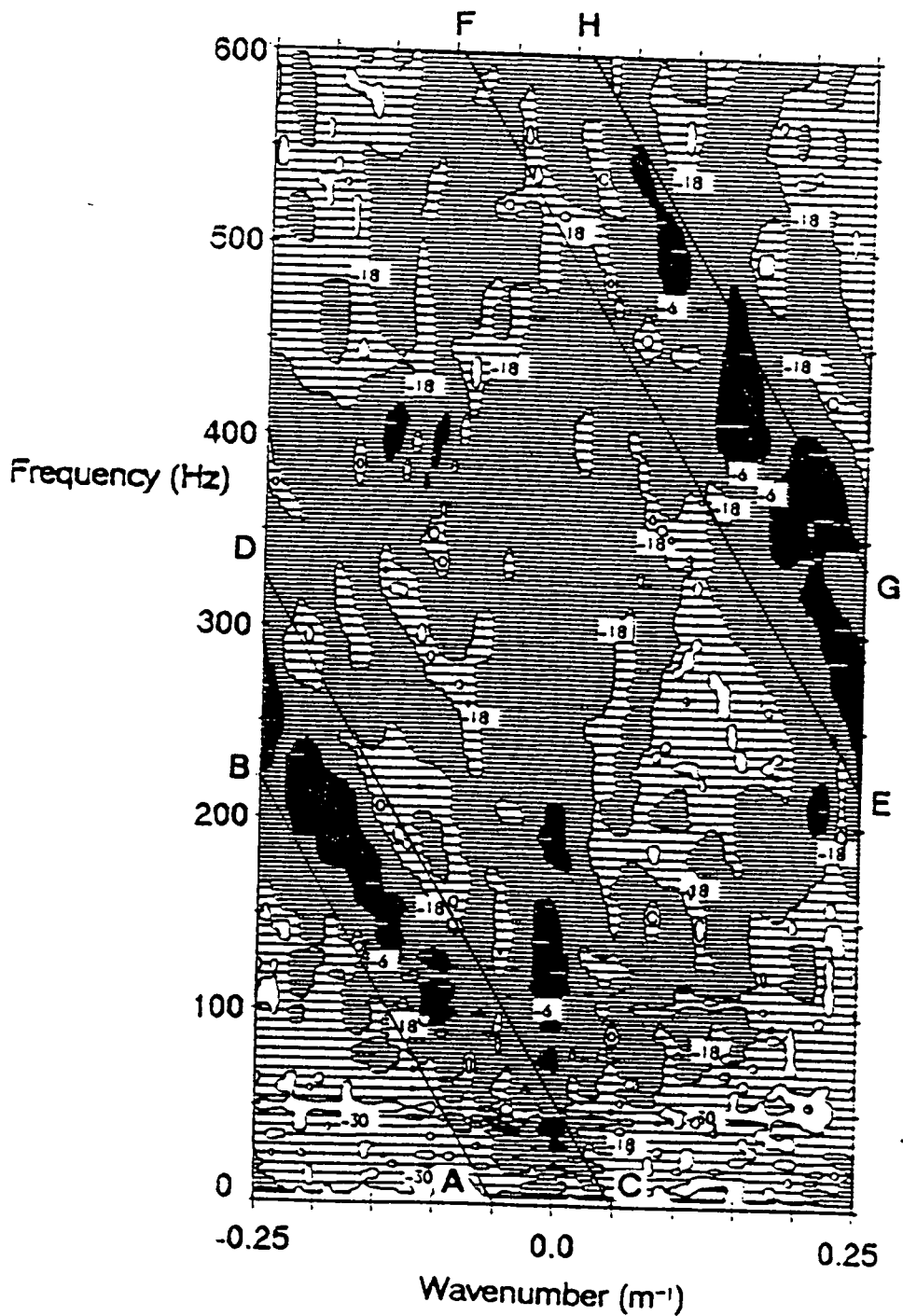


Figure 3.16a f-k amplitude spectrum of the data shown in figure 3.15. Filter edges for the removal of the downgoing tube wave are marked by AB, CD, EF, GH.

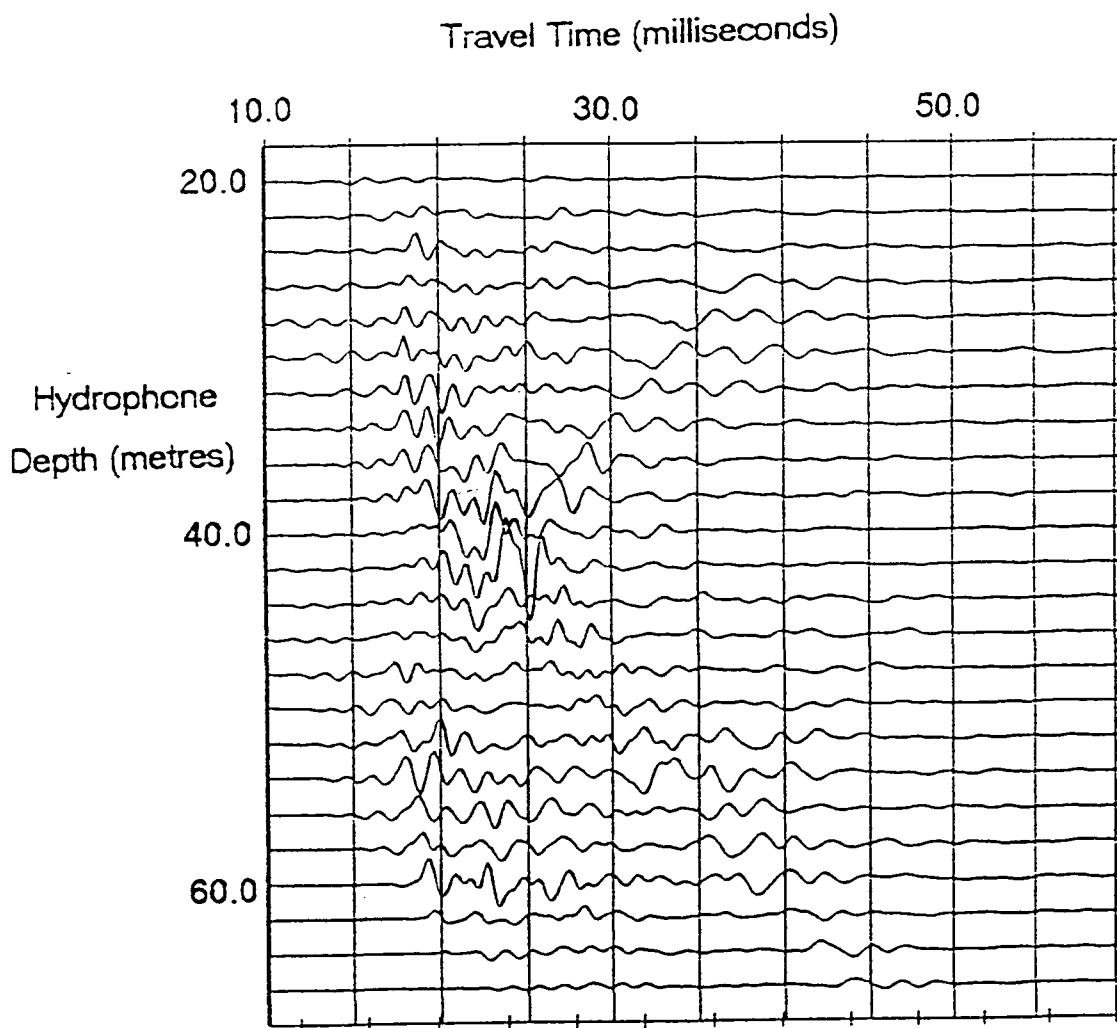


Figure 3.16b Common shot gather after the application of a reject filter to remove the downgoing tube wave.

although this has been at the expense of some of the reflected arrival information in the data. Ringing has also been introduced into the traces because of the sharp filter edges required to remove the tube wave.

3.4 Waveshaping deconvolution

In order to enhance the higher frequencies present in the data before they are imaged, a band pass filter may be applied which can suppress the low frequency (in this case below 200 Hz) components. In addition, there are advantages in changing the phase of the source wavelet in the data. The recorded seismograms may be thought of as convolutions of a real (causal) source function with the reflectivity response of the earth. As such, the onset of each arrival corresponds to the traveltimes along the reflected raypath. It is preferable to produce a section in which the central peak of the arrival waveform corresponds to this traveltimes. Such a waveform may be thought of as a band-limited spike with zero phase.

A least-squares energy Wiener waveshaping filter (see e.g. Robinson and Treitel, 1985) may be calculated to shape an estimate of the source wavelet into a zero-phase wavelet with a modified amplitude spectrum. This filter may then be convolved with the data to yield a zero-phase section. For the purposes of this deconvolution, the upgoing and downgoing wavefields in each common shot gather are processed separately because of possible directivity effects in the source signature. An estimate of the source waveform is obtained by assuming that the source function is a minimum phase wavelet and then applying a spectral factorisation (Claerbout, 1976) to a window of each trace in the common shot gather. The estimate of the

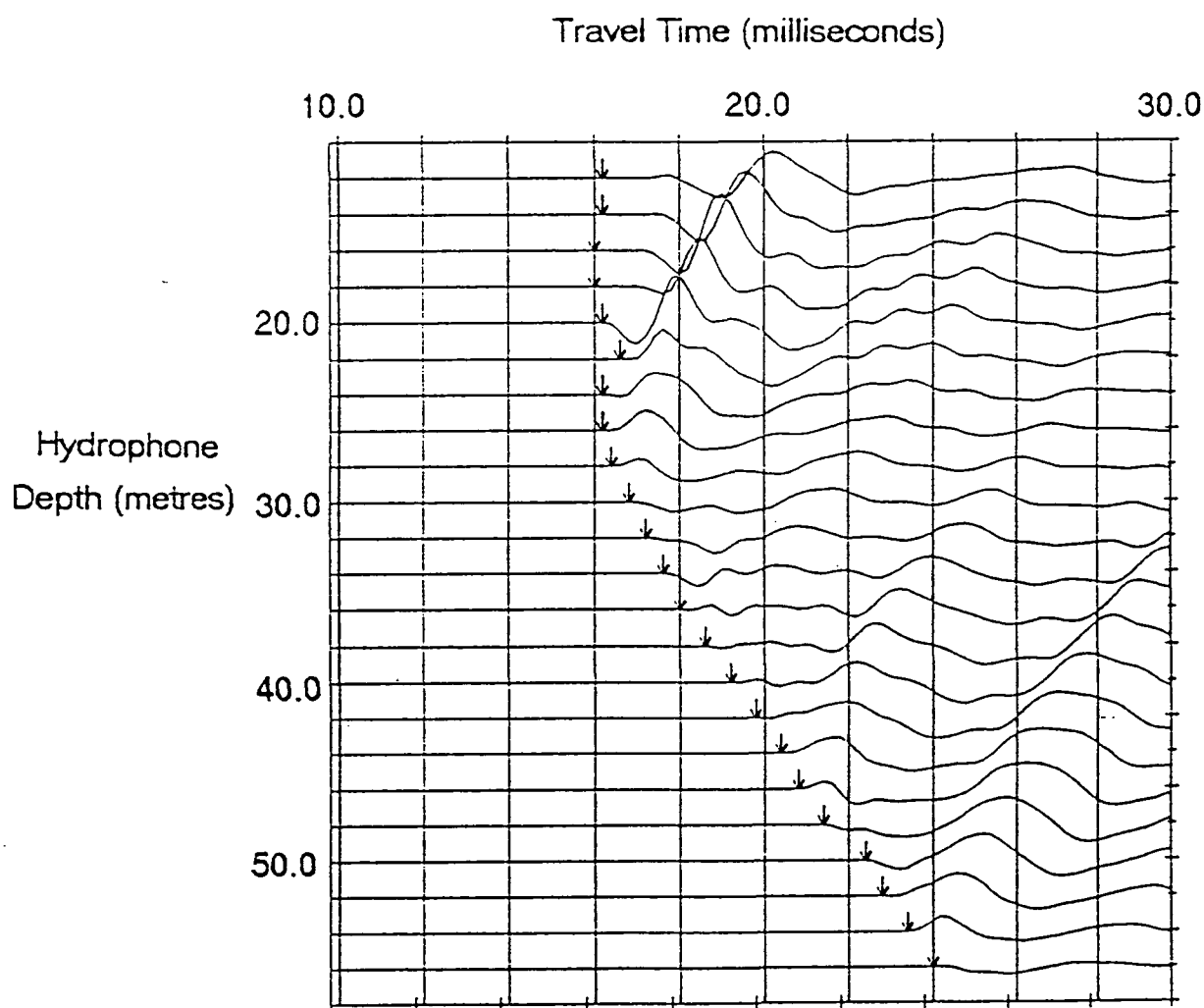


Figure 3.17 Waveshaping deconvolution: Upgoing wavefield common shot gather. Arrows indicate direct arrival picks.

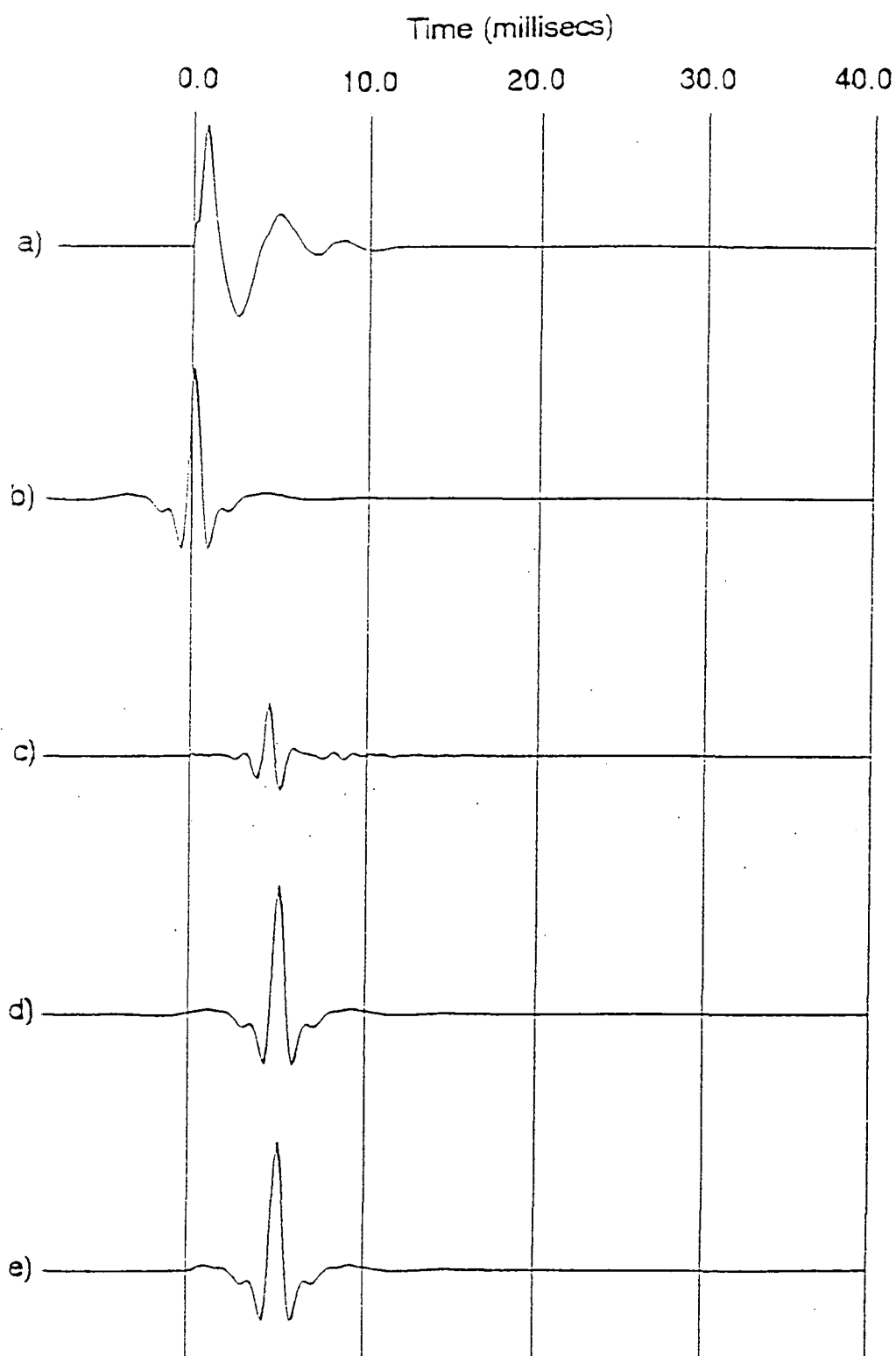


Figure 3.18a Waveshaping deconvolution: a) input wavelet, b) desired output zero-phase wavelet, c) waveshaping filter, d) lagged desired output, e) convolution of input wavelet and filter.

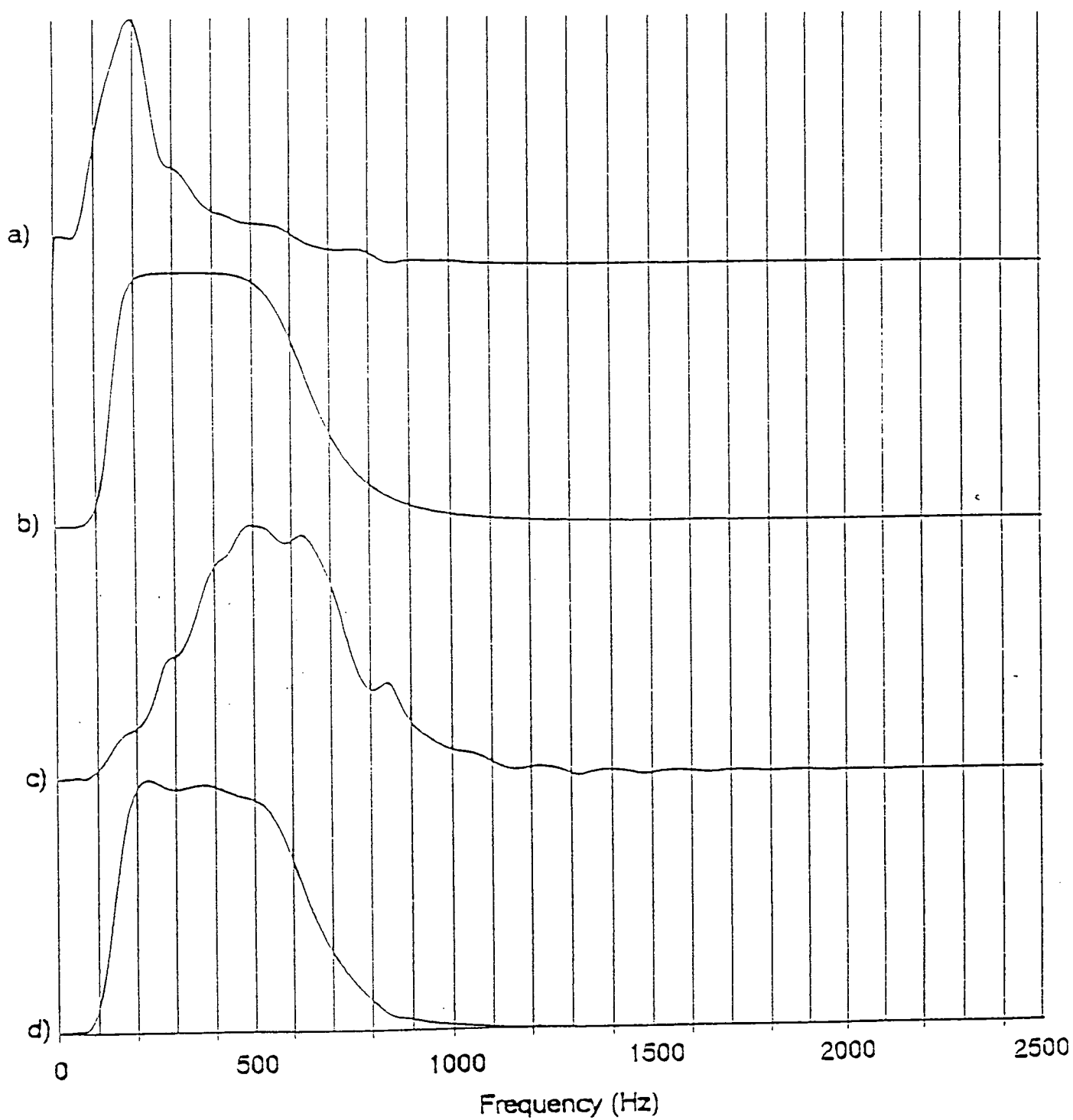


Figure 3.18b Waveshaping deconvolution amplitude spectra: a) input, b) desired output, c) filter, d) filtered input.

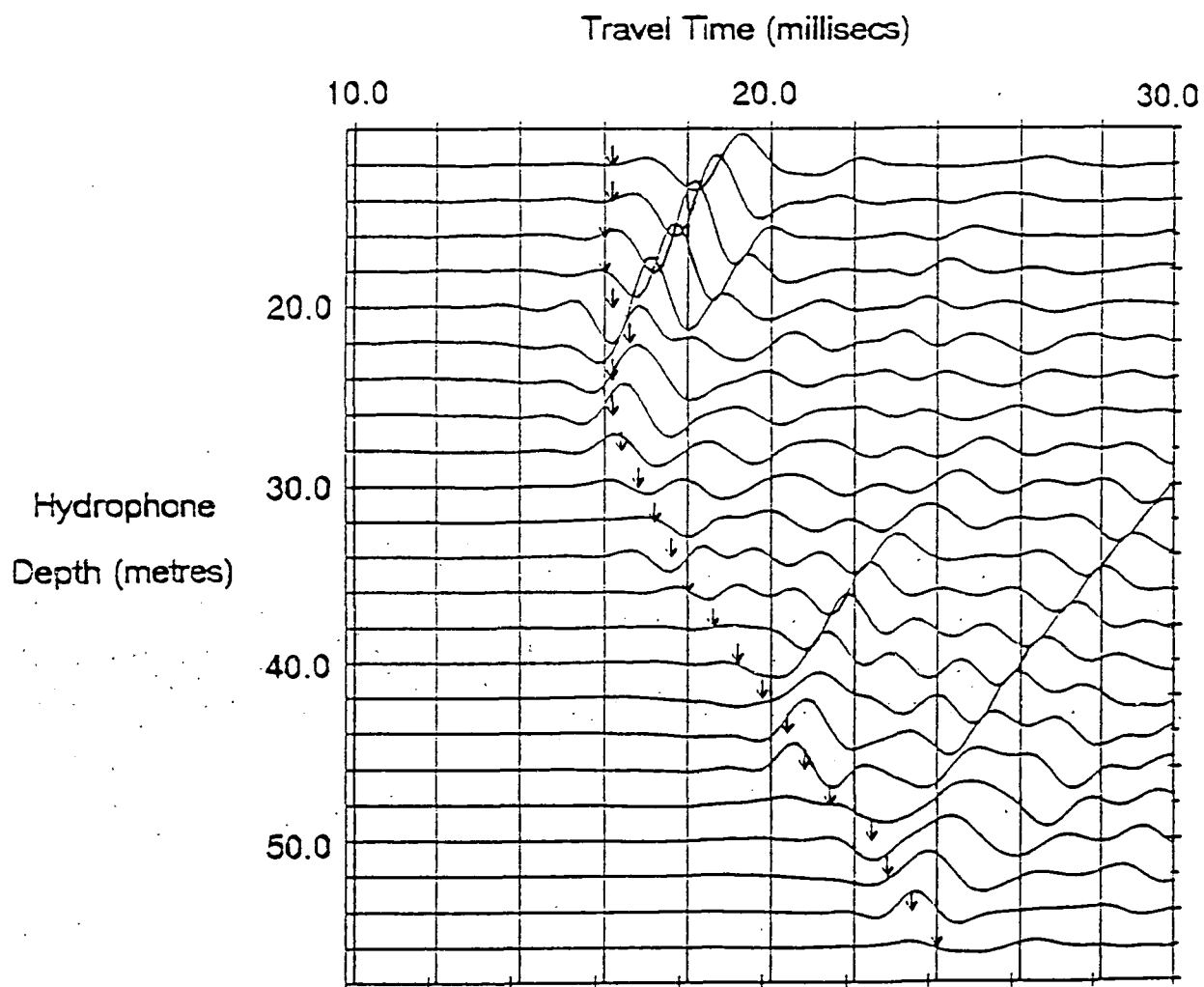


Figure 3.19 Common shot gather of figure 3.17 following waveshaping deconvolution.

waveform is then assumed to be an average of the estimates from each trace. Figures 3.17-3.19 illustrate the application of this method to the upgoing wavefield in a common shot gather.

Figure 3.17 is a plot of the upgoing wavefield after direct arrival suppression. Figure 3.18a consists of four waveforms. The first is the estimate of the source function obtained in the manner described above. The second is the desired output, a zero-phase Butterworth wavelet, and the third is the calculated filter coefficients. The bottom trace is a convolution of the filter with the input wavelet. The respective amplitude spectra of these traces are plotted in figure 3.18b. Figure 3.19 is the same common shot gather after the application of the designed waveshaping filter to all the traces. A comparison with figure 3.17 shows that the lower frequencies have been suppressed, and that zero crossings in figure 3.17 now correspond to peaks in figure 3.19.

3.5 Velocity field estimation

The direct arrival times, which are picked in the editing stage, can be used as the input to a Simultaneous Iterative Reconstruction Tomography (SIRT) velocity inversion program, written by Dyer (1988) and developed by Wye (1986), Findlay (1987) and Kragh (1990). Other P-wave velocity estimates can be obtained from uphole shots and by inspection of the near horizontal raypath traveltimes.

The results of tomographic inversion of the direct arrival times were disappointing with significant artifacts appearing in the final image at the corners. The pitfalls of using a tomographic inversion method on the cross-hole survey

geometries which are available from shallow opencast exploration boreholes in the U.K. are discussed by Kragh (1990). Essentially, poor discrimination between headwave and direct arrival types and the lack of vertical raypaths provided by cross-hole surveys results in poor imaging conditions. Figure 3.20 is typical of the types of tomographic inversion obtained from cross-hole surveys in shallow Coal Measures strata. It was found to be more useful to estimate the inter-hole velocity fields by using uphole shots and by inspection of the near-horizontal raypath travel times. Velocity models were assumed to have no lateral velocity variations between the boreholes (other than some overall dip) because of the evenly bedded nature of the strata, which is typical of Coal Measures sequences, and also because no data were available to justify including lateral variations.

3.6 The reflection point mapping scheme or VSP-CDP transform

In the general case of the cross-hole survey method, sources and receivers are located at different depths. Therefore, the image points of reflected arrivals in a time section (common shot gather) do not correspond to the simple common depth point (CDP) case used in surface seismic imaging (see figure 3.21). Instead of the vertical CDP locus, the locus of possible image points follows a curved path defined by the source and receiver positions and the velocity field between the boreholes. This locus has been applied by Dillon and Thomson (1984) to offset VSP data. It has been termed the 'VSP-CDP transform' as it provides a processing route which converts the data from VSP coordinates (depth of borehole receiver, or source, and traveltimes) into the familiar coordinates of surface seismic sections (lateral position and depth or

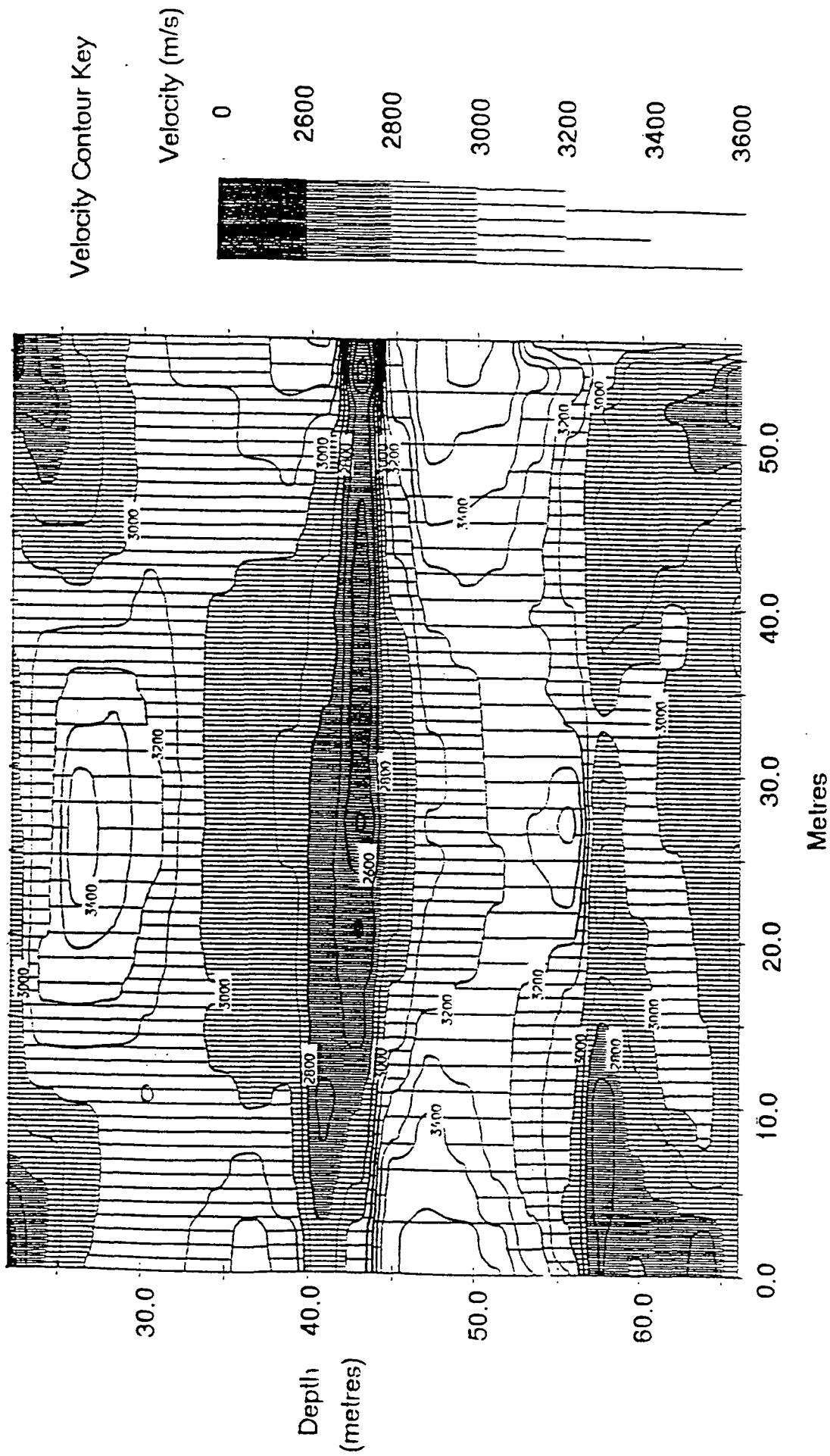


Figure 3.20 A sample tomographic velocity inversion (from Kragh, 1990).

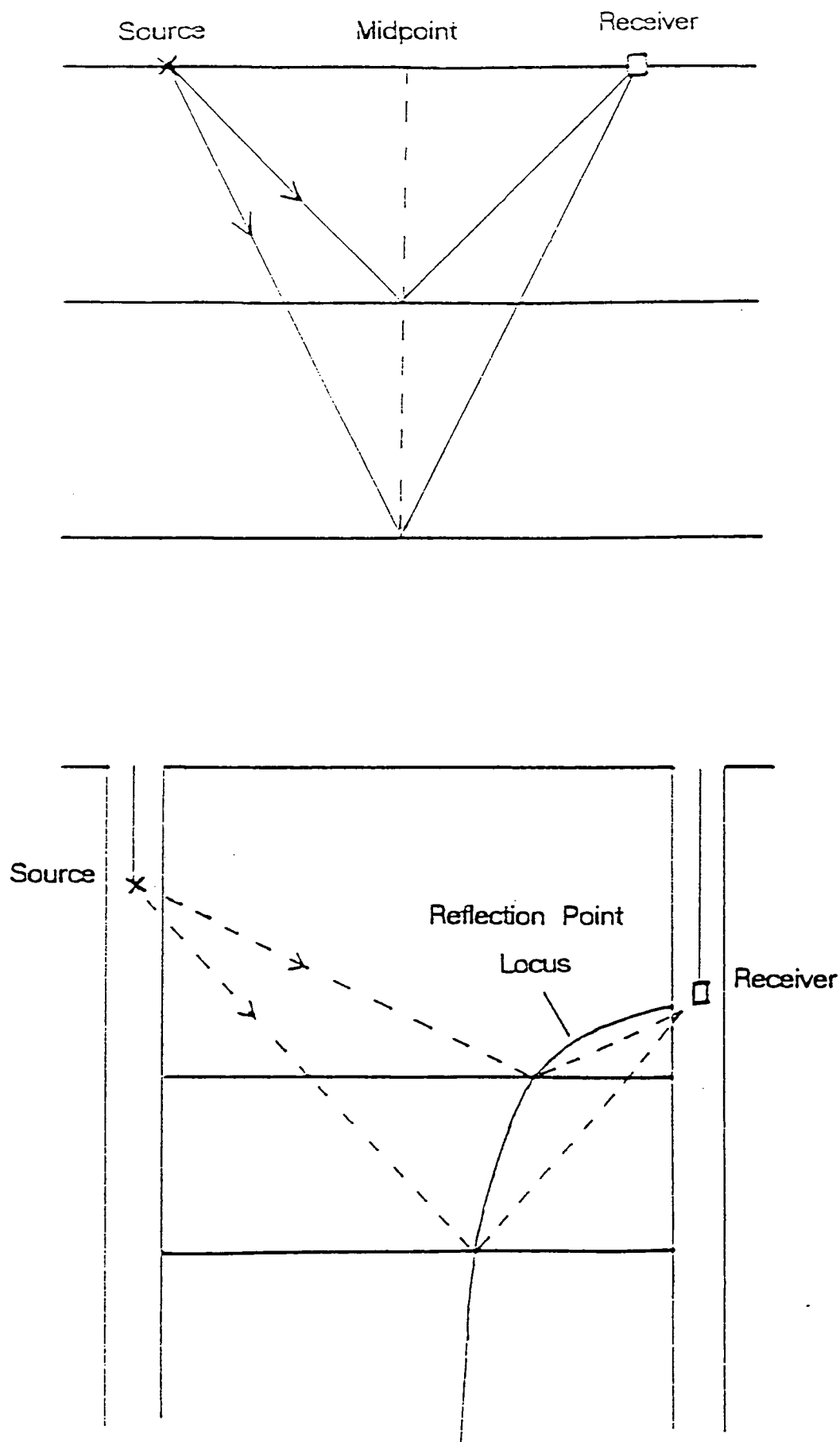


Figure 3.21 A comparison of the common midpoint of surface seismics and the reflection point locus of the cross-hole reflection method.

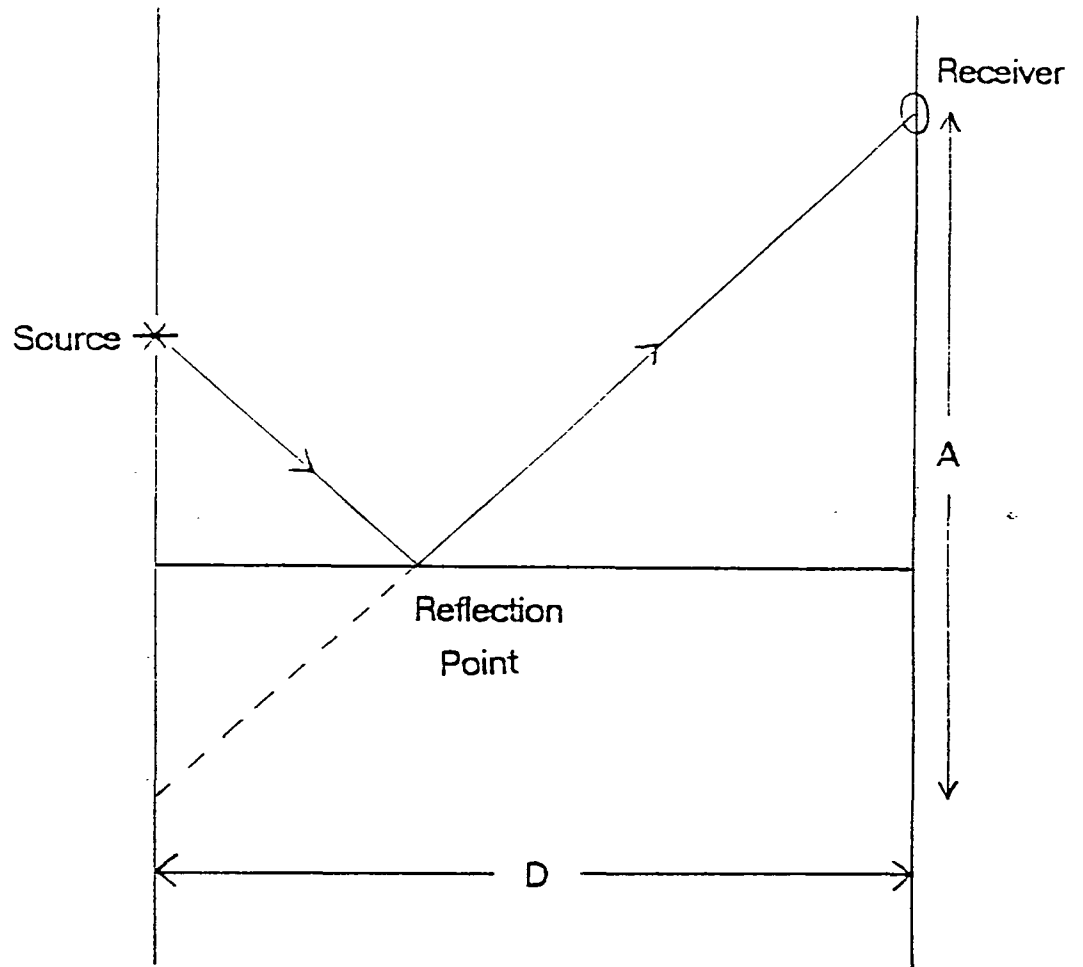


Figure 3.22 Reflection point geometry assuming a uniform velocity field.

vertical traveltime). The method involves considering each source and receiver pair independently. It is a single-channel process, not a true migration method, and therefore will not collapse Fresnel zones. The locus may be calculated by raytracing reflection raypaths between the source and the receiver over a range of take-off angles. This simultaneously provides the reflected ray traveltime, T_r for each point calculated on the locus. If a constant, isotropic velocity field and vertical boreholes are assumed (figure 3.22), then this traveltime is given by

$$VT_r = \sqrt{A^2 + D^2} \quad (3.2)$$

where V is the constant velocity,

D is the borehole separation,

and A is the total vertical distance travelled by the raypath.

The next processing stage is to distribute the appropriate trace for this source-receiver combination along the locus, by putting the signal amplitude recorded at a particular time T_{rec} at the point on the locus which has the corresponding reflection time $T_r = T_{rec}$. Interpolation is carried out to ensure that the data are not undersampled in the imaging space. The mapping can then be applied to all the traces in a common shot gather, and to all the common shot gathers in a survey.

Two programs are used to apply the VSP-CDP transform. REFLOC (Appendix A.2) calculates the reflection point loci from the given parameters of the source and receiver locations and the velocity model. The program allows velocity models which consist of plane parallel layers to be specified. It has been further modified by Kragh (1990) so that the layers may exhibit the elliptical approximation to transversely isotropic media. Raytracing is carried out by shooting a ray from the source towards the receiver borehole, and shooting a ray with the same ray parameter

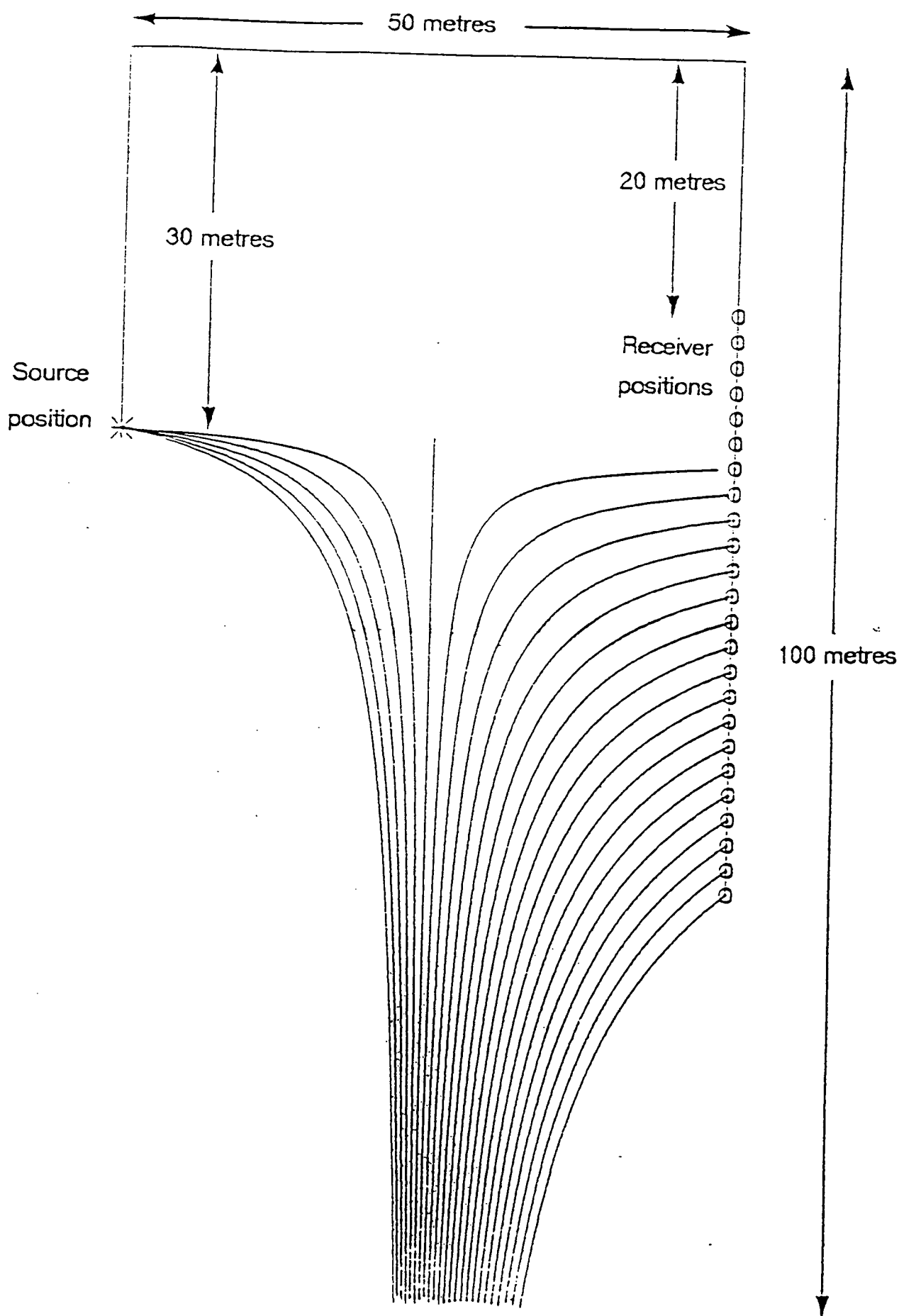


Figure 3.23 Reflection point loci for a common-shot gather assuming a uniform velocity field with recording at 24 receiver positions. The source is located in the left borehole at a depth of 30m and there are 24 receivers at 2m depth intervals ranging from 20m to 66m in the right borehole.

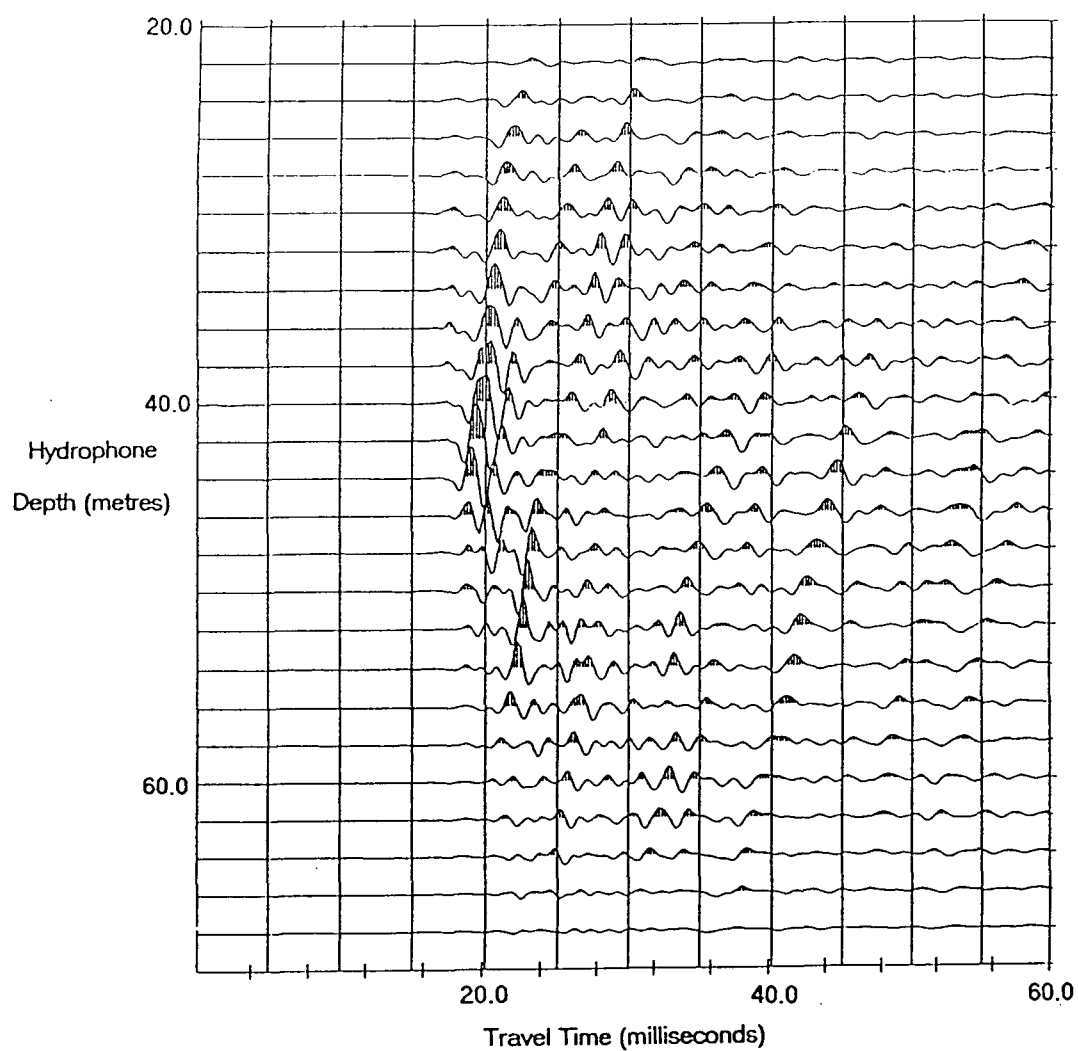


Figure 3.24 The upgoing wavefield of a common shot gather from survey A of chapter V.

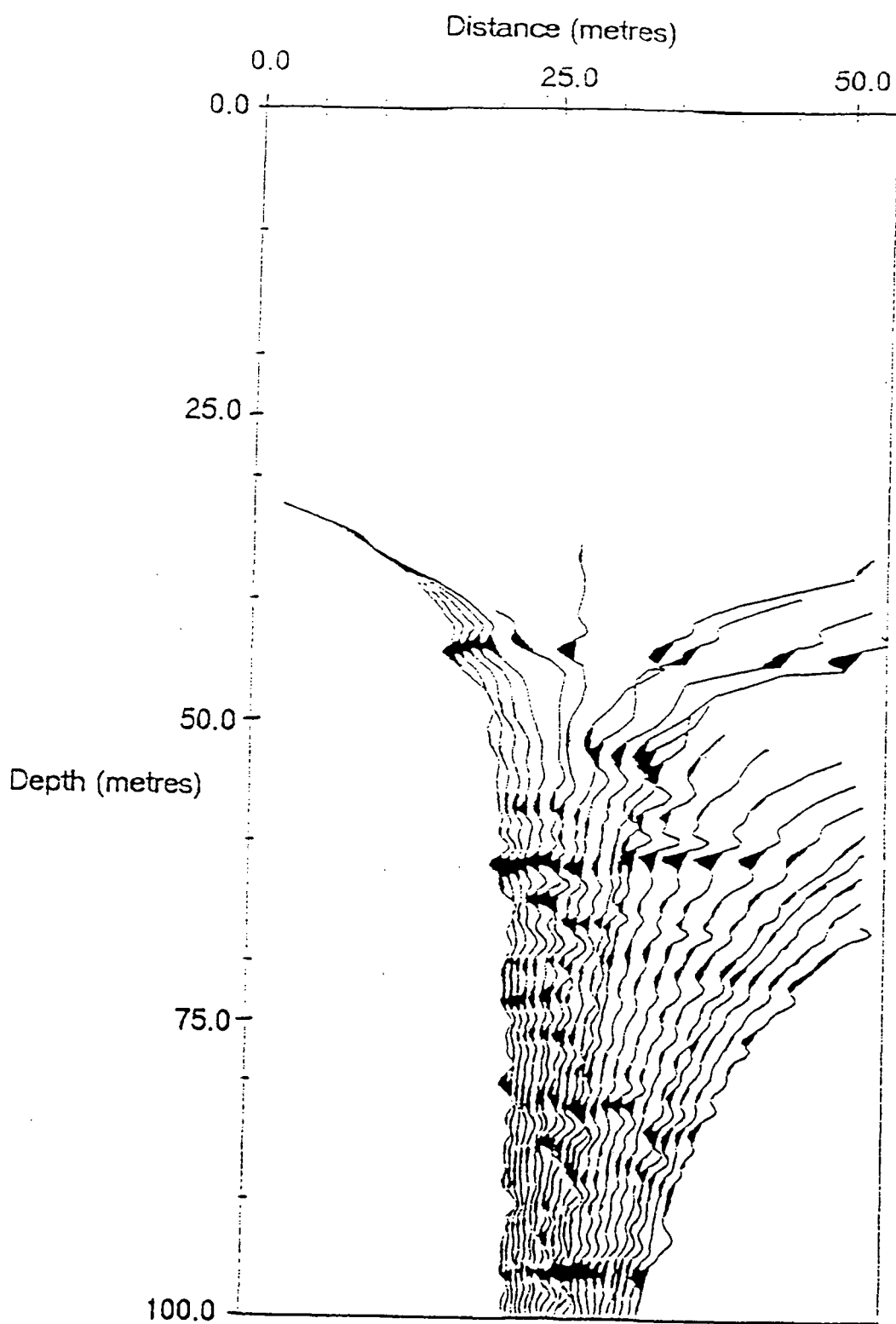


Figure 3.25 Upgoing wavefield of figure 3.24 mapped into a depth section using the VSP-CDP transformation.

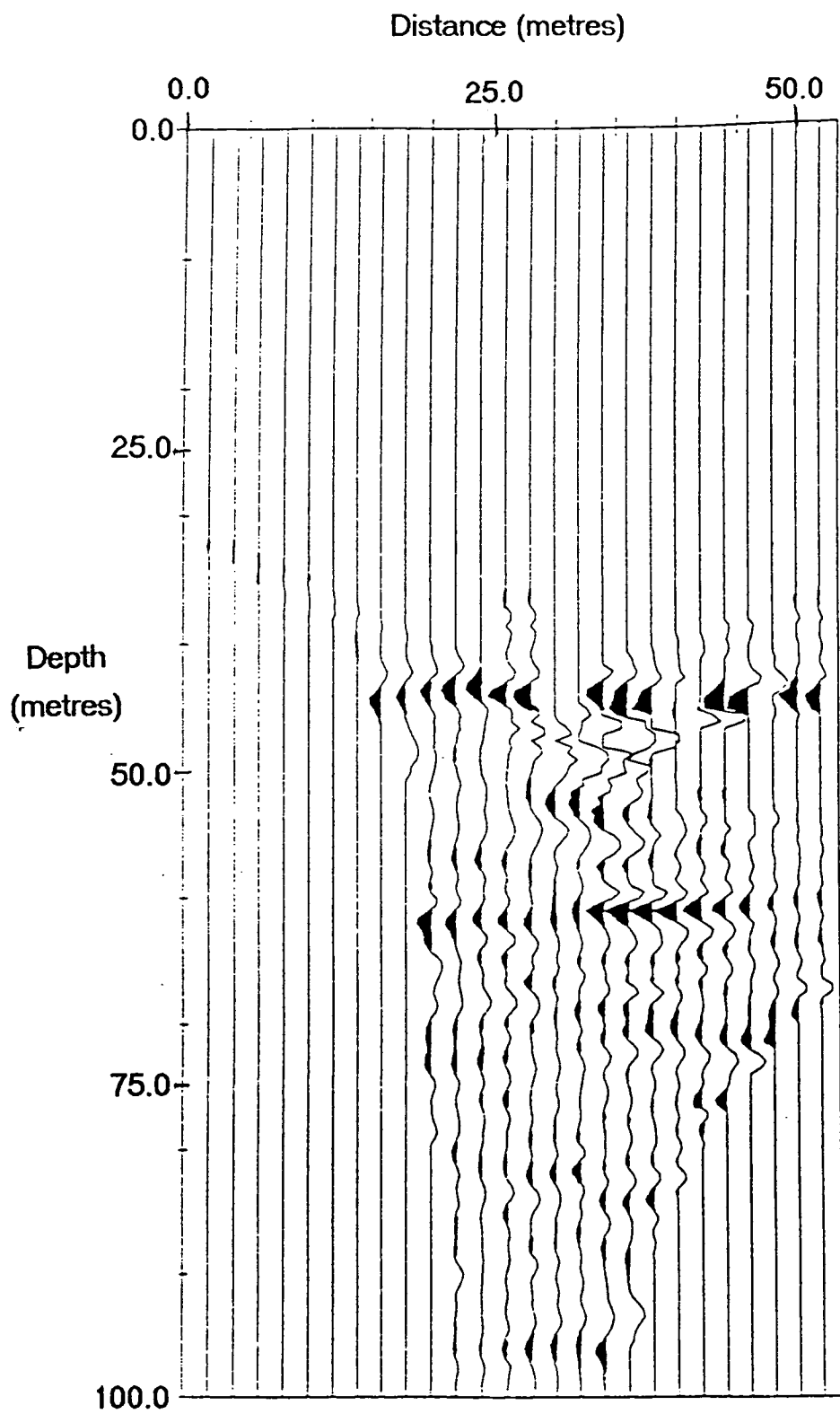


Figure 3.26 Depth section of figure 3.25 following binning on to a discrete grid.

(i.e. $\sin\theta / V$ where θ is the angle between the ray and the vertical, always taken as positive) from the receiver towards the source borehole. The point where these rays intersect will lie on the reflection point locus.

The program XHRMAP (Appendix A.3) is a menu-driven program which distributes the reflection data on to the loci and samples the mapped section on to a regular grid. The sampling is done by spreading each amplitude on each locus into its four nearest grid points with a weighting factor proportional to $(1-x_i)$ where x_i is the normalised distance of the locus point from grid point i . With the data for all the common shot gathers binned on to a regular grid, the separate sections may be stacked together, with a normalisation factor which depends on the number traces contributing at a grid point. Other program options include the ability to mute out data contributions which correspond to near-horizontal raypaths, where a relatively small error in the velocity field would lead to a significant positioning error in the locus, and the option of previewing CDP gathers (i.e. the contributions of each shot to the stack at a particular offset).

Examples of real data undergoing the VSP-CDP transform are illustrated in figures 3.23-3.26. The data form part of survey A in chapter V. Figure 3.23 illustrates the field arrangement. The source was located at a depth of 30m and 23 receivers were positioned at 2m intervals between 22m and 66m depth at an offset of 56m. The upgoing wavefield of the common shot gather is shown in figure 3.24. This has been mapped on to reflection point loci in figure 3.25 and then resampled on to a regular grid in figure 3.26.

The reflection point mapping technique also gives an indication of the type of subsurface coverage we might expect from a cross-hole reflection survey. Figure 3.27 is a plot of all the (upgoing) reflection loci for a survey in which the boreholes

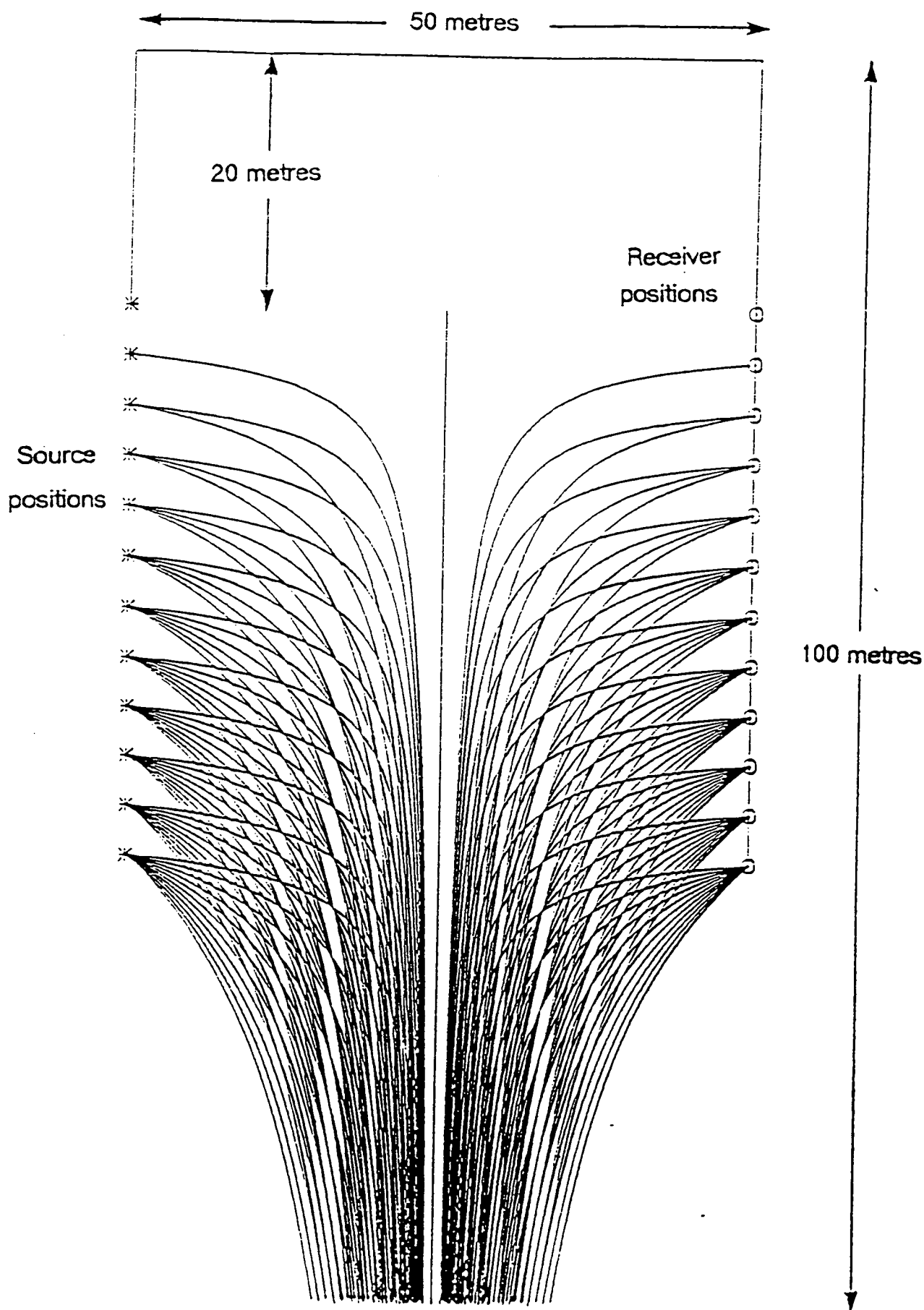


Figure 3.27 Reflection point coverage for an acquisition geometry with sources located in the left borehole and receivers in the right borehole. A uniform velocity field has been assumed. Source and receiver positions are illustrated at 4m spacings for clarity.

separation is 50m, and 13 sources and receivers are positioned at 4m intervals in the boreholes. This emphasises the funnel-shaped nature of the reflection point coverage obtained from cross-hole surveys, with no coverage at all immediately below the bottom source and the bottom receiver. The low density of reflection points near the boreholes means that the number of contributions to the stack is lower there and consequently the image quality will be poorer. Furthermore, muting direct arrivals will remove the corresponding earliest segment of each reflection point locus.

3.7 Velocity analysis using the VSP-CDP transform

Prior to stacking, the reflection point mapping technique may also be used as a velocity analysis tool by inspecting CDP gathers (i.e. stack contributions from each source position for a particular offset). It is not feasible to carry out a precise velocity analysis on real data in a single pass, due to the inability to perform a layer-stripping analysis when the source and receiver arrays are parallel to the direction of major velocity variations. Nevertheless, an estimate of velocity field corrections may be made by examining the moveout of reflections in CDP-space.

Let us assume that the velocity field V is constant between the boreholes. For any particular source and receiver combination (see figure 3.22), the reflection point travel time is given by equation (3.2)

$$\text{i.e. } V^2 T_r^2 = A^2 + D^2 \quad (3.2)$$

Let Z_s be the vertical separation of the source and the reflector,

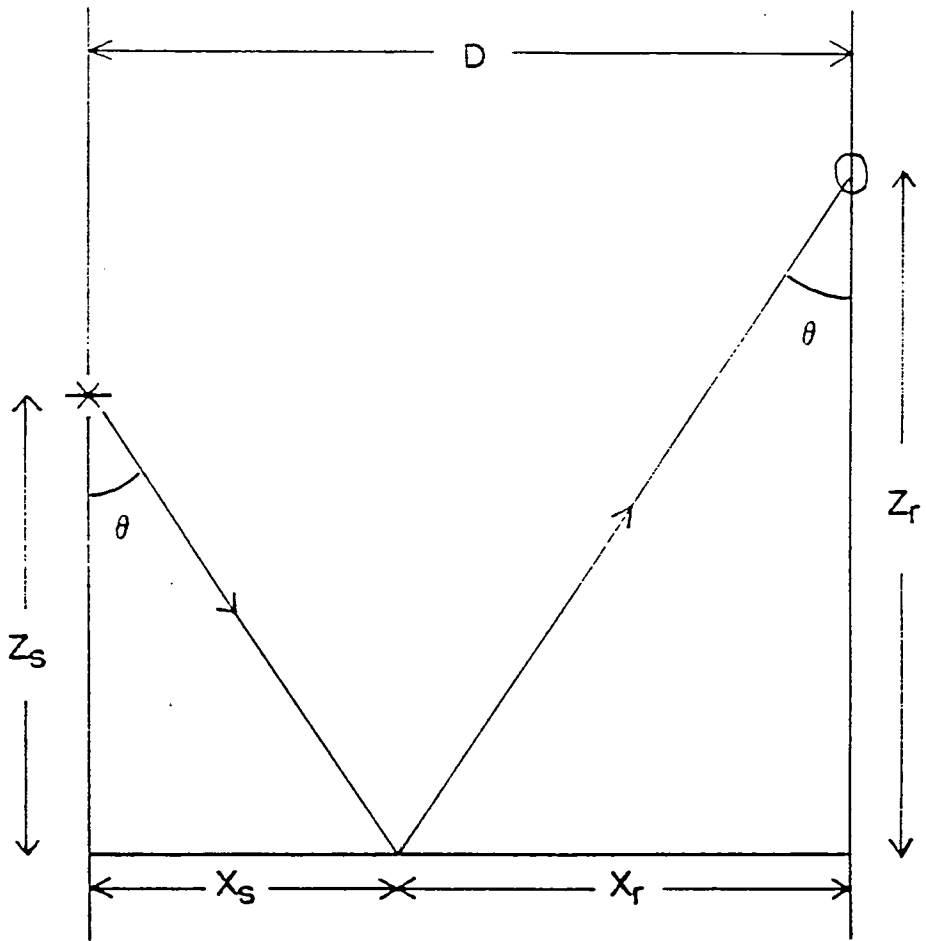


Figure 3.28 The parameters used in velocity analysis calculations.

Z_r be the vertical separation of the receiver and the reflector,
 X_s be the offset of the reflection point from the source,
 and θ be the angle which the raypath makes to the vertical.
 Therefore,

$$\frac{X_s}{Z_s} = \frac{D}{A} = \tan \theta \quad (3.3)$$

To estimate the vertical positioning error Δz which would be caused by an error Δv in the velocity field, equation (3.2) may be differentiated with respect to V , with T_r and D held constant.

$$VT_r^2 = A \frac{dA}{dV}$$

but from (3.2)

$$VT_r^2 = \frac{A^2 + D^2}{V}$$

therefore

$$A \frac{dA}{dV} = \frac{A^2 + D^2}{V} \quad (3.4)$$

Now $\Delta A = 2 \Delta z$, so we may rewrite the above expression as an approximation for small, but finite, Δv

$$\Delta z \cong \frac{1}{2} \frac{A^2 + D^2}{A} \frac{\Delta v}{V_0} \quad (3.5)$$

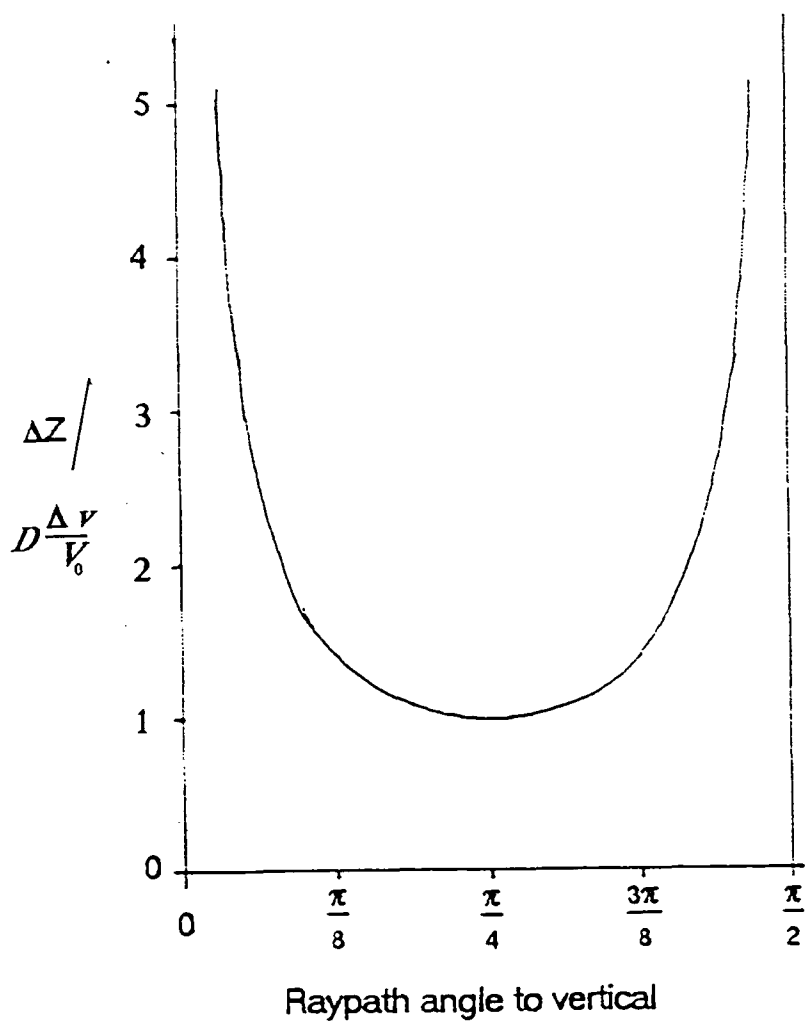


Figure 3.29 The vertical positioning error (Δz) as a function of the ray angle to the vertical.

Substituting for A from (3.3)

$$\Delta z \cong \frac{1}{2}D (\tan\theta + \cot\theta) \frac{\Delta v}{V_0} \cong \frac{D}{\sin 2\theta} \frac{\Delta v}{V_0} \quad (3.6)$$

Thus, the vertical positioning error Δz is positive for $\Delta v > 0$ and negative for $\Delta v < 0$ and the dependence on raypath angle is shown in figure 3.29. For a given velocity error, the positioning error is a minimum when the raypath makes an angle of 45° to the vertical and increases as this angle deviates from 45° . If the velocity values are too large then one would expect to see the reflector depth increase as the source depth approaches that of the reflector; if the velocity values are too small the reflector depth will decrease as the source approaches the reflector depth. This therefore provides a trial-and-error method of improving the velocity field estimate by firstly applying the VSP-CDP transform, inspecting the CDP gathers, modifying the velocity field and then re-applying the VSP-CDP transform with the new velocity field.

Great care is needed in this process, because the shots and receivers are at different depths. The raypaths for contributions to the stack at a particular image point will thus be subject to different corrections when the layer velocities are modified. A change in a particular layer velocity will (in the case of upgoing reflections) affect only those CDP contributions from shots and receivers which are above or within the layer.

It is also useful to appreciate the lateral positioning error Δx which arises from a velocity field error.

From equation (3.3)

$$X_s = Z_s \tan\theta = Z_s \frac{D}{A}$$

Differentiating with respect to V ,

$$\frac{dX_s}{dV} = \frac{D}{A} \frac{dZ_s}{dV} - Z_s \frac{D}{A^2} \frac{dA}{dV} \quad (3.7)$$

Applying $2dZ_s = dA$ and $Z_s = \frac{AX_s}{D}$

$$\frac{dX_s}{dV} = \frac{1}{A} \left(\frac{D}{2} - X_s \right) \frac{dA}{dV}$$

$$= \left(\frac{D}{2} - X_s \right) \frac{A^2 + D^2}{A^2 V} \quad \text{using (3.4).}$$

For small, but finite Δv we may write this as the following approximation using (3.3).

$$\Delta x \cong \frac{D}{2} \left(1 - 2 \frac{X_s}{D} \right) \frac{\Delta v}{v_0} \sec^2 \theta \quad (3.8)$$

Thus Δx is zero at $X_s = \frac{1}{2}D$ (i.e. no lateral moveout at the midpoint of the boreholes) and increases to $\frac{1}{2} \frac{D}{V} \sec^2 \theta$ (see figure 3.30) at the boreholes. One would thus expect that the image quality will be much better towards the centre of a cross-

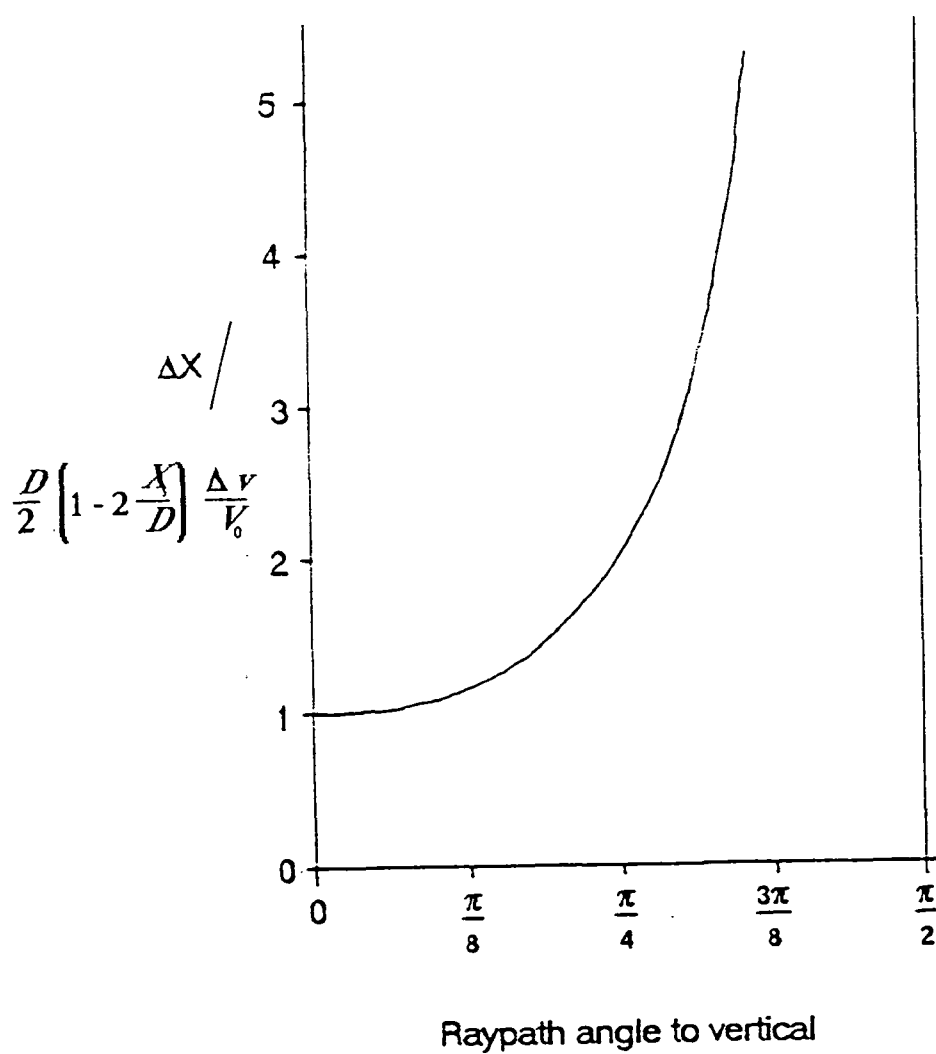


Figure 3.30 The lateral positioning error (Δx) as a function of raypath angle to the vertical.

hole reflection survey than it is nearer the boreholes, because of the increased number of stack contributions as well as the lower sensitivity of these areas to lateral positioning errors.

Once a sufficiently accurate velocity model has been obtained through the combination of experimental measurements and velocity analysis via the VSP-CDP transform, the imaging of the data can be carried out by applying migration techniques. These are discussed in chapter IV.

Chapter IV

Migration of cross-hole reflection data

4.1 Introduction

The migration of seismic data is the means whereby reflection events are relocated to their true subsurface positions and diffraction events are collapsed towards a point. Surface seismic data are generally migrated post-stack. The stacked data are assumed to be the equivalent of a zero-offset (i.e. coincident sources and receivers) section. Data will not be migrated correctly when normal moveout corrections result in a section which is not truly zero-offset, such as when conflicting reflector dips are present at the same traveltimes. To get around NMO problems pre-stack migration may be used, but this is an expensive processing procedure because the data-volume is many times greater before stacking than it is after stacking. With cross-hole seismic surveys the data-volume is relatively small and so pre-stack migration methods can be more readily applied.

Two different two-dimensional migration schemes have been implemented in this work: a finite-difference method and a Kirchhoff integral method. Both algorithms were implemented specifically for cross-hole recording geometries, but with subsurface sources and receivers being the most general case, the programs will also migrate VSP and surface seismic profile data.

4.2 Finite-difference migration

A finite-difference modelling scheme for OVSP data has been developed in the (x,z,t) domain by McMechan (1985). Migration schemes based on this method have been applied to real OVSP data by Chang and McMechan (1986) and to synthetic and scale-model cross-hole data by Hu et al. (1988). In these cases the data were migrated pre-stack in common shot gathers and an acoustic migration scheme was used. Sun and McMechan (1986) have applied an elastic finite-difference algorithm to image synthetic OVSPs and Zhu and McMechan (1988) have migrated both acoustic synthetic and scale-model cross-hole data after stacking by assuming that the sources and receivers could be combined and treated as vertical planes lying perpendicular to the plane of the boreholes.

The algorithm applied here takes the upgoing or downgoing reflection energy from each common shot gather and drives a two-dimensional wavefield on a regular grid by using the traces as boundary conditions to solve the two-dimensional acoustic wave equation. By repeatedly stepping backwards in time, the wavefield $P(x,z,t)$ between the boreholes can be found for all image points at all relevant times. The relevant time at each image point is selected according to the imaging condition as defined by Claerbout (1971). Reflectors exist at points in the subsurface where the first arrival of the source-emitted wave is time-coincident with a scattered wave. Thus the time sample of interest at a particular image point (x_i, z_i) corresponds to the excitation time T_s of the point by a direct arrival wave emitted from the source position. The excitation time may be obtained by raytracing through a velocity model between the source position and the image point. When the wavefield is back-propagated in time, it is a simple matter to extract the appropriate amplitude of the wavefield $P(x_i, z_i, T_s)$ and to store it in an array which represents the imaged data.

This technique is known as reverse-time finite-difference migration with the excitation-time imaging condition.

4.3 Implementation of the finite-difference migration algorithm

The migration method involves two separate stages. Firstly rays are traced from the source to each imaging point in order to obtain the excitation time of the image points. The excitation times are then used as input parameters to the finite-difference migration program which back-propagates the recorded wavefield.

Two raytracing algorithms have been used. The first is the program TRACER (listed in Appendix A.4) which uses the ray-equation method of Lee and Stewart (1981) and which allows lateral velocity variations to be taken into account. This method was found to be too computationally expensive and so a faster two-point iterative algorithm was also developed (subroutine RAYTRA in Appendix A.5). RAYTRA ignores lateral velocity variations, but does allow transversely isotropic media to be modelled by means of an elliptical velocity approximation (Levin, 1978). The RAYTRA program was used in preference to TRACER because of its speed, particularly because lateral velocity variations between boreholes in a cross-hole survey could not be determined with sufficient confidence to justify using a laterally variant velocity structure.

The excitation time for image reconstruction is obtained by raytracing from the source location to each point in the image space. The traveltimes along the appropriate raypaths are stored in a data file for future reference by the imaging algorithm. The reverse-time wavefield extrapolation program EXTRAPREV is listed

in Appendix A.6. A similar program EXTRAP is listed in Appendix A.7. EXTRAP is a forward modelling program which allows wavefields to be propagated from a source point with a given time-dependent signature and allows the wavefield at particular receiver positions to be recorded, thus generating synthetic seismograms.

The EXTRAP programs use an algorithm which carries out a second order finite-difference formulation of the acoustic wave equation. This is a two-dimensional (2-D) modelling algorithm which allows sources and receivers to be positioned at any subsurface (or surface) location in a 2-D velocity field. The finite-difference algorithm corresponds to that of McMechan (1985) which was written for offset VSPs, but with the additional provision of a variety of absorbing boundary conditions as discussed by Renaut and Petersen (1989). The programs allow the user to specify a laterally variant isotropic velocity field sampled on a rectangular grid and then the pressure response of the medium to a change in pressure with time (such as from a point source or from boundary conditions provided by a receiver array) is obtained by solving the acoustic wave equation in the time domain.

4.3.1 The finite-difference approximation

The 2-D acoustic wave equation is (Claerbout, 1976)

$$\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial z^2} = V^2(x,z) \frac{\partial^2 P}{\partial t^2} \quad (4.1)$$

where P is the acoustic response and V is the velocity of acoustic waves in the medium. The solution of equation (4.1) is achieved by a second order finite-difference approximation with discretisation in time and space. A fourth order

approximation (Alford et al., 1974) was tried but resulted in no noticeable improvement in imaging results. Three wavefields are involved at any particular time step t_i . These are

$P(x,z,t_i)$ the wavefield at time t_i ;

$P(x,z,t_{i-1})$ the wavefield at time t_{i-1} , the previous step; and

$P(x,z,t_{i-2})$ the wavefield at time t_{i-2} .

The acoustic response at any particular grid point (x_k, z_j) may be defined as $P(x_k, z_j, t_i)$ where (x_k, z_j) is the intersection of the j^{th} horizontal grid line with the k^{th} vertical grid line.

The finite difference approximations to partial differentiations with respect to x are given by

$$\frac{\partial P}{\partial x} \cong \frac{P_{k+1} - P_k}{\Delta x} \quad (4.2)$$

$$\frac{\partial^2 P}{\partial x^2} \cong \frac{P_{k+1} - 2P_k + P_{k-1}}{\Delta x^2} \quad (4.3)$$

where Δx is the grid spacing in the x -direction. Thus equation (4.1) may be approximated by equations like (4.3) and rearranged to give a value of $P(x_k, z_j, t_i)$ which depends solely on the values of $P(x_k, z_j)$ calculated at times t_{i-1} and t_{i-2} .

$$\begin{aligned}
 P(x_k, z_j, t_i) = & 2(1-2A^2) P(x_k, z_j, t_{i-1}) - P(x_k, z_j, t_{i-2}) + \\
 & A^2 [P(x_{k+1}, z_j, t_{i-1}) + P(x_{k-1}, z_j, t_{i-1}) + P(x_k, z_{j+1}, t_{i-1}) + \\
 & P(x_k, z_{j-1}, t_{i-1})]
 \end{aligned} \tag{4.4}$$

where $A = V(x_k, z_j)\Delta t/h$, Δt is the time step ($\Delta t = t_i - t_{i-1}$) and h is the distance between both horizontal and vertical grid lines. This algorithm has been shown to be stable (Mitchell, 1969) provided that the following condition is met.

$$\Delta t < \frac{h}{\sqrt{2}V} \tag{4.5}$$

4.3.2 Boundary conditions

A computational problem occurs at the edges of the finite-difference grid. If values of zero are used to define the acoustic response at the edges of the grid then a free surface boundary condition (zero pressure) has been created and outgoing wavefields will be reflected back into the grid. Methods of avoiding this problem include enlarging the grid size, a computationally expensive choice, or applying absorbing boundary conditions (Clayton and Engquist, 1977) at the edges of the grid. Absorbing boundary conditions are essentially a finite-difference approximation to the one-way or paraxial wave equation (Claerbout, 1976). A dispersion relation which

restricts the range of propagating rays to a cone around the z-axis is (Clayton and Engquist, 1977)

$$\frac{Vk_z}{\omega} = 1 - \frac{1}{2} \left(\frac{Vk_z}{\omega} \right)^2 + O \left(\left| \frac{Vk_z}{\omega} \right|^4 \right)$$

which translates to the following differential form.

$$\frac{\partial^2 P}{\partial z \partial t} + \frac{1}{V} \frac{\partial^2 P}{\partial t^2} - \frac{V}{2} \frac{\partial^2 P}{\partial x^2} = 0 \quad (4.6)$$

The above equation is an approximation which would allow a wave to propagate in the positive z direction only. Renaut and Petersen (1989) have derived several different finite-difference approximations similar to (4.6) as follows :

$$\frac{\partial^2 P}{\partial z \partial t} + \frac{p_0}{V} \frac{\partial^2 P}{\partial t^2} + V p_1 \frac{\partial^2 P}{\partial x^2} = 0 \quad (4.7)$$

where p_0 and p_1 are defined according to Table 4.1.

Approximation	Pade	L^2	Chebyshev-Pade
p_0	1.000	$21\pi/64$	$10/3\pi$
p_1	-0.5	$-15\pi/64$	$-8/3\pi$

Table 4.1 Finite difference approximations

The program EXTRAP permits selective use of these boundary conditions and also allows the option of free-surface boundaries. The p_0 and p_1 parameters may be chosen as appropriate to the survey geometry. Generally, the L^2 approximation, which yields a least squares reflected amplitude over all incident angles, seemed to provide the best absorption for cross-hole data where waves may be striking the boundary at a wide range of incident angles.

4.3.3 Grid dispersion

One problem associated with the finite-difference approximation is that of grid dispersion (Alford et al., 1974). This manifests itself in the dispersion of the source waveform as it propagates through the discrete-model medium, with lower frequencies travelling faster than higher frequencies, the pulse becomes delayed, broadened and develops a ringing tail. The effect arises because of the discrete approximation to the differential wave equation by equation (4.4). Grid dispersion effects can be reduced by decreasing the grid spacing parameter h but, as well as increasing the data volume, this necessitates a reduction in the size of the time step Δt in order that the stability equation (4.5) remains satisfied. Reduction of the grid

spacing by a factor of two thus results in a four-fold increase in the data volume and an eight-fold increase in the computational time. Alford et al. (1974) have shown that dispersive effects are not significant for wavelengths greater than about 10 times the grid spacing, and this provides a useful benchmark in determining the optimal values of Δt and h to use for a given source function.

4.3.4 Modelling seismic data with EXTRAP

Two cross-hole synthetic datasets were constructed using the EXTRAP finite-difference algorithm in order to test the Kirchhoff and reverse-time wavefield extrapolation finite-difference migration methods. These models are fully discussed in section 4.5. The required input parameters are:

- Source function specification
- Grid size
- Temporal sampling interval
- Source location
- Receiver locations
- Two-dimensional velocity model of medium

Program output allows the pressure response of the medium to be viewed at any given time, as well as the generation of synthetic seismograms. These implementations of the finite-difference algorithm are for the acoustic (scalar) wave equation. It is hoped that future development work may be carried out to enhance this modelling method in order to include anisotropy and elastic wave propagation.

4.3.5 Migrating seismic data with EXTRAPREV

EXTRAPREV functions in a similar manner to EXTRAP, but instead of providing a source function to the algorithm, the data recorded at the receiver positions is time-reversed and then input to the finite-difference wavefield at all receiver grid points simultaneously at the appropriate time samples. The drawback of this approach is that it is necessary to match the receiver spacing to the grid size and so a certain amount of spatial interpolation is necessary to achieve this (since a relatively small grid spacing may be necessary in order to avoid grid dispersion effects). Reduction of the grid spacing may also force an interpolation in time in order that the stability requirement (4.5) is met. This results in a substantial increase in computing time. Examples of this migration on synthetic model data are discussed in section 4.5.

.. .

4.4 Kirchhoff migration

A Kirchhoff migration technique was also developed. This is an extension of the diffraction stack migration approach which is readily understood from geometrical considerations.

4.4.1 Diffraction stack migration

The diffraction stack migration method was one of the first migration types to be applied to seismic data. It follows simple ray and wavefront theory. In the case of



Figure 4.1 Locus of possible reflector locations for an impulsive arrival at time T for a particular source and receiver combination.

zero-offset (or post-stack) surface seismic data, diffraction stack migration is implemented by summing along hyperbolic trajectories and placing the results at the apices of the hyperbolas. For cross-hole data, however, the summation operator is more complicated.

Consider a particular source-receiver combination with a single impulsive arrival recorded at time T_g and assume that the velocity of the medium is uniform and isotropic. The locus of possible reflection points for a particular traveltimes must be the ellipse in image space as shown in figure 4.1 with the source and the receiver at the focal points. Migration may be accomplished by distributing the recorded amplitudes along the appropriate ellipses for each trace in a common shot gather. Constructive imaging takes place where ellipses intersect at a diffracting point.

Alternatively, each imaging point may be considered in turn. Raytracing from the source position to the point and from the receiver position to the point provides a total traveltimes for scattering from the image point with the particular source-receiver combination. The amplitude corresponding to this traveltimes is then summed into the imaging array together with similar contributions from other source-receiver pairings. This is the migration procedure utilised in this work.

4.4.2 The Kirchhoff operator

Kirchhoff wave equation migration is an extension of the diffraction stack imaging concept based upon Kirchhoff's integral (see French, 1975 and Schneider, 1978).

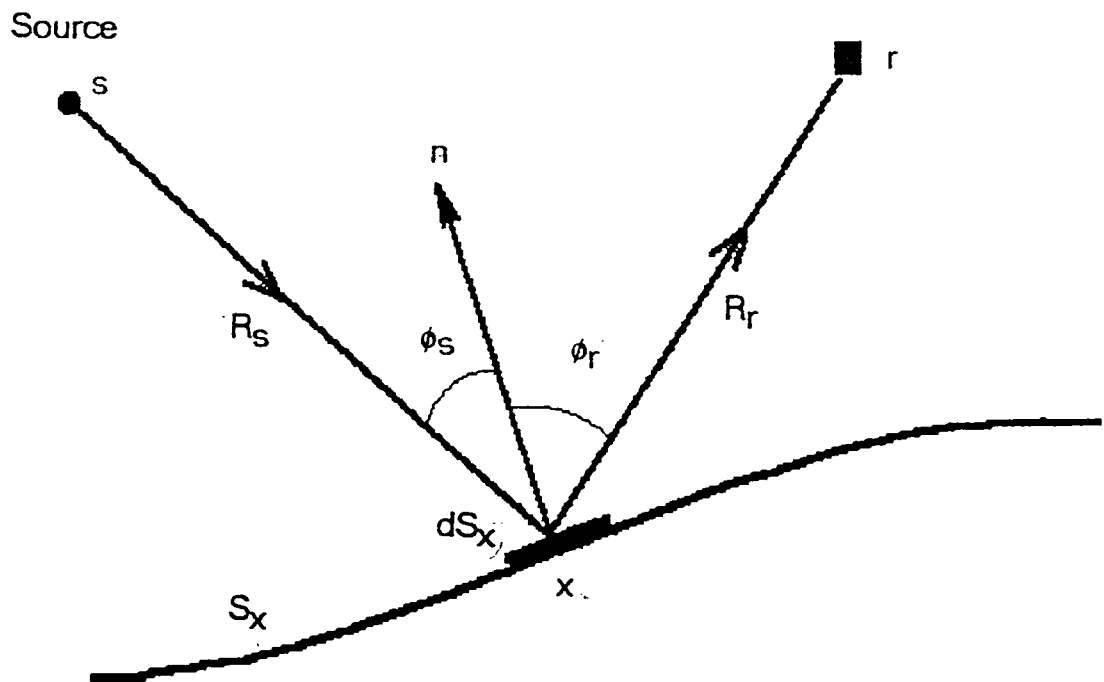


Figure 4.2 Scattering angles and raypath lengths for equation 4.8.

If we consider a wave incident on a scattering interface, then the wavefield $U(\mathbf{r}, \mathbf{s}, t)$ recorded at $\mathbf{r}$ as a result of an impulsive source at $\mathbf{s}$ being scattered from the interface S_x is given by (Dillon, 1990 equation (1)):

$$U(\mathbf{r}, \mathbf{s}, t) = \int_{S_x} dS_x \frac{C(\mathbf{x}, \phi_s)}{4\pi V} \frac{\cos(\phi_s) + \cos(\phi_r)}{R_s R_r} \delta\left(t - \frac{R_s + R_r}{V}\right) \quad (4.8)$$

The scattering angles ϕ_s and ϕ_r and raypath lengths R_s and R_r are illustrated in figure 4.2. The reflectivity $C(\mathbf{x}, \phi_s)$ is angle-dependent. Equation (4.8) is the three-dimensional response recorded at a receiver. If one assumes that the geological structure is invariant perpendicular to the plane of the survey, then by integrating along one direction, the equivalent $2\frac{1}{2}$ D formula is obtained. Dillon (1990) has derived a $2\frac{1}{2}$ D migration integral which yields the reflectivity $C(\mathbf{x})$.

$$C(\mathbf{x}) = \frac{1}{\pi} \int_{L_r} dL_r \sqrt{\frac{R_s(R_s + R_r)}{2VR_r}} \cos(\theta_r) M\left(\mathbf{r}, \frac{R_s + R_r}{V}\right) \quad (4.9)$$

where $M(\mathbf{r}, T)$ is the source wavefield $U(\mathbf{r}, T)$ following the application of a Newman half-differential filter. This filter has a spectrum with a $\frac{\pi}{4}$ phase shift and a high frequency amplitude boost of f^n and is implemented in the frequency domain. Equation (4.9) may be approximated by a summation over the receiver array L_r .

$$C(\underline{x}) = \sum_{L_T} \Delta L_T \sqrt{\frac{R_S}{R_T}} \cos(\theta_T) \sqrt{R_S + R_T} M\left(\underline{r}, \frac{R_S + R_T}{V}\right) \quad (4.10)$$

Equation (4.10) is similar to a diffraction stack with the amplitude and phase corrections of the Newman filter and a wavefront spreading correction term being incorporated. Clearly this expression is inadequate for cross-hole geometries where we should obtain similar reflectivity responses across the survey. In equation (4.10) the reflectivity will be larger where $R_S > R_T$ (i.e. nearer the receiver borehole) and an expression which is symmetrical with respect to an exchange of source and receiver arrays would be preferable.

4.4.3 The Generalised Kirchhoff operator

Dillon (1990) derived a Generalised Kirchhoff (GK) migration integral which is reciprocal with respect to sources and receivers and which is closely related to the Generalised Radon Transform (GRT) migration integral proposed by Miller et al. (1987). The 2-D form of the GK summation operator is given by

$$C(\underline{x}) = \sum_{L_T L_S} \left[\Delta L_T \sqrt{\frac{R_S}{R_T}} \cos(\theta_T) + \Delta L_S \sqrt{\frac{R_T}{R_S}} \cos(\theta_S) \right] \times \sqrt{R_S + R_T} M\left(\underline{r}, \frac{R_S + R_T}{V}\right) \quad (4.11)$$

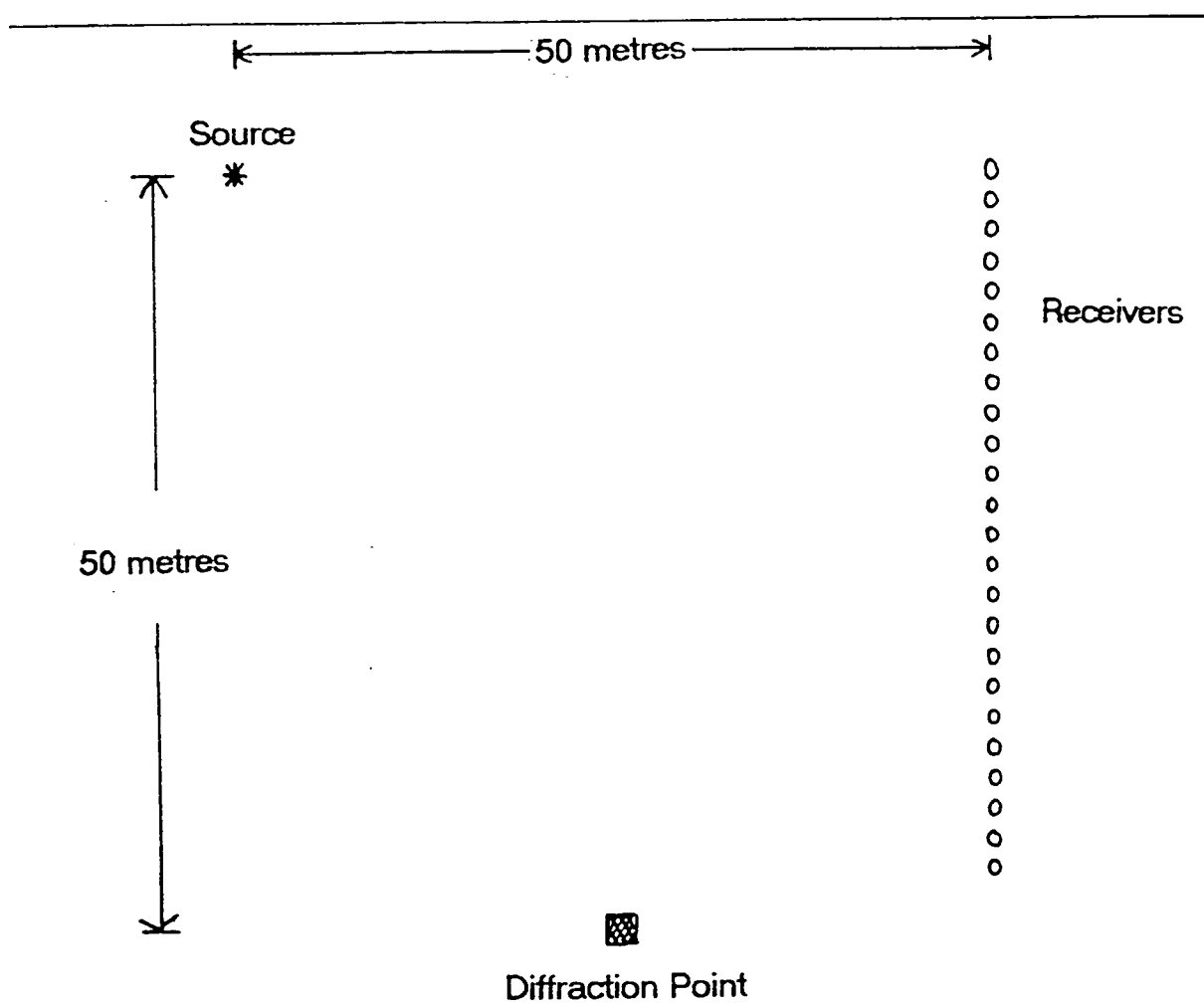


Figure 4.3 Model used to acquire synthetic dataset of point scatterer.

There is an additional small error term to the above equation which is proportional to $(\cos(\phi_S) - \cos(\phi_R))$ (Dillon, 1990), ϕ_S and ϕ_R being the scattering angles defined in figure 4.2.

The program KIRCHMIG in appendix A.8 was written to carry out diffraction stack, Kirchhoff summation and Generalised Kirchhoff summation of cross-hole data with a variety of options being variable. Amongst these is the migration aperture by which the summation locus (equivalent to a hyperbola in surface seismic migration) is spatially restricted to realistic geological dips. A very small aperture is equivalent to specular reflections and corresponds to the reflection point mapping technique (Dillon, 1988, Van der Poel and Cassell, 1989). Comparisons of the impulse responses of different migration operators are included in chapter VI.

4.5 Comparison of finite-difference and Kirchhoff migration

Synthetic datasets were created by forward modelling using the EXTRAP finite-difference method. Models of point diffractors and planar reflectors were used to study the effectiveness of the two migration techniques. In these models, no allowance for density variations was made; the only variable parameter being the structure of the velocity field.

4.5.1 Point diffractor model

The following model (see figure 4.3) was used to simulate a point diffractor. A velocity of 2500 m/s was used. The grid size h was 0.5 m, and a time step Δt of 0.1

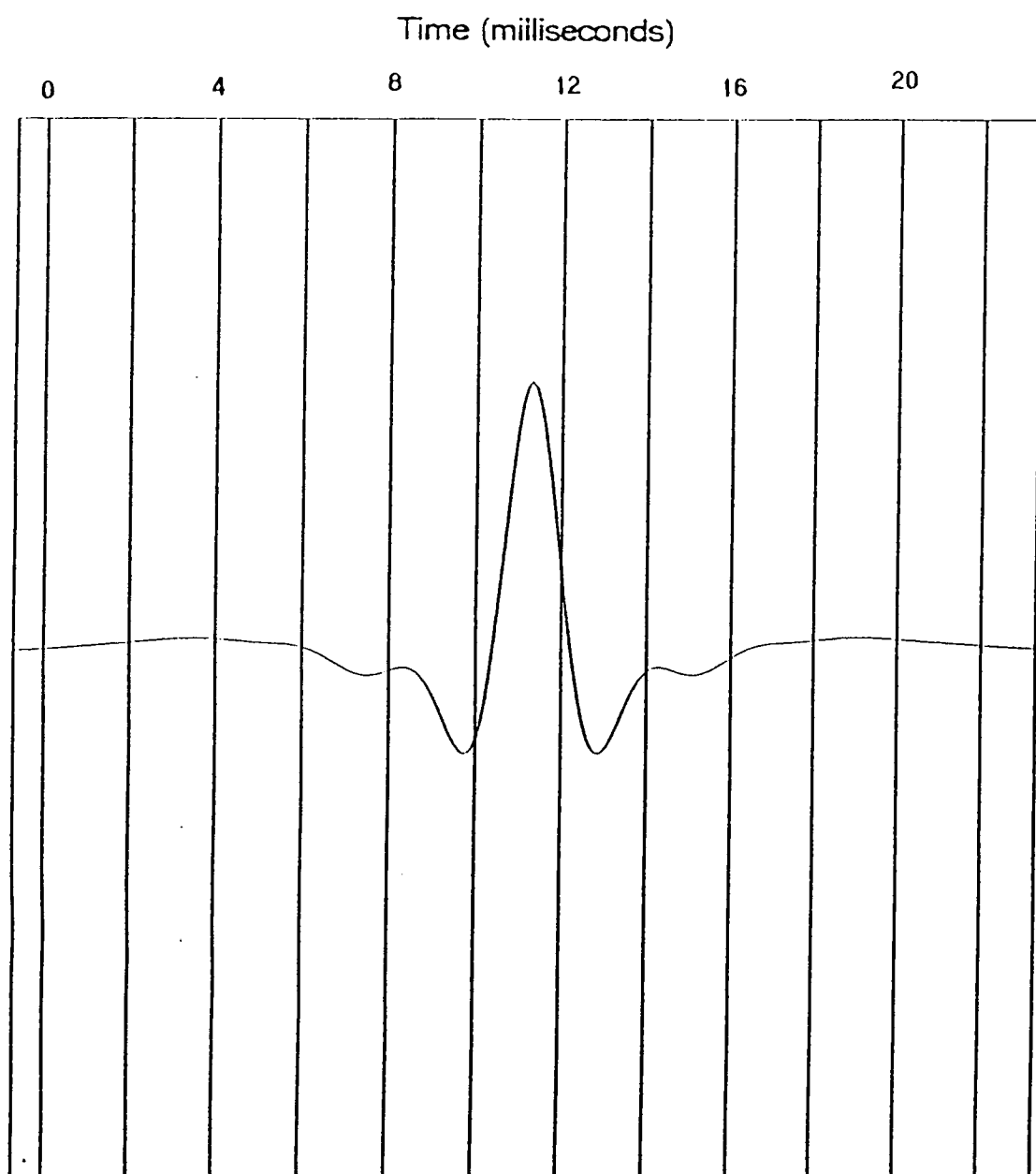


Figure 4.4 Source wavelet used in synthetic seismogram modelling.

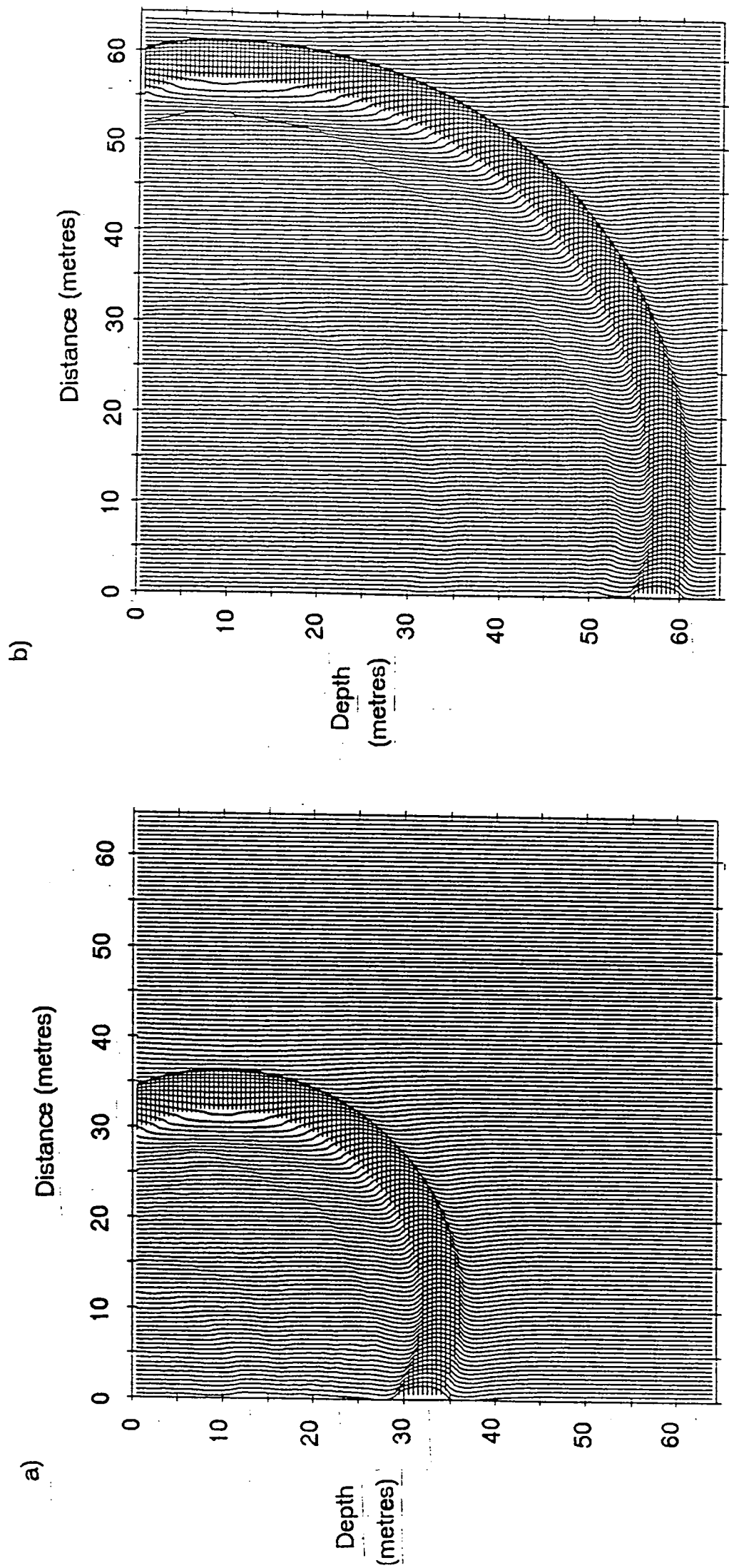
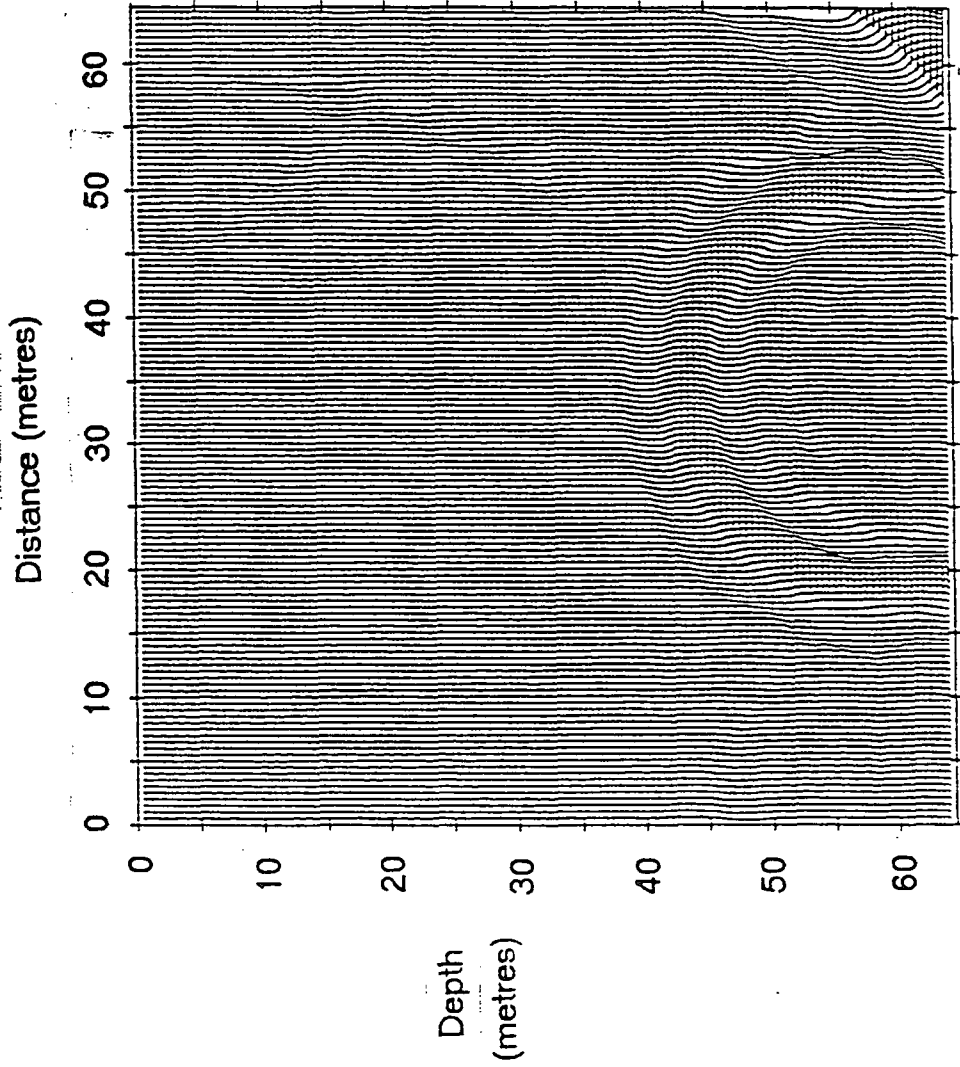
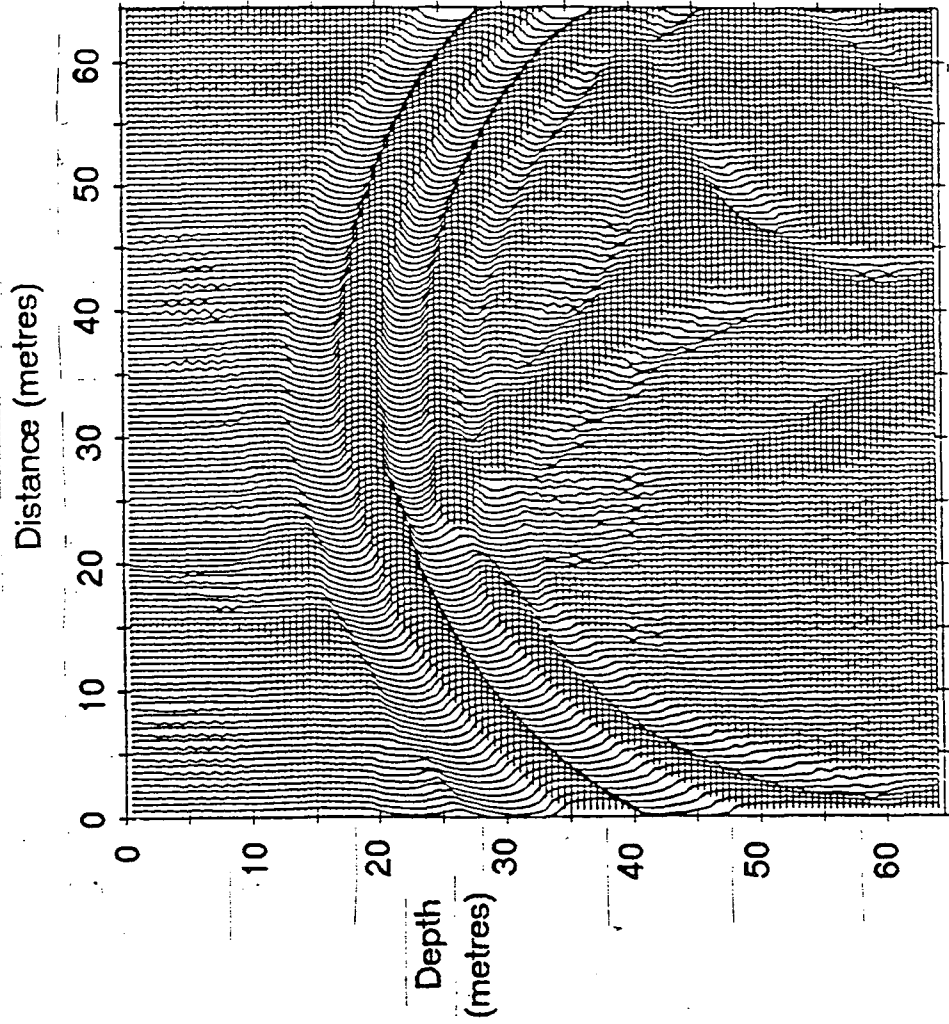


Figure 4.5 Evolution of the wavefield with time: a) 10ms, b) 20ms, c) 30ms, d) 40ms. The original source position was at (10,10).

c)



d)



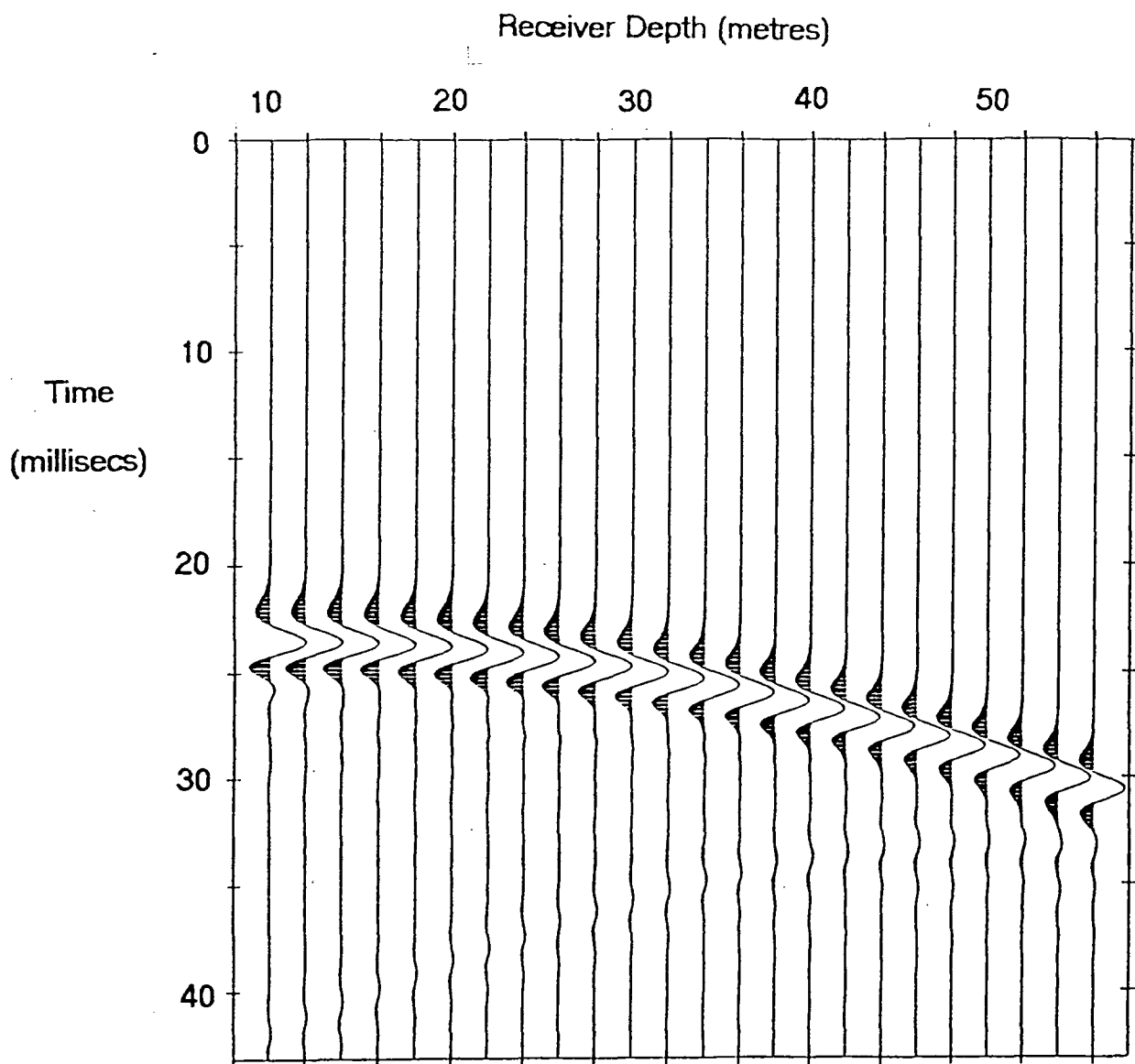


Figure 4.6 Synthetic seismogram produced for diffraction event.

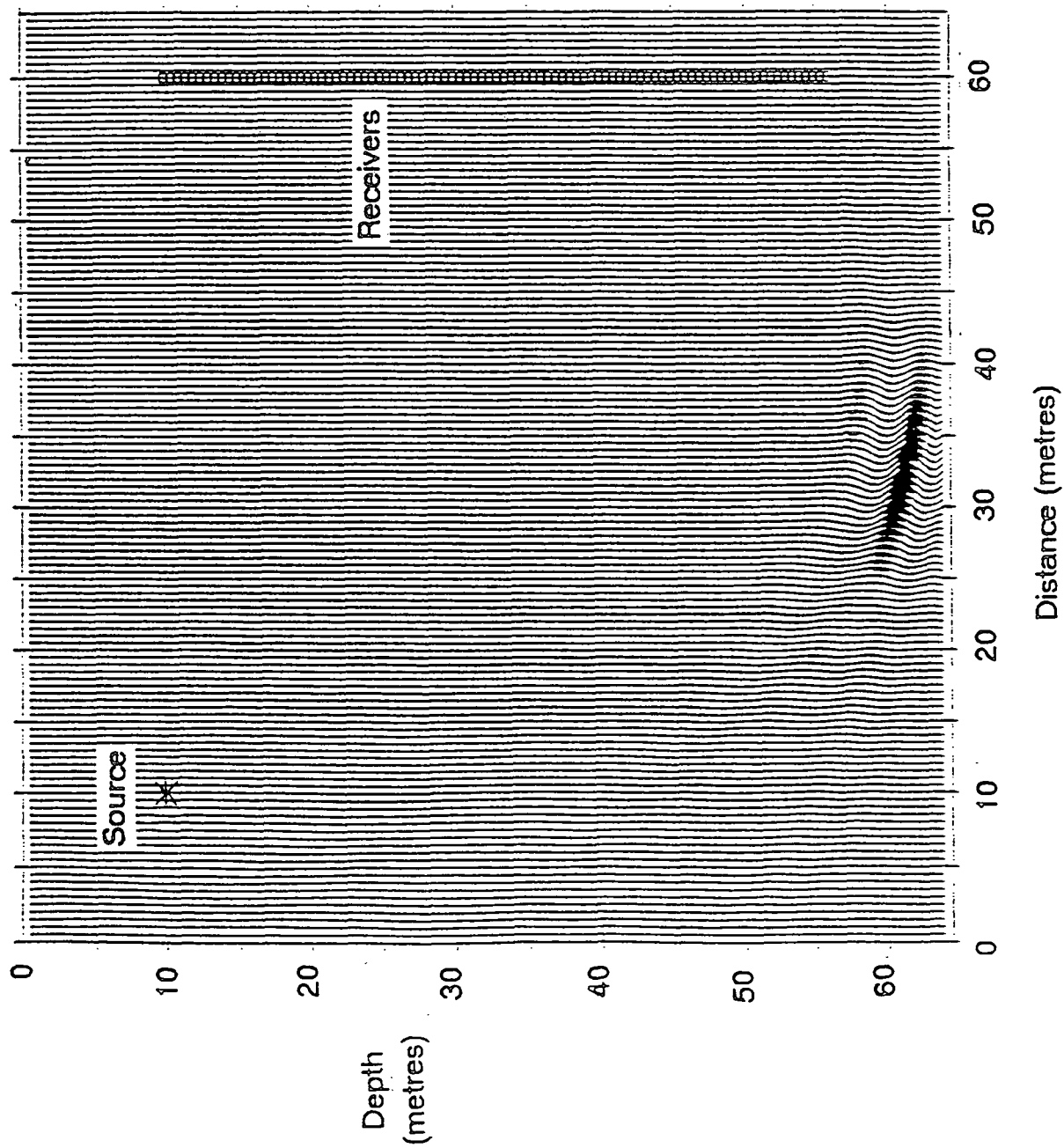


Figure 4.7 Finite-difference migrated image of the diffraction event.

ms was used. Thus the stability relation of equation (4.5) was satisfied. A Butterworth zero-phase source wavelet was used (figure 4.4) with a bandwidth of 150-500 Hz. The bandwidth was chosen to minimise the problem of grid dispersion which would occur for wavelengths greater than $10h=5\text{m}$. A square diffractor of size $2\text{m} \times 2\text{m}$ and velocity 1000m/s was positioned at the coordinates (35,60). The source was positioned at coordinates (10,10) and 24 receivers were positioned at 2m intervals, ranging in depth from 10m to 56m, at an offset of 50m from the source. The zero time of the data was set to coincide with the central peak of the source wavelet. The model was run using the L^2 absorbing boundary conditions discussed in §4.3.2. These were applied to all sides (i.e. no free surface reflections were allowed at the top of the model). Figure 4.5 illustrates the progression of the source pulse through the medium at times of 10, 20, 30 and 40 ms. Signal amplitudes have been scaled by taking the square root and retaining the polarity of the signal to enhance the diffracted energy in the diagrams. The circle of energy diffracted from the diffractor point location can be seen clearly in figures 4.5c and d. Data were calculated up to a time of 40 ms. The data "recorded" at the receiver positions are illustrated in figure 4.6. These data were then processed to remove the direct arrival. Since the data have been created by a 2-D modelling algorithm, no amplitude recovery is necessary in the preprocessing stage. The data were migrated by the finite-difference method resulting in the image of figure 4.7. This image is quite disappointing, with significant "smiles" around the diffraction point. The tilt of the migration smearing in this case is characteristic of the geometry of this survey.

Generalised Kirchhoff migration of the same data results in the image shown in figure 4.8. In this case, a full aperture (i.e. allowing all possible structural dips)

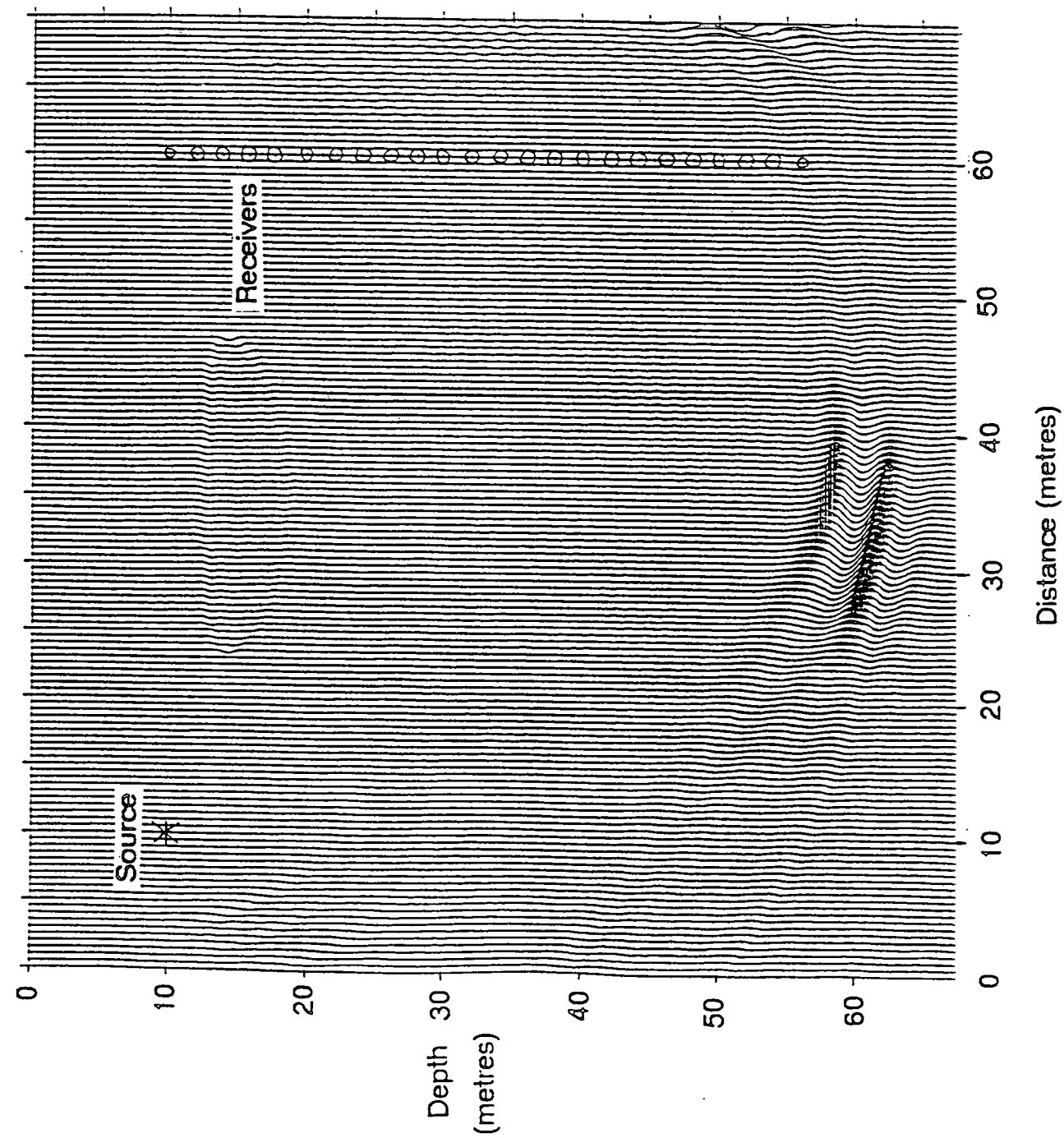


Figure 4.8 Generalised Kirchhoff migrate image of the diffraction event.

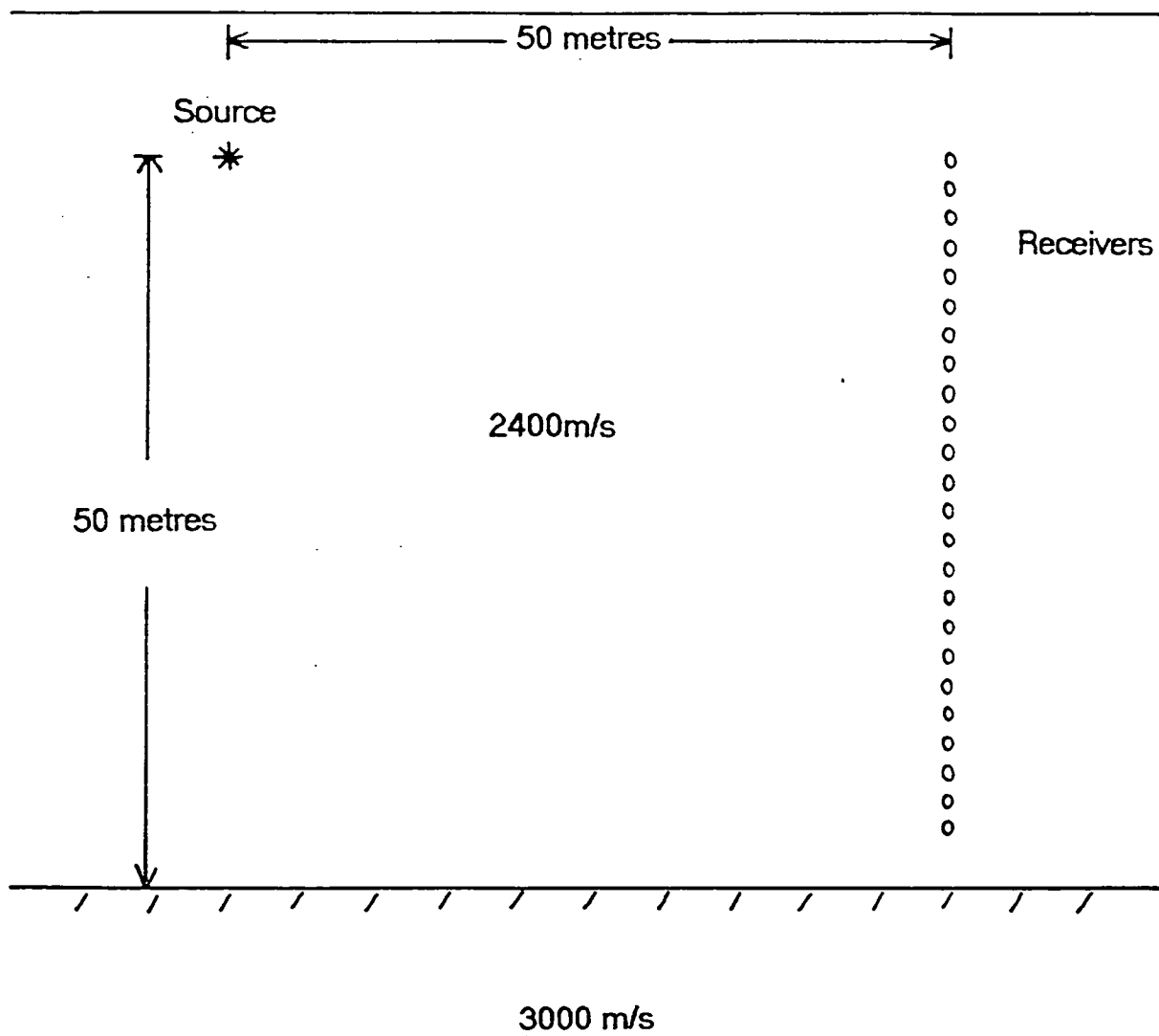
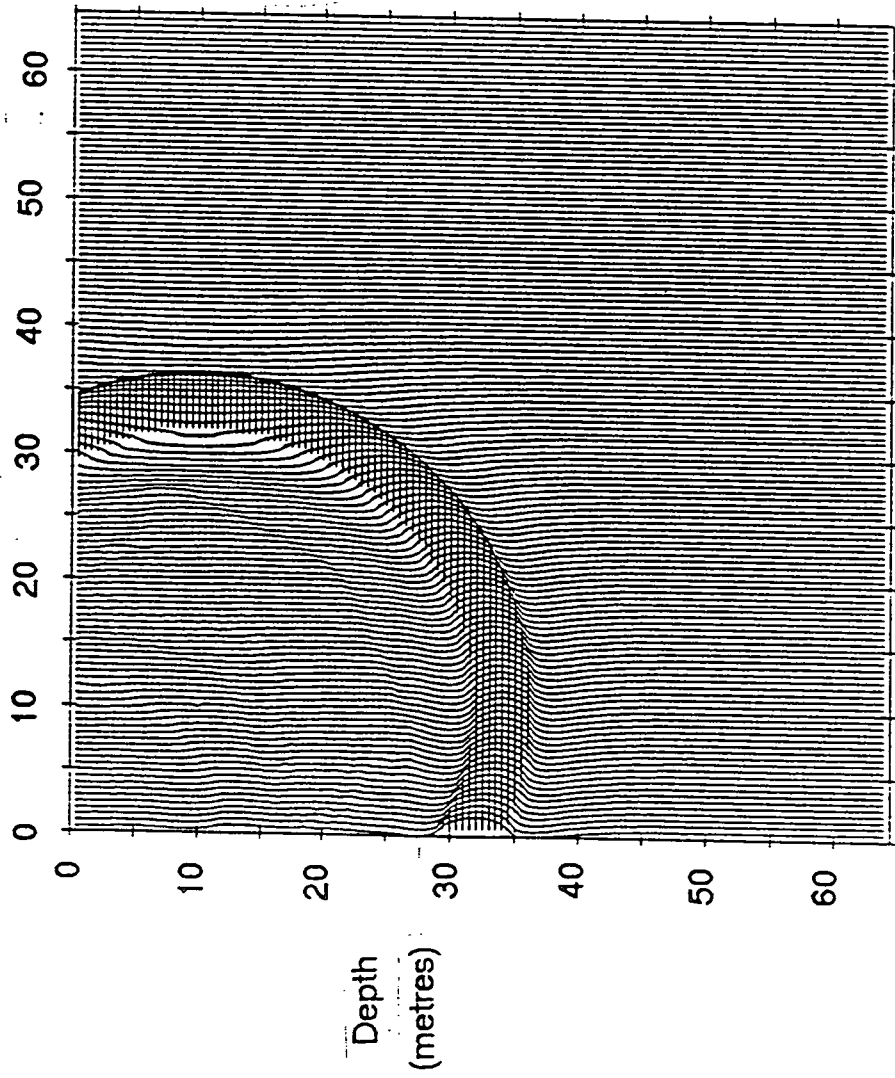


Figure 4.9 Model used to generate synthetic data for cross-hole reflectivity

a)

Distance (metres)



b)

Distance (metres)

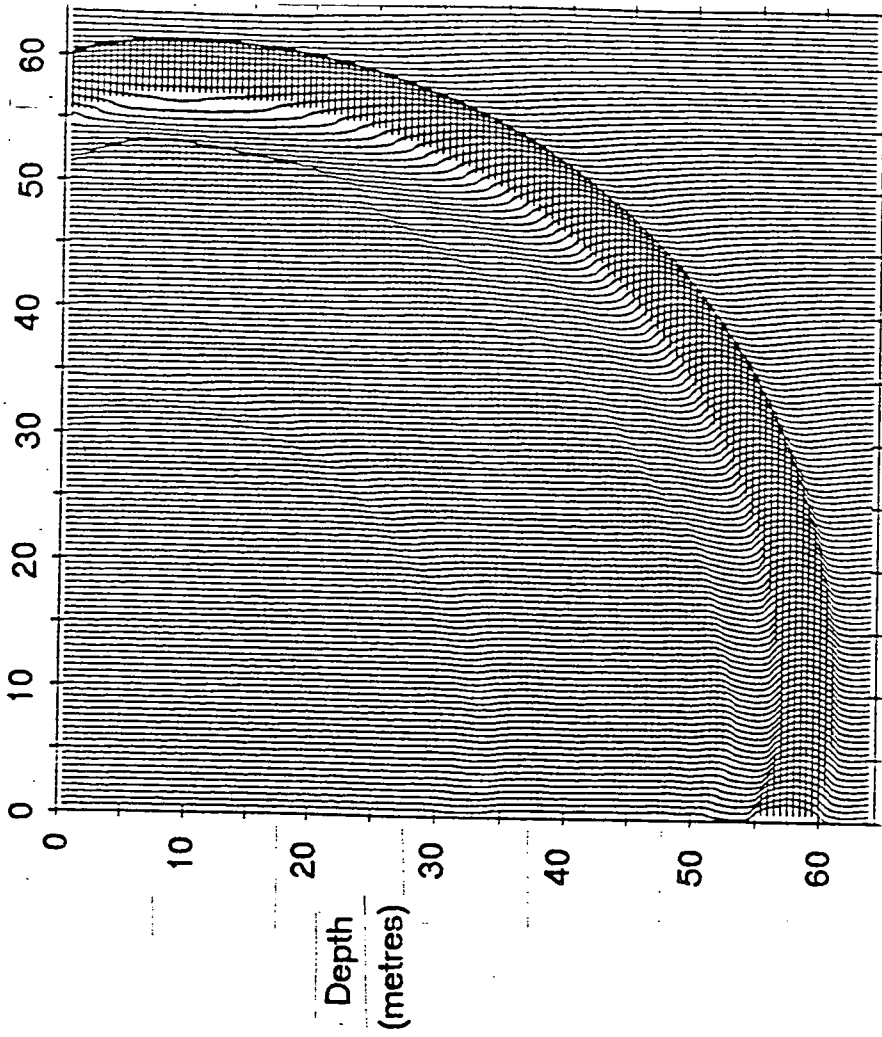
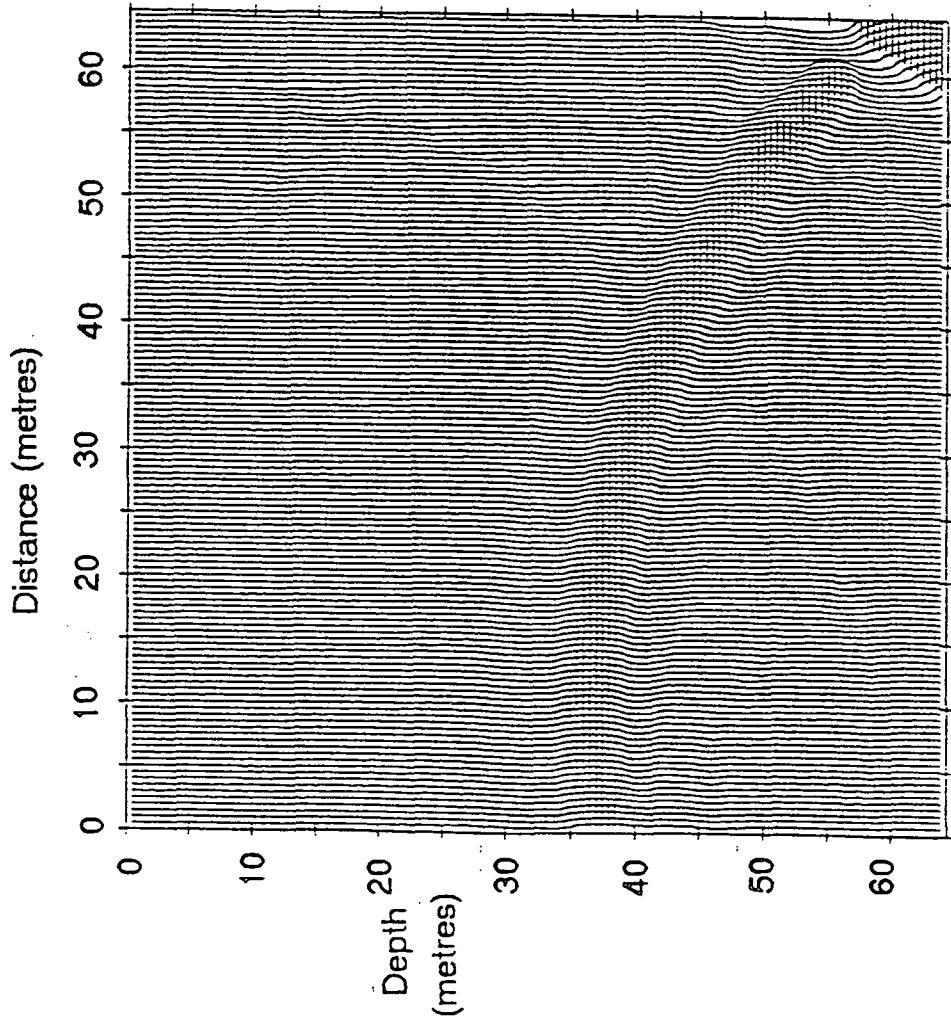
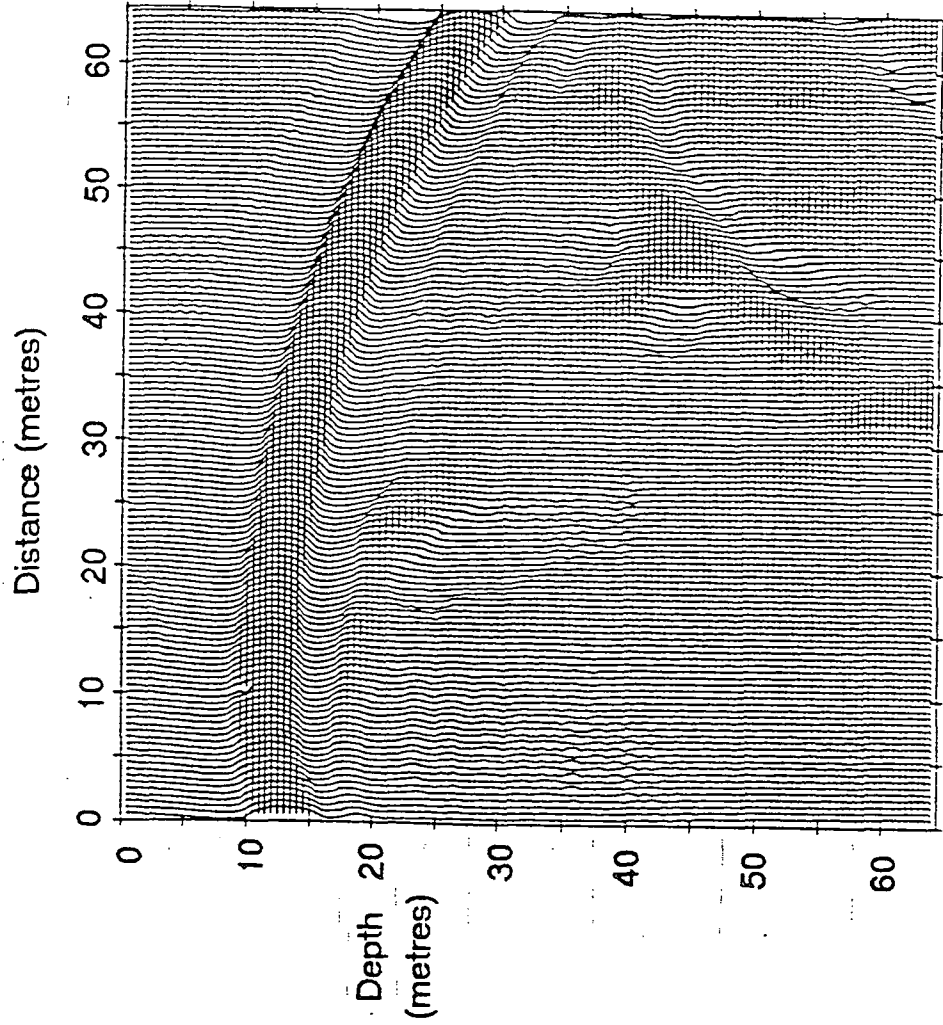


Figure 4.10 Evolution of the wavefield with time: a) 10ms, b) 20ms, c) 30ms, d) 40ms. The original source position was at (10,10).

c)



d)



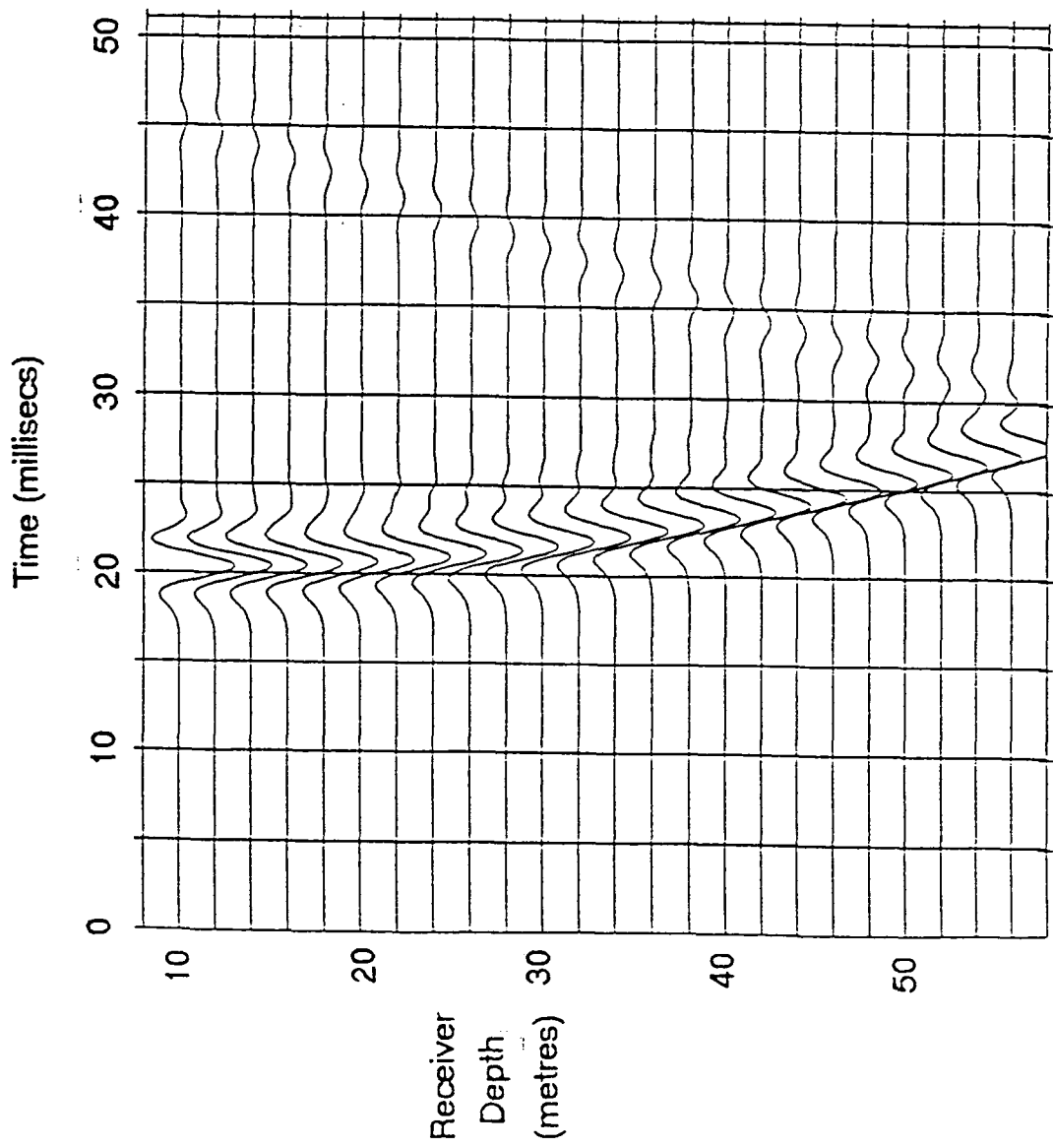


Figure 4.11 Synthetic seismogram produced for reflection event.

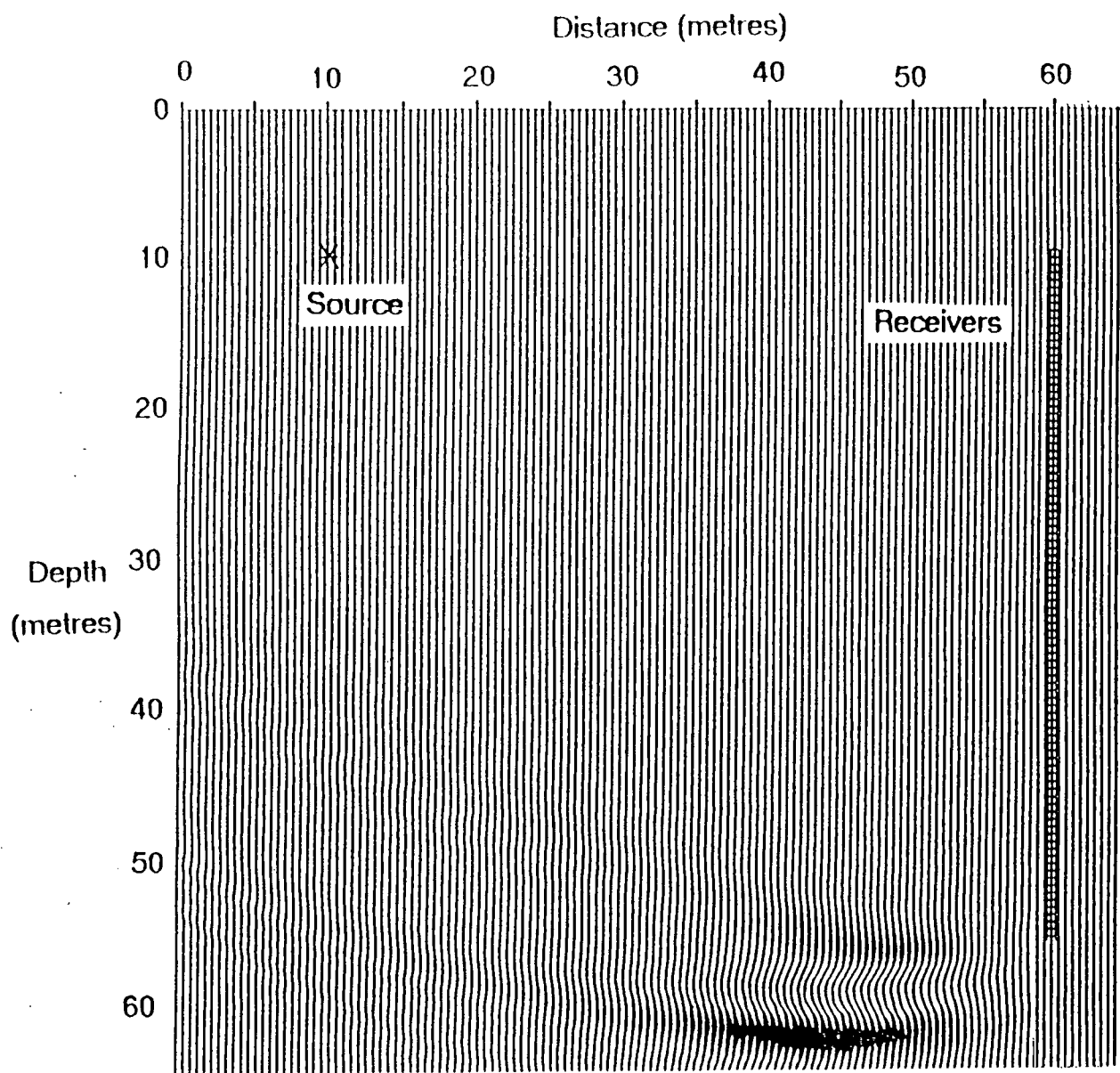


Figure 4.12 Finite-difference migrated image of the reflection event.

was used and the Newman filter was also applied. Again, smiling of the data is apparent and the image is similar to that in figure 4.7.

4.5.2 Horizontal reflecting interface model

Another synthetic dataset was modelled using the EXTRAP program. This time the reflectivity of a horizontal reflector was considered. A two-layer model was used (figure 4.9). The top layer of velocity 2400m/s was 50m thick and the bottom layer was assigned a velocity of 3000m/s. The source (again a Butterworth wavelet with a bandwidth of 150-500 Hz) was positioned at coordinates (10,20) and a vertical line of receivers were positioned at 2m intervals at depths from 10m to 56m at an offset of 50m from the source position. L^2 absorbing boundary conditions were applied on all sides of the model. The propagation of the source pulse has been plotted at times of 10, 20, 30 and 40 ms in figure 4.10, and the resultant seismogram following the application of an amplitude recovery ramp, linearly dependent on time, is shown in figure 4.11. The direct arrival and the primary reflected arrival are clearly visible.

Finite difference migration of the data following direct arrival suppression resulted in the image of figure 4.12. Substantial trailing "smiles" are apparent beyond the area of illumination of the reflecting interface. Generalised Kirchhoff migration of the same synthetic data with a full imaging aperture (i.e. allowing all possible reflector dips to be imaged) led to a very similar image (figure 4.13). Subsequently, the migration aperture was reduced to allow geological dips of less than 22.5° and the data re-migrated (figure 4.14). This image is sharper defining the zone of illumination more clearly. The Generalised Kirchhoff approach is preferable to the

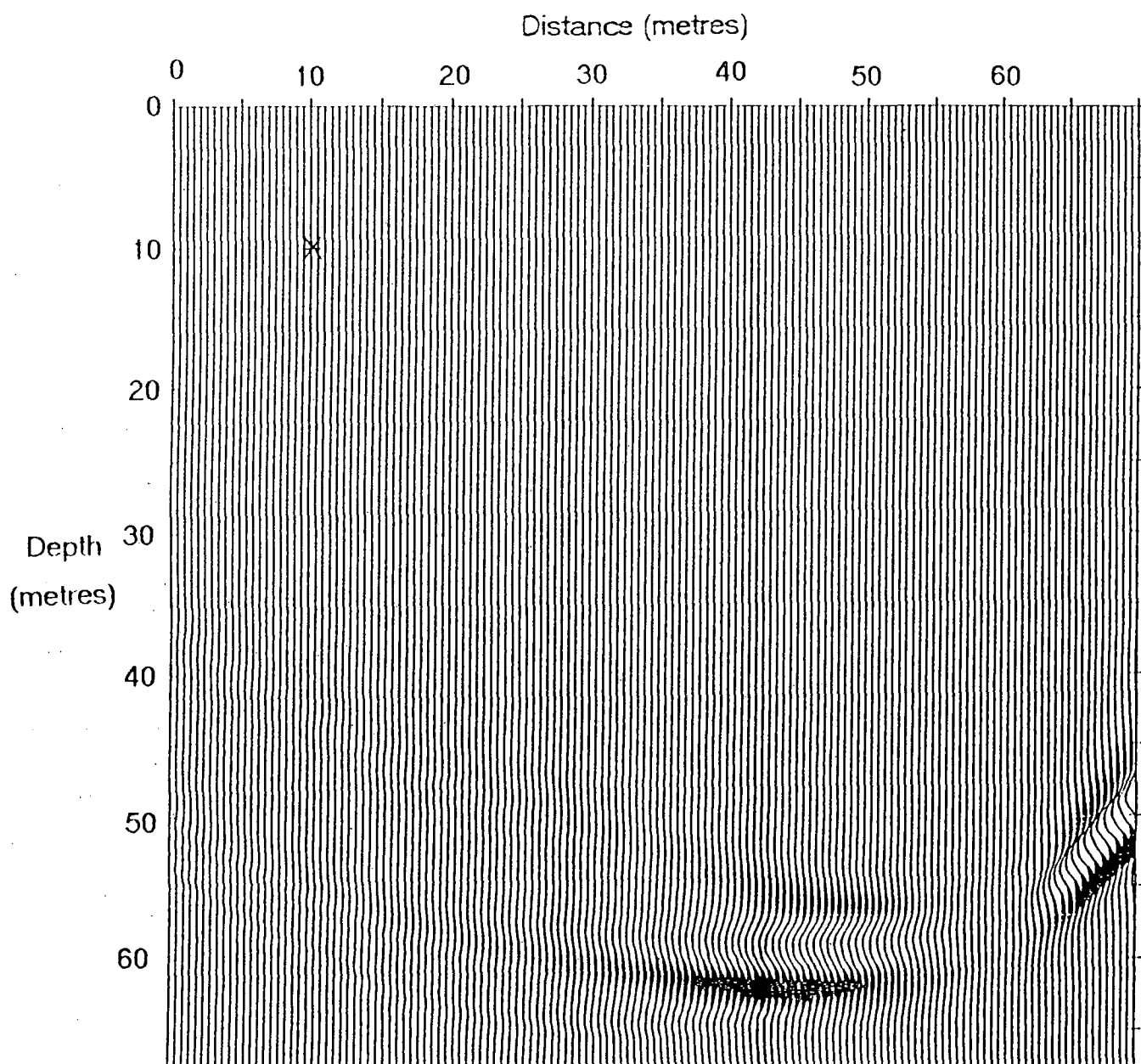


Figure 4.13 Generalised Kirchhoff (full aperture) migrated image of the reflection event.

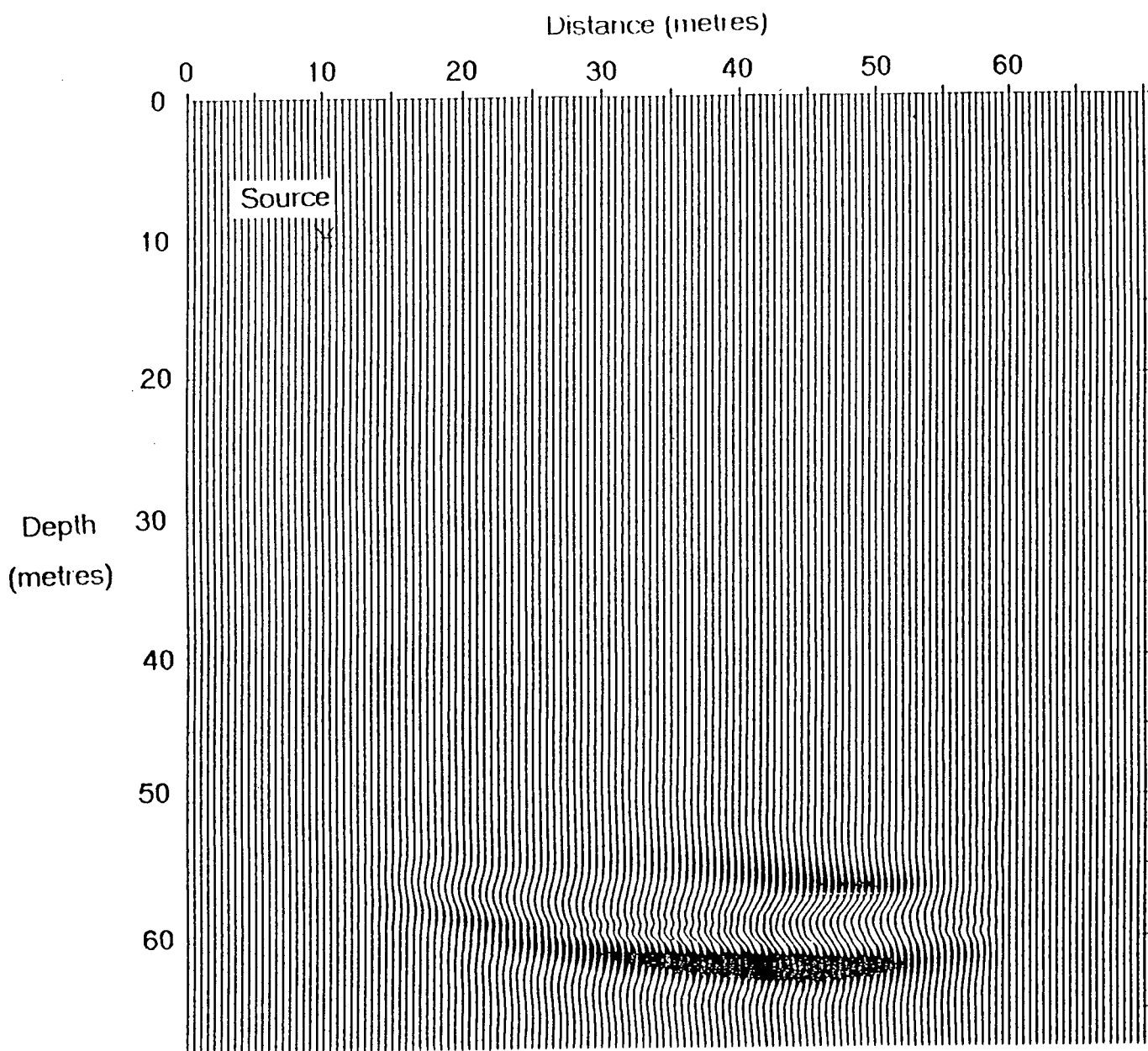


Figure 4.14 Generalised Kirchhoff (22.5° dip aperture) migrated image of the reflection event.

finite-difference approach in this case, because it is faster and also because of the extra control of image quality available during the migration process by the use of an imaging aperture.

Chapter V

Results from cross-hole surveys in Coal Measures strata

5.1 Introduction

Trial cross-hole surveys were carried out at three British Coal Opencast exploration sites in the North of England. These were Tinsley Park in South Yorkshire, Lowther South in West Yorkshire and Lostrigg in Cumbria.

5.2 Tinsley Park

Test survey A was carried out at Tinsley Park in 1988 in order to assess the feasibility of the acquisition technique. Preliminary tests had been carried out with sources and receivers positioned at depth intervals of 4m corresponding to those used in earlier cross-hole tomographic work (Findlay, 1987 and Kragh, 1990), but this resulted in spatial aliasing of the data, prohibiting the use of wavefield separation techniques. The survey presented in this section utilised 2m source and receiver intervals.

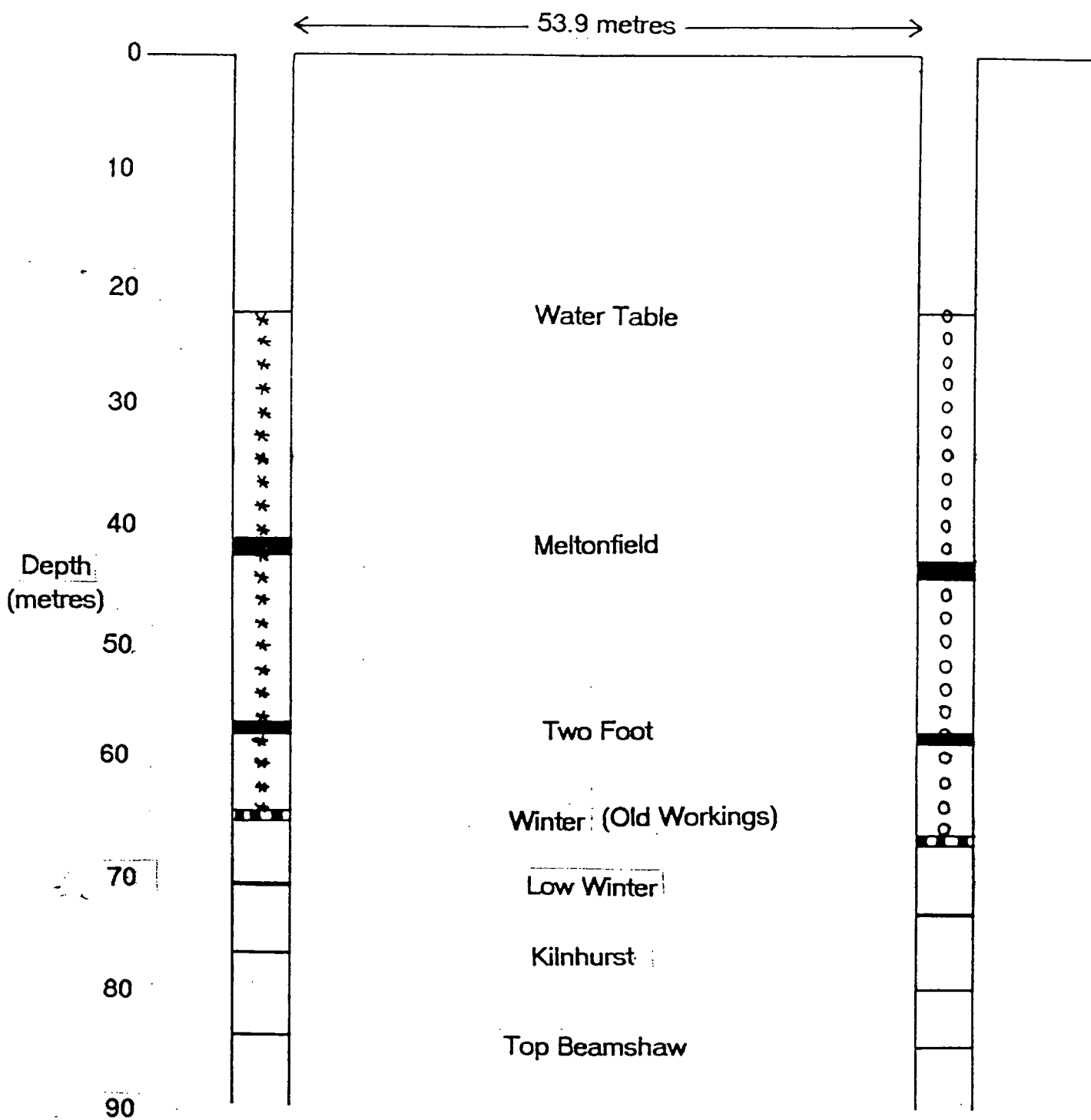


Figure 5.1 Logged coal seams and source and receiver positions for the boreholes used in survey A.

Figure 5.1 shows the interpreted stratigraphic logs and shot and receiver locations for survey A. The water table was at a depth of 22m and the boreholes were blocked at depths of 65m (source hole) and 66m (receiver hole). Repeated shots were fired at depths ranging from 22m to 64m and the hydrophone depths ranged from 22m to 66m. Seismic detonators were used as sources. The principal coal seams logged are listed in Table 5.1.

Seam	Depth to base (metres)	Thickness (metres)
Meltonfield	41.85	1.55
Two Foot	57.00	0.90
Winter	64.70	0.80
Low Winter	70.30	0.15
Kilnhurst	76.20	0.20
Top Beamshaw	83.35	0.45

**Table 5.1 Principal coal seams in stratigraphic logs from
survey A at Tinsley Park**

Of these seams, the Winter seam is known to have been worked with virtually total extraction and the remaining seams are believed to be solid near these boreholes.

The data from this survey were processed and imaged by means of the VSP-CDP transform. A simple velocity field (see table 5.2), derived from a combination of cross-hole tomography and uphole shots, was used to map the data in to a depth section.

Depth range (metres)	Interval Velocity (metres/sec)
0-21	2100
21-25	2850
25-40	2950
40-43	2300
43-55	3250
55-57	2450
57-64	2800
64-66	2400
>66	3000

Table 5.2 VSP-CDP velocity field used for survey A

In the data processing stage the zero-phase waveshaping deconvolution was omitted and so reflection depths will correspond to the first break of a wavelet rather than to a peak or a trough. The depth section for the upgoing wavefield is presented in figure 5.2. An AGC of length 25m has been applied vertically down the section to compensate for variations in signal amplitude. The horizontal trace separation is 2m and the vertical sampling interval is 0.25m. Normal SEG polarity of reflection amplitudes (i.e. white amplitudes are compressive and black amplitudes are dilatational) has been used in this display and all subsequent sections. Clear reflections are observed from the two shallowest and thickest seams (the Meltonfield

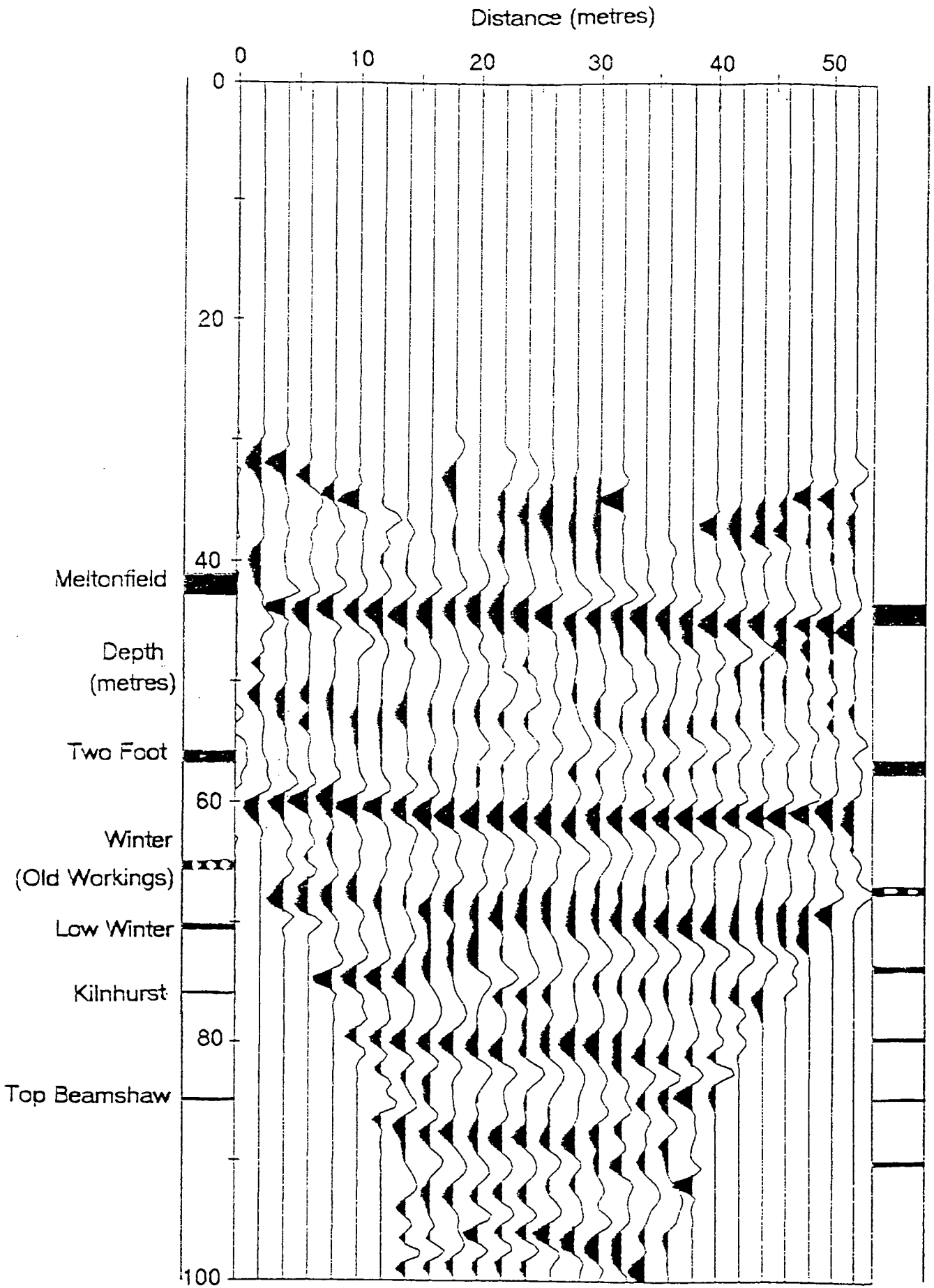


Figure 5.2 Depth section produced by VSP-CDP transform of upgoing reflections for survey A.

and the Two Foot). The reflection event seen at a depth of around 65 metres from the worked Winter seam is disrupted probably as a result of collapses in the roof of the seam and the irregular nature of the room and pillar mineworkings. Signal penetration was poor below this seam and no shots or receivers were located below the seam depth because of borehole blockages. As a result the quality of the image below the Winter seam is poor. The seams are also thinner in this area and one would expect that the reflection response from these seams would be weaker as well. The vertical resolution in the data is indicated by the wavelength of the reflections from the seams. In this case, the wavelengths are around 4m and one would therefore expect to be able to resolve reflector discontinuities caused by faulting or old mineworkings on a scale of 2m or even better. The strength of the reflections from targeted coal seams in this initial survey was very encouraging for the development of this technique.

5.3 Lowther South, West Yorkshire

Four cross-hole surveys were acquired at the Lowther South exploration site in West Yorkshire in the autumn of 1989. The first three surveys were acquired using three collinear boreholes (I, II and III) near to, but not within, a major fault zone known as the Methley-Saville fault. The line of the boreholes was selected to be perpendicular to the strike of the fault. The fourth survey was acquired between two boreholes (IV and V) located within the fault zone and lying approximately collinear

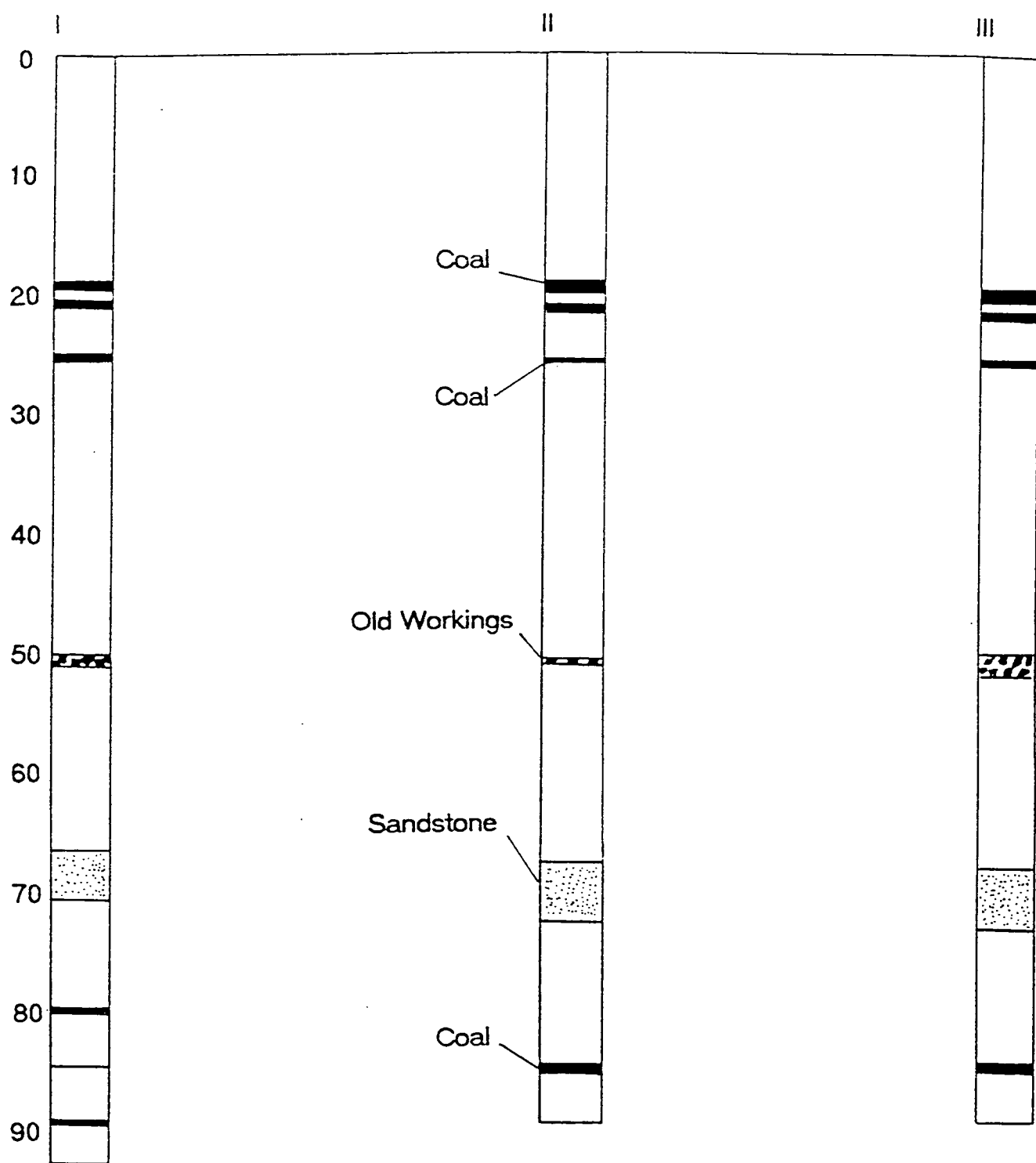


Figure 5.3 Interpreted stratigraphic log for the collinear boreholes I, II and III at Lowther South, West Yorkshire. Principal seams and their thicknesses are depicted.

to the first three boreholes. The interpreted stratigraphic logs for the three collinear boreholes are shown in figure 5.3. The principal seams are listed in Table 5.3.

Seam	Depth to base (metres)	Thickness (metres)
Kents Thick	21.4	1.3
1AY	25.5	0.2
Barnsley Top Softs	51.0	1.1
Dunsil	80.2/84.5*	0.7
*Seam is shallower in borehole I than in boreholes II and III		

Table 5.3 Principal seams in boreholes I,II and III

The logs indicate that the strata lie horizontally between these boreholes with the exception of a small fault which must cut the Dunsil seam between boreholes I and II. The seams are solid apart from the Barnsley Top Softs which is known to have been extensively worked in the area. The water table intersected these boreholes at a depth of 8m and the boreholes were blocked at depths of 62m, 57m and 52m, respectively.

5.3.1 Survey B, illustrating the combination of up- and down- going sections

Shots were fired from 27 depth levels at 2m intervals ranging from 10m to 62m in borehole I and the data were recorded at 23 hydrophone depth levels at 2m intervals ranging from 16m to 60m in borehole II (see figure 5.4). The borehole

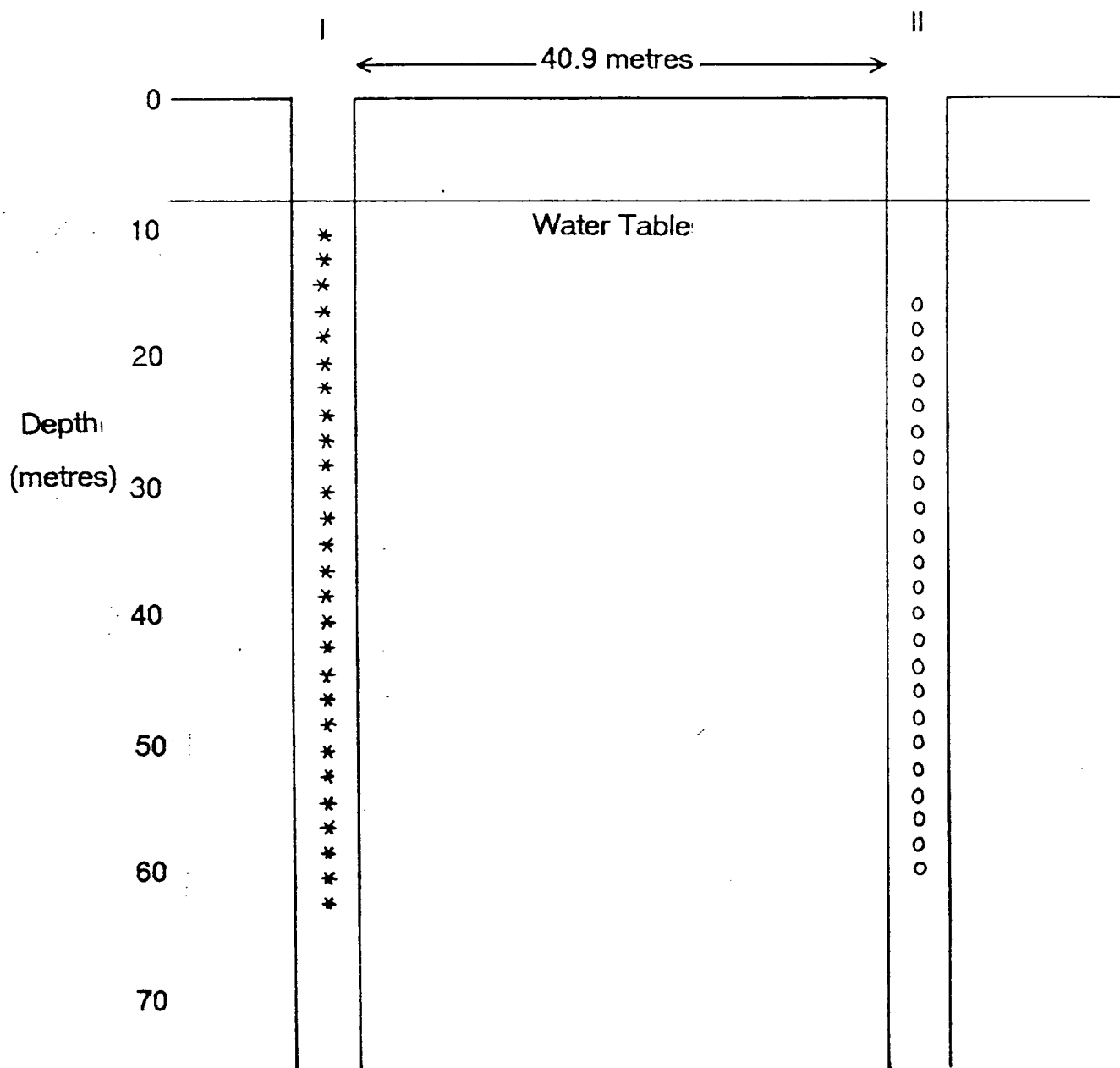


Figure 5.4 Shot and receiver positions for survey B.

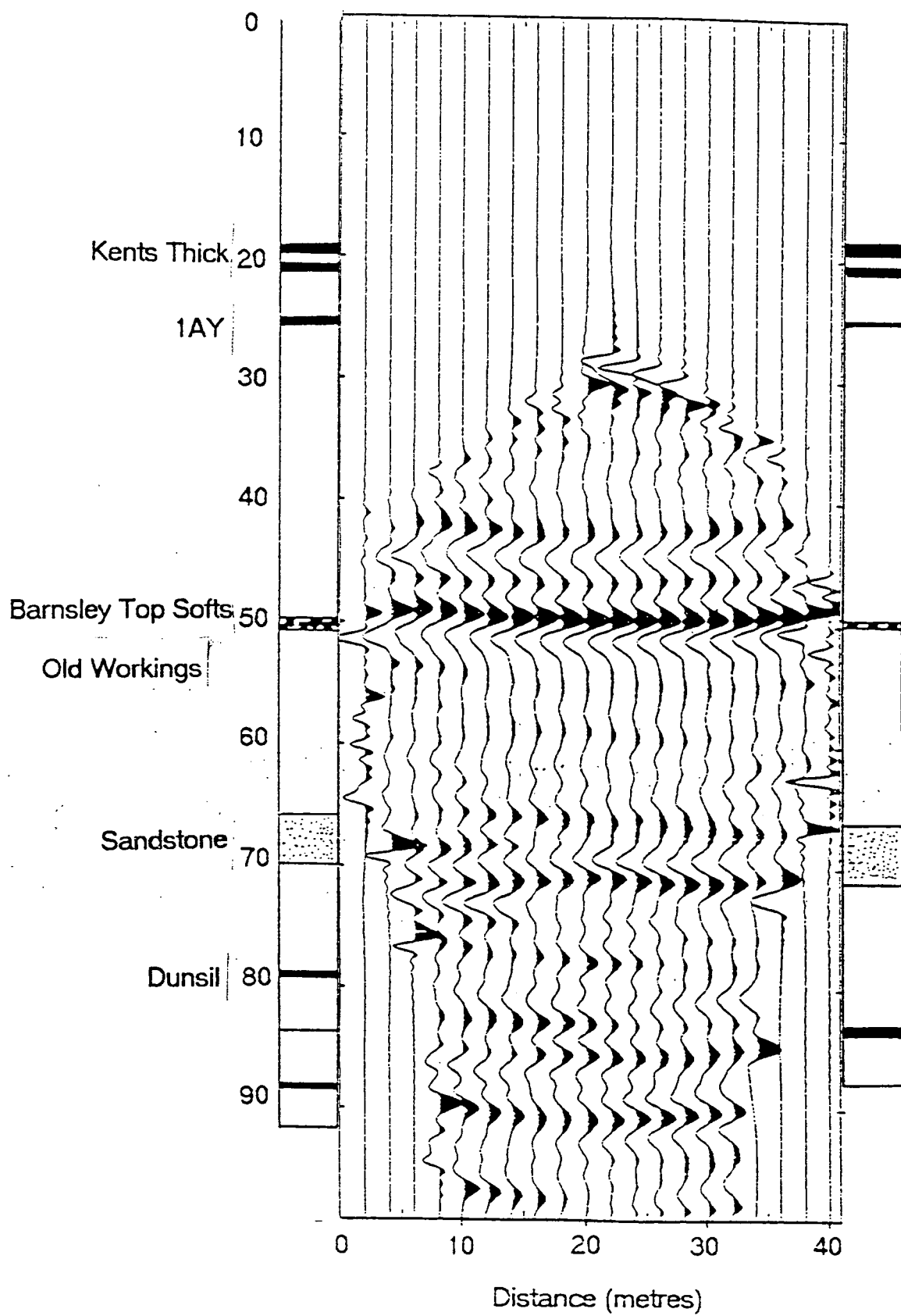


Figure 5.5a Upgoing Generalised Kirchhoff migrated section (22.5° dip aperture) for survey B.

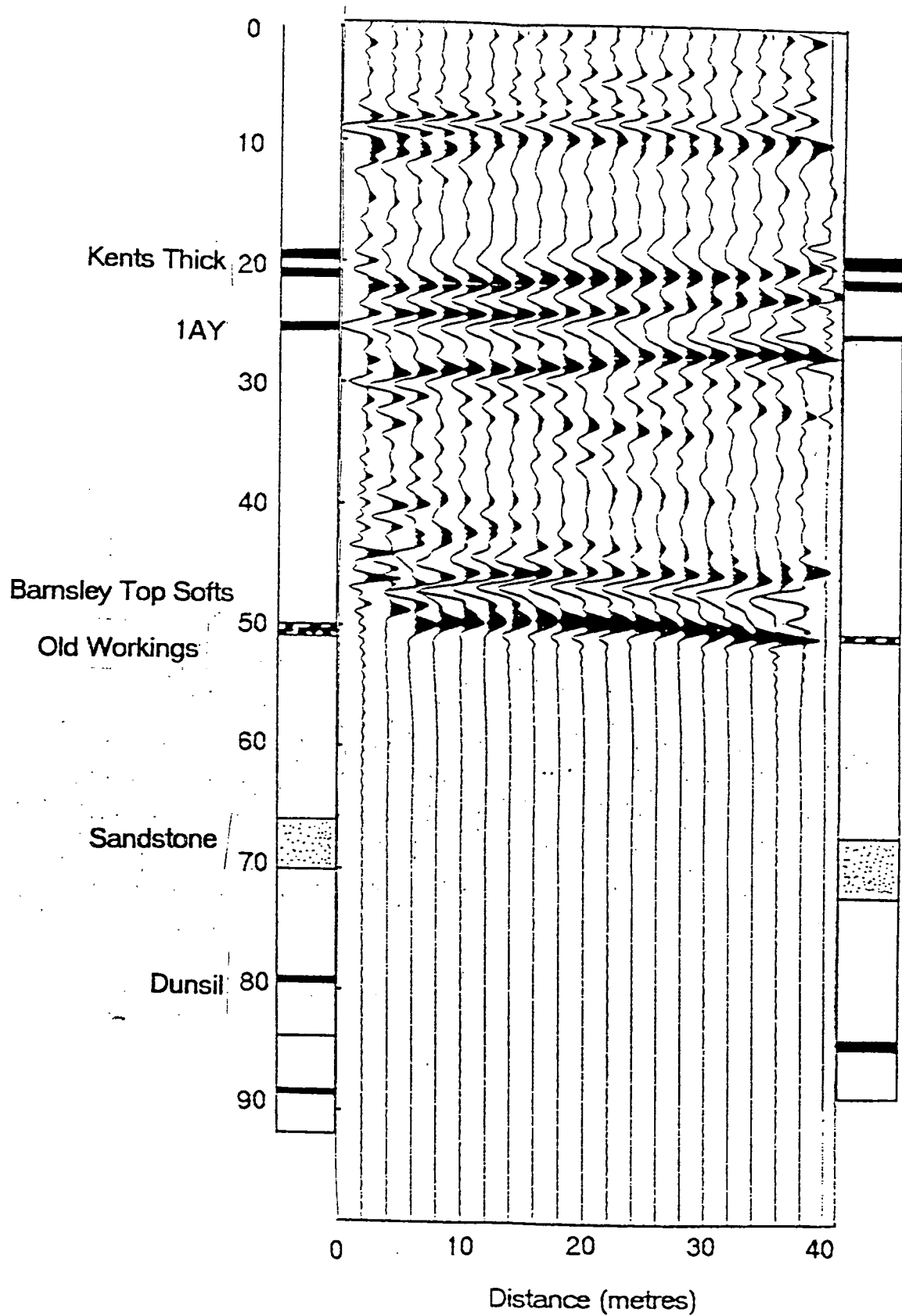


Figure 5.5b Downgoing Generalised Kirchhoff migrated section (22.5° dip aperture) for survey B.

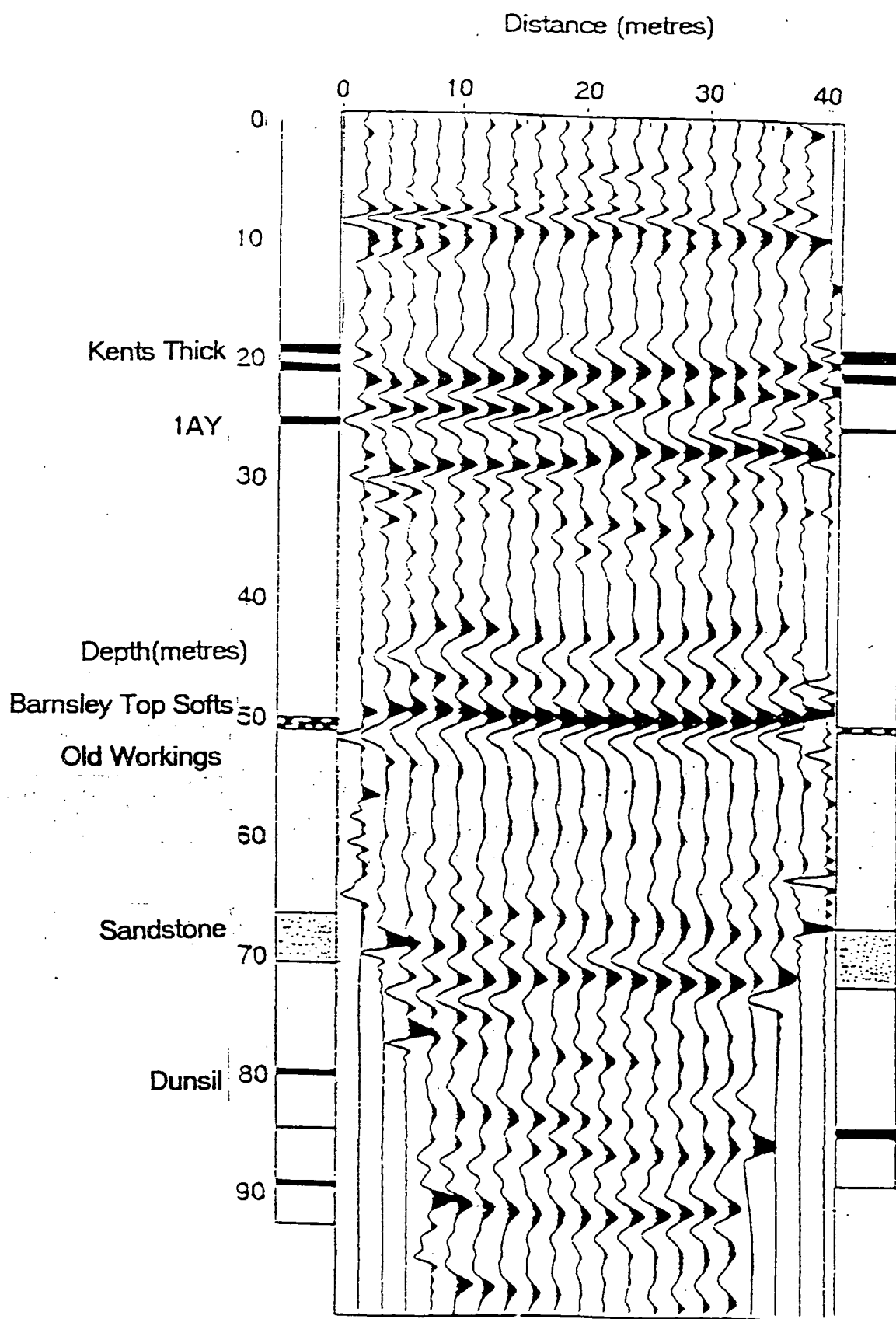


Figure 5.5c Combined upgoing and downgoing Generalised Kirchhoff migrated section for survey B. Sections are merged between 40m and 50m depth using cosine tapers.

separation was 40.9m at the surface. Both boreholes were blocked at a depth of around 62m. The data were processed as described in chapter III and imaged by means of the G-K migration algorithm. A migration dip aperture of +/- 22.5° was used. The migration velocity field is described in table 5.4.

Depth range (metres)	Migration interval velocity (metres/sec)
0-10	1200
10-22	2250
22-54	2600
54-100	2700

Table 5.4 Migration velocity field used for survey B

The depth sections are presented in figure 5.5. Figure 5.5a is the depth section imaged from the upgoing reflections and figure 5.5b is the depth section imaged from the downgoing reflections. Figure 5.5c is the depth section of the combined up- and down- going sections produced by melding the two sections together over a depth range of 40-50m. An AGC of length 25 metres has been applied vertically to balance up the amplitudes at all depths. Several clear reflectors are apparent corresponding to the coal seams at depths of 8m, 20m, 25m and 50m. The seams at 20m and 25m have a flat, continuous character as might be inferred from the stratigraphic logs. The Barnsley Top Softs seam at 50m depth has been worked right across the section, but still shows a strong, coherent reflection. The presence of a fault intersecting borehole

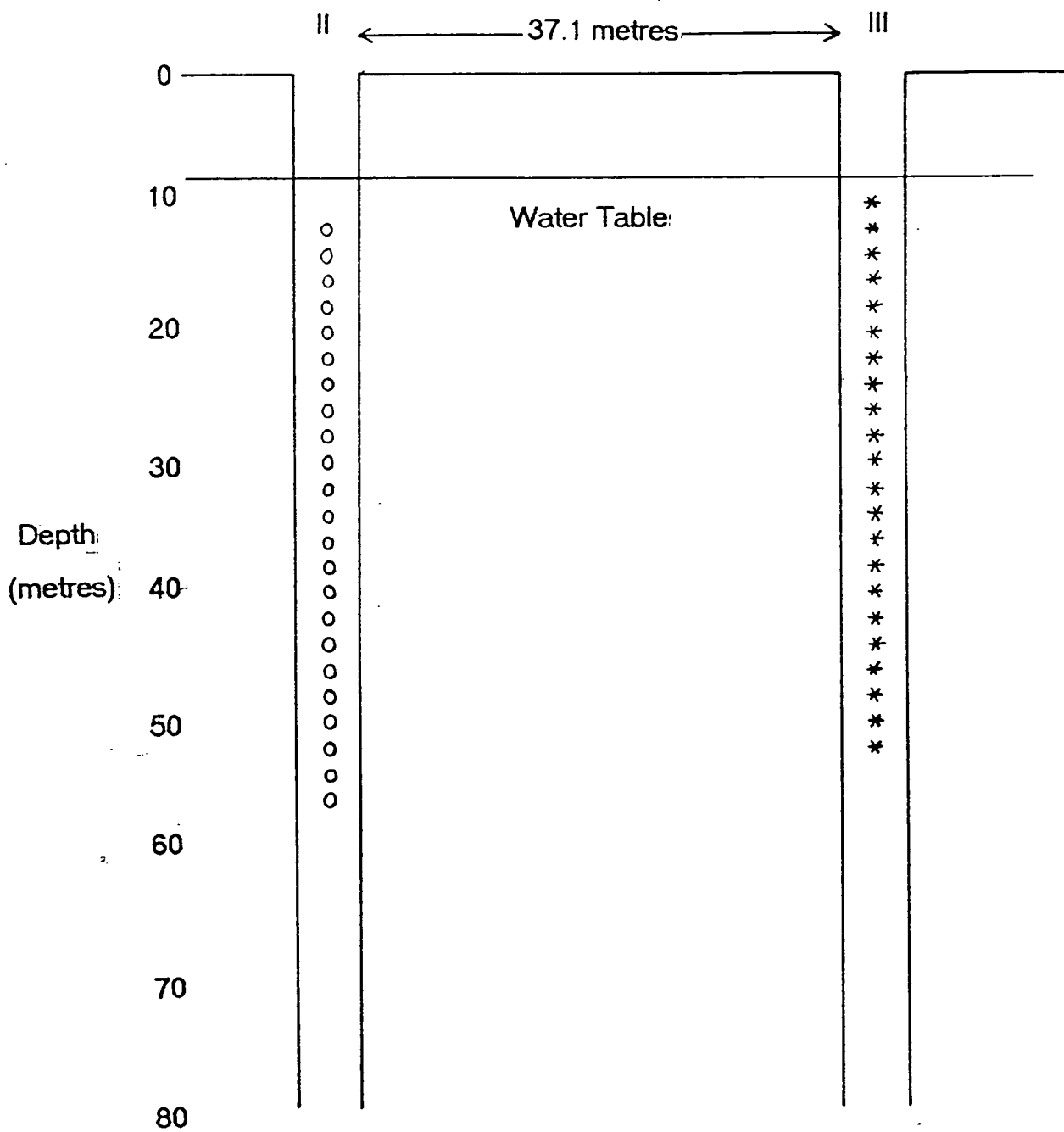


Figure 5.6 Shot and receiver locations for survey C.

B at a depth of between 50m and 80m might be expected from interpretation of the borehole logs. Unfortunately, the fault is not apparent on the cross-hole survey either because it is located too close to the borehole where reflector coverage is very poor, or else because of the lack of penetration of the signal through the old mineworkings at 50m depth. The Dunsil seam (log depth 85m) may have been imaged closer to 90m due to errors in the velocity field below the bottom source and receiver depths. Image quality is poor in those regions of the image where specular reflection raypath coverage is of a low density. These regions are in the vicinity of the boreholes and in the areas near the boreholes above the top source and receiver positions and below the bottom source and receiver positions.

5.3.2 Survey C; between boreholes II and III

In this survey a total of 22 shot positions were used in borehole III at 2m depth intervals ranging from a depth of 10m to a depth of 51.9m (see figure 5.6). Electrical detonators were used as an explosive source. Hydrophone locations were at 2m depth intervals ranging from a depth of 12m to a depth of 56m in borehole II. The borehole separation was 37.1m at the surface. The data were processed as described in chapter III and imaged by means of the G-K migration algorithm with a migration dip aperture of $\pm 22.5^\circ$. The isotropic velocity model used to migrate the data is described in table 5.5.

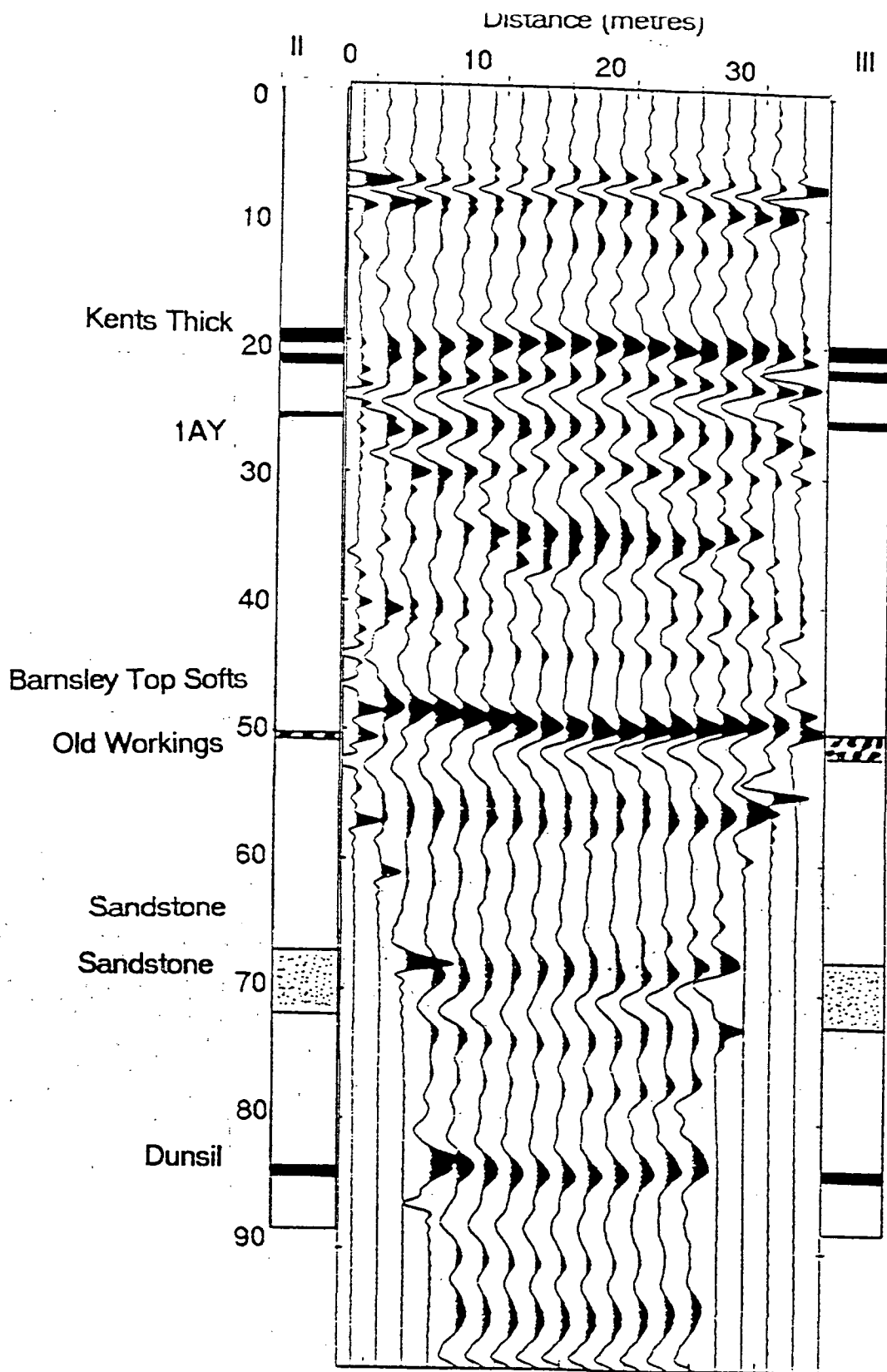


Figure 5.7 Depth section produced by combining Generalised Kirchhoff migrated sections of the upgoing and downgoing reflections for survey C.

Depth range (metres)	Migration interval velocity (metres/sec)
0-10	1200
10-22	2400
22-40	2600
40-100	2700

Table 5.5 Migration velocity field used for survey C

The depth migrated section produced by combining the sections produced from the upgoing and downgoing wavefields from this survey are shown in figure 5.7. Again an AGC of length 25 metres has been applied in the vertical direction. The section is very similar to that produced for survey B (figure 5.5c) with clear reflections from the principal coal seams. The seams appear to be of a flat and continuous nature.

5.3.3 Survey D; wider-spaced cross-hole reflection survey

A further survey was acquired between boreholes I and III (at a separation of 78.0m) for comparison purposes with surveys B and C and also in the expectation of imaging the strata near borehole II, where the image quality is very poor for surveys B and C. When the fieldwork was carried out it was found that borehole I had collapsed still further since survey B and was blocked at a depth of 52m. The borehole collapse was probably aggravated by the detonation of the source charges for survey B in this borehole. A larger charge size (25g of dynamite) was used to improve the signal-to-noise ratio because of the greater borehole spacing in this

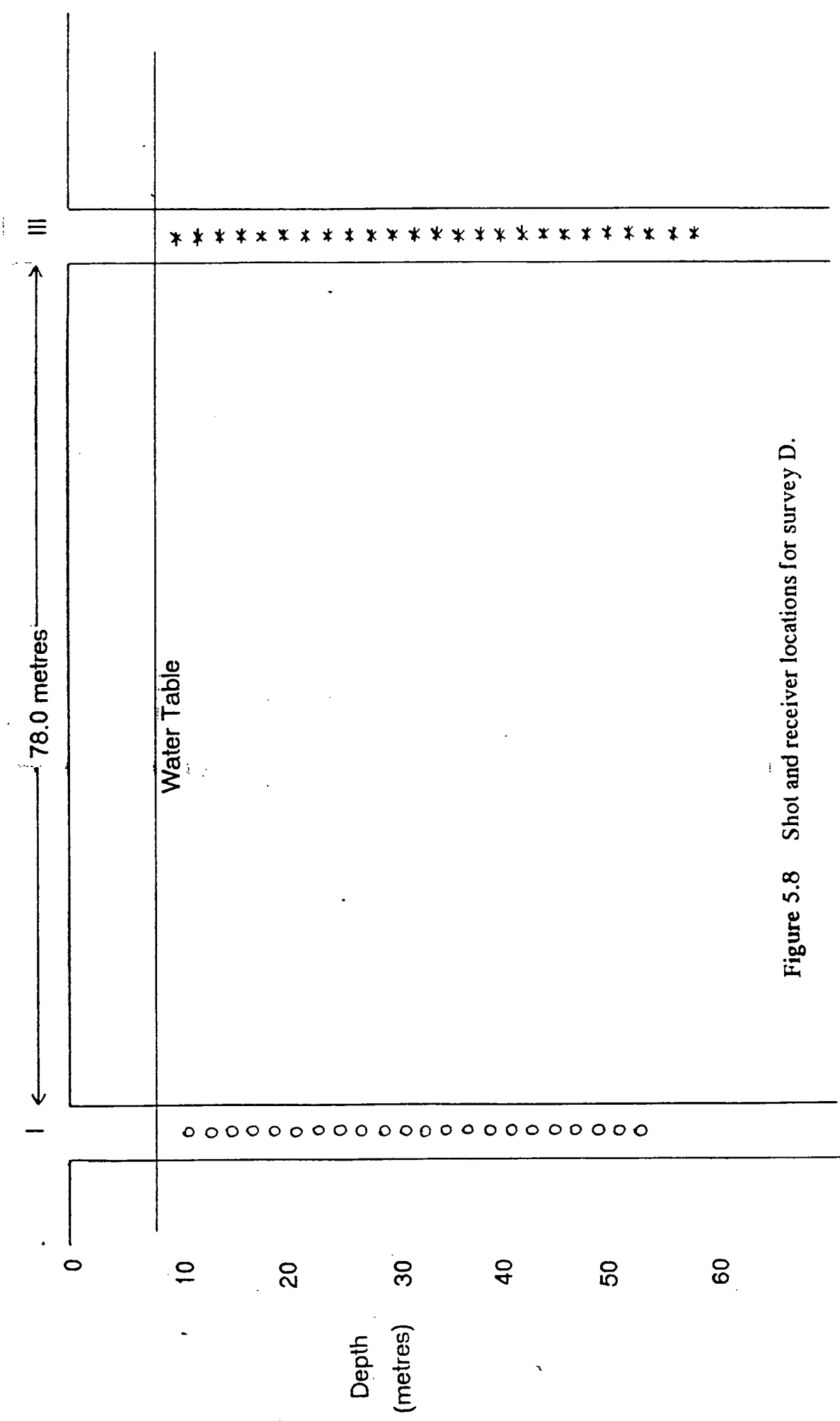


Figure 5.8 Shot and receiver locations for survey D.

survey. The data were processed as discussed in chapter III and the imaging was once more carried out by means of the Generalised Kirchhoff method with a dip aperture of $\pm 22.5^\circ$. The interval velocities used in the migration are listed in table 5.6.

Depth range (metres)	Migration interval velocity (metres/sec)
0-10	1200
10-22	2400
22-50	2700
50-54	2500
54-100	2800

Table 5.6 Migration velocity field used for survey D

The combined upgoing and downgoing reflection sections are presented in figure 5.9. An AGC of length 25 metres has been applied to balance out the variations in signal amplitude down the section. The section possesses some similarities to the previous two surveys. In particular, the reflection from the old mineworkings at around 50m depth has a comparable character. The reflections from the shallower seams whose true depth is around 20m have been imaged shallower, presumably because the velocity field was too large for the migration of the downgoing wavefield. The quality of this image is generally much poorer than that obtained in surveys B and C. This illustrates that there is a maximum limit to the borehole spacing which can be used successfully for a given range of depths of the source and receiver positions.

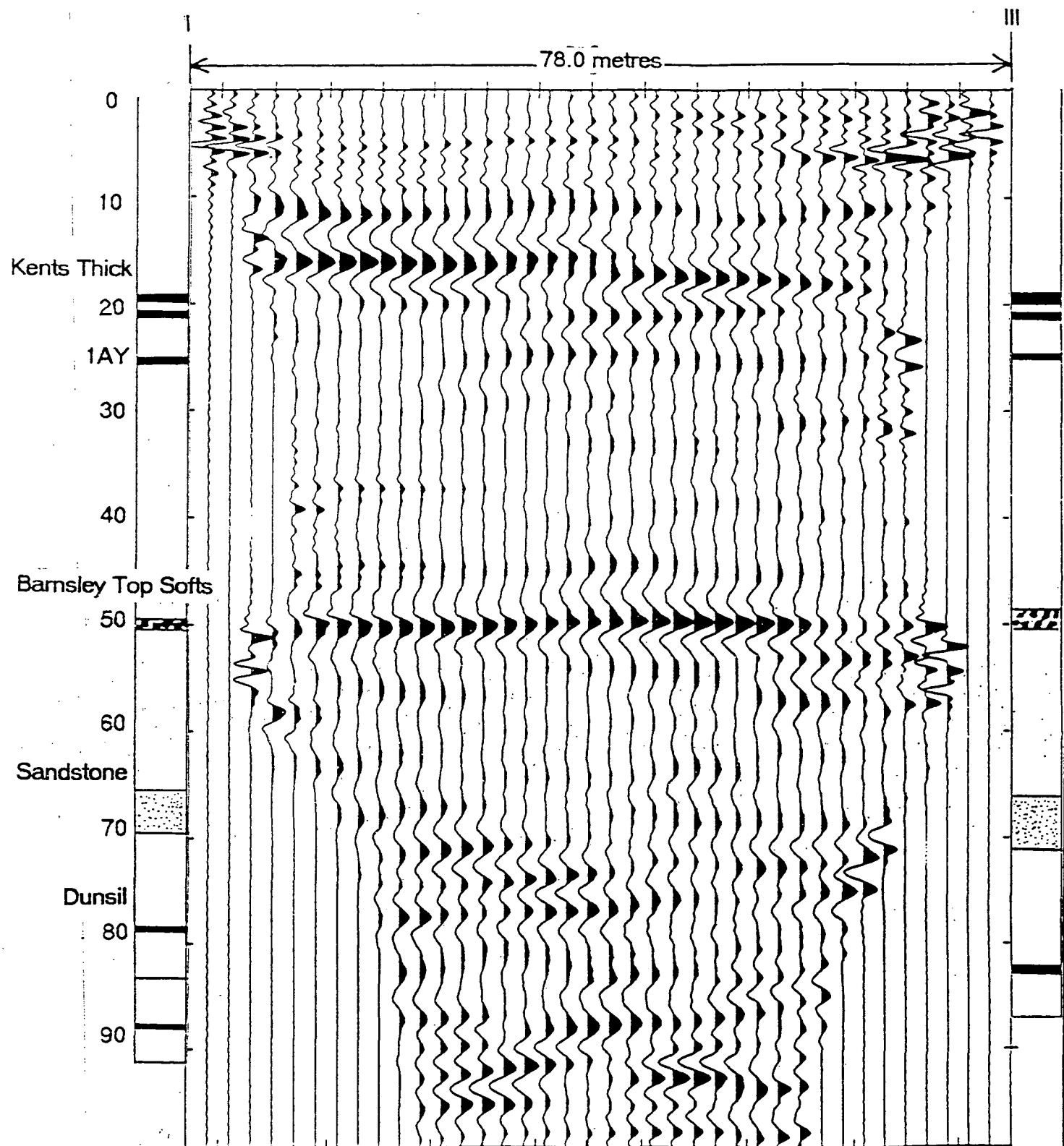


Figure 5.9 Depth section produced by combining Generalised Kirchhoff migrated sections of the upgoing and downgoing reflections for survey D.

The deterioration in the image quality is a consequence of the reflection raypaths becoming more horizontal as the borehole separation increases. This results in a problem in that a very accurate velocity model specification is needed to reduce positioning (and hence stack contribution) errors to an acceptable level (see section 3.7). The processing of shallow dipping reflections is also made more difficult because of the problems in separating the reflected energy from the direct arrivals when the travel times and moveouts are very similar.

5.3.4 Survey E; cross-hole survey through major fault zone

Another survey was carried out at the Lowther South exploration site between boreholes IV and V in a field adjacent to boreholes I, II and III. The boreholes were 32.0m apart, located in the zone of the Methley-Saville fault and the line of the boreholes was approximately perpendicular to the strike of the fault. The principal seam depths taken from the stratigraphic logs are indicated in table 5.7.

Seam	Hole IV Depth to base (metres)	Hole V Depth to base (metres)	Thickness (metres)IV/V
Kents Thick	13.4	22.36	1.4/2.4
1AY	ABSENT	27.84	- /0.2
Barnsley Top Softs	28.46	53.32	0.86/1.2
Dunsil	59.26	84.11	0.7/0.5

Table 5.7 Principal seams in boreholes IV and V

Shots were fired in borehole IV at 21 locations at 2m intervals in the depth range 10-50m. Electrical detonators were used as a source. 23 hydrophone positions were used in borehole V at depths between 9.79m and 53.79m. The shot and receiver locations and summary stratigraphic logs are depicted in figure 5.10.

The data were processed as described in chapter III and imaged by means of the G-K migration method with a migration dip aperture of +/-22.5°. The migration interval velocities are described in table 5.8.

Depth range (metres)	Migration interval velocity (metres/sec)
0-10	1200
10-20	2200
20-24	2250
24-40	2330
40-46	2210
46-90	2400

Table 5.8 Migration velocity field used for survey E

The combination of the migrations of the upgoing and downgoing wavefields is presented in figure 5.11. The quality of the section is good with clear reflections from the coal seams on the downthrown (right) side of the fault and from the Dunsil seam (around 60m) on the upthrown side. Errors in seam depths are a result of inaccuracies in the migration velocity model.

Interpretation of the borehole logs suggested that the fault intersects borehole IV between the Kents Thick and Barnsley Top Softs seams. It was not known whether the difference in levels of the Kents Thick seam (~7 metres) between the two boreholes was due to dipping strata or a small fault. The cross-hole seismic survey removes this ambiguity since the reflection from the Kents Thick seam is near-horizontal across the section. Therefore a small fault with a throw of 7 metres must intersect the Kents thick seam close to borehole IV where the image quality of the seismic section is poor. The location of the larger fault is clearly defined by the

IV

V

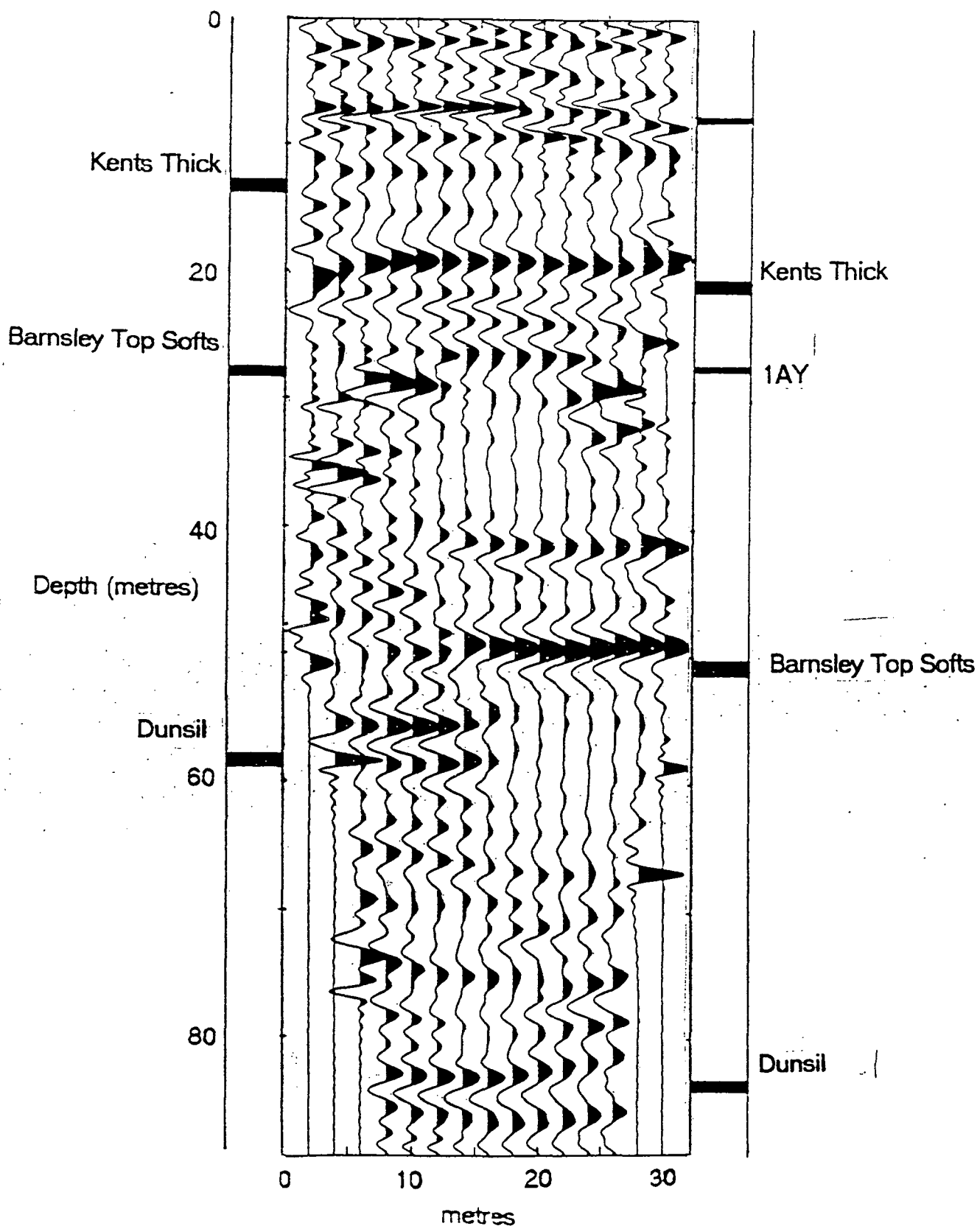


Figure 5.11 Migrated depth section with combined upgoing and downgoing reflections for survey E acquired at Lowther South.

truncations of reflections from the Barnsley Top Softs seam to the right and from the Dunsil seam to the left at the fault zone.

5.4 Lostrigg, Cumbria

The results of a further survey acquired at a British Coal opencast exploration site at Lostrigg in Cumbria are presented.

5.4.1 Survey F; old mineworkings

The target of this survey was the edge of a longwall panel in the Sixquarters seam. The boreholes (A and B) used were separated by 48.4m at the surface. The principal coal seams logged for the two boreholes are listed in table 5.9. The Sixquarters seam was worked in borehole A, but solid in borehole B. It was thought that the extent of the old mineworkings might be delimited by a small fault.

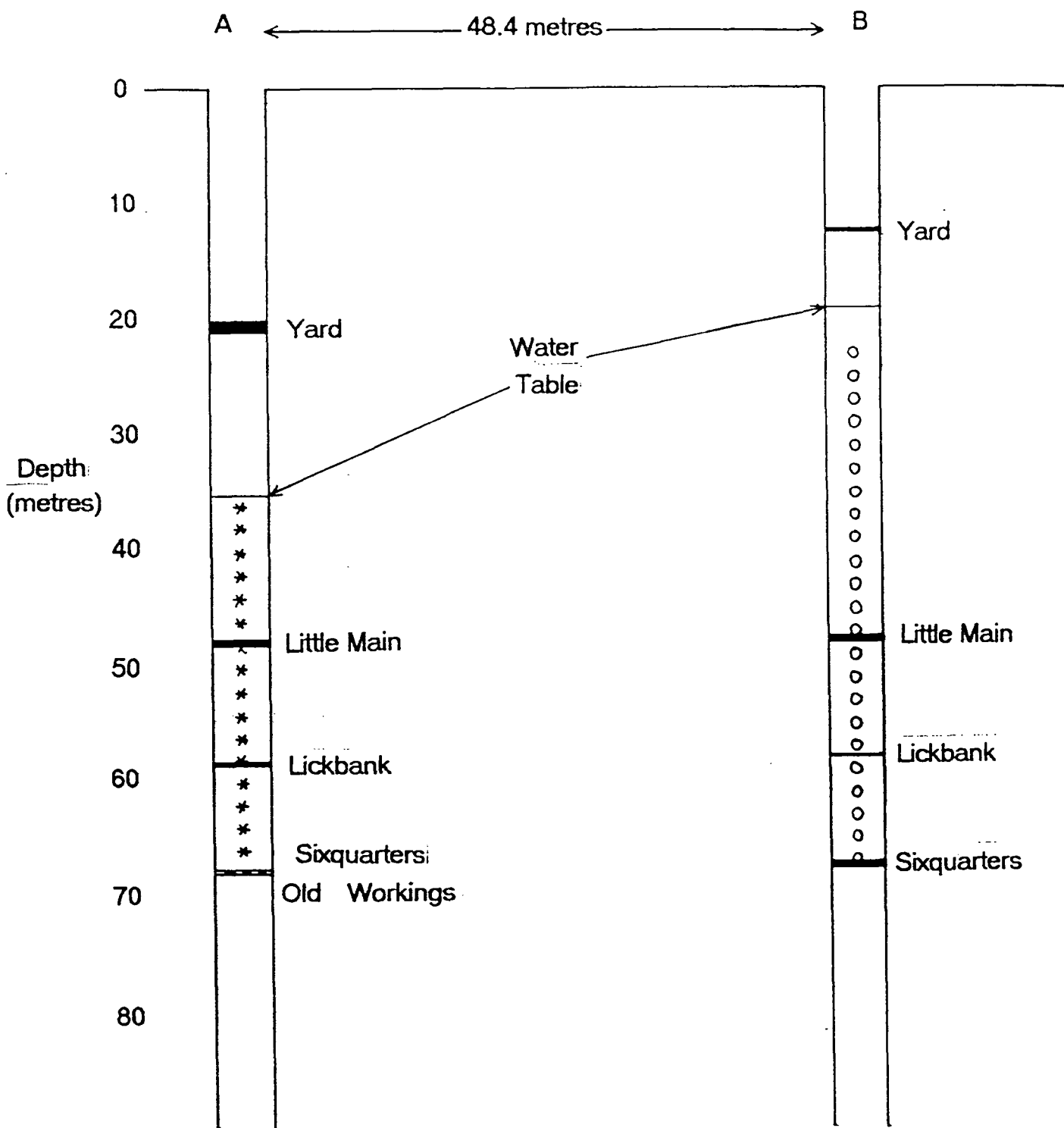


Figure 5.12 Shot and receiver locations and principal coal seams for survey F.

Seam	Hole A	Hole B	Thickness
	Depth to base	Depth to base	(metres)A/B
	(metres)	(metres)	
Yard	21.20	12.68	1.0/0.3
Lower Yard	21.42	20.87	0.1/0.2
Top Half Yard	34.41	33.40	0.2/0.2
Btm Half Yard	36.26	absent	0.2/ --
Little Main	48.21	48.06	0.6/0.6
Eighteen Inch	52.99	51.33	0.2/0.2
Lickbank	58.58	57.99	0.4/0.4
Sixquarters	68.12	67.75	0.3*/0.5
* Seam worked with some coal present			

Table 5.9 Principal seams in boreholes A and B

There were several acquisition problems with this survey. Both boreholes were blocked at the level of the Sixquarters seam and the water tables were at quite different levels in the two boreholes (35m in borehole A and 19m in borehole B). There were also large amounts of tube wave noise generated on most of the shot records and this was difficult to remove successfully.

Altogether 16 shot positions were used in the survey at 2m depth intervals in borehole A ranging from 36m-66m. The hydrophones were positioned at 23 positions at 2m intervals in borehole B ranging from 22.7m-66.7m. The shot positions and principal coal seams are shown in figure 5.12. The data were imaged using the G-K

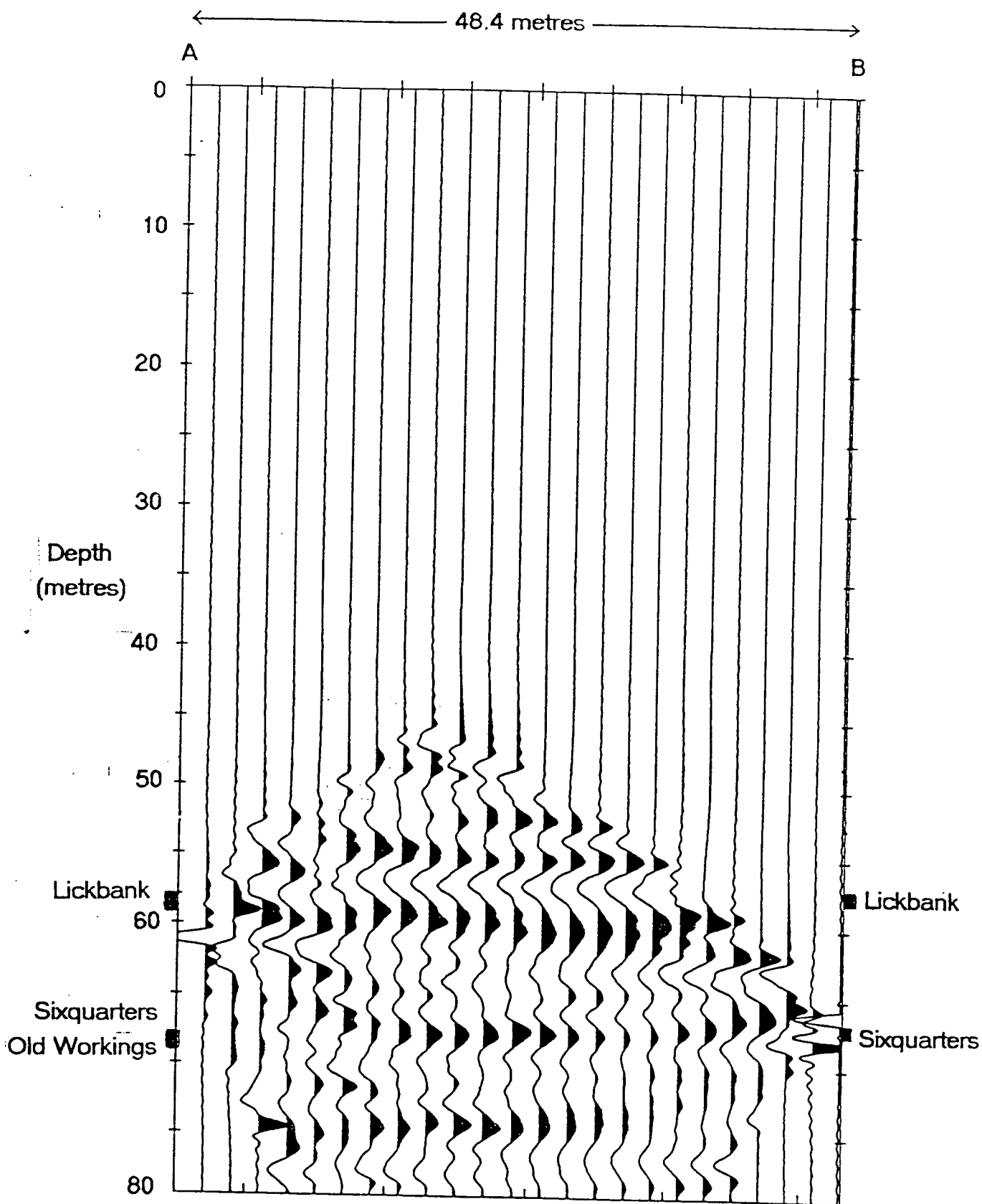


Figure 5.13 Depth section of survey F produced by Generalised Kirchhoff migration of the upgoing reflections.

migration method with a dip aperture of $\pm 25^\circ$. The migration velocity field is shown in table 5.10.

Depth range (metres)	Migration interval velocity (metres/sec)
0-20	1580
20-36	2470
36-80	3000

Table 5.10 Migration velocity field used for survey F

The migrated upgoing wavefield is shown in figure 5.13. The results are disappointing, but not surprising given the problems of reduced shot position coverage, probable significant lateral velocity variations produced by the variation in the water table, and the tube wave noise. A strong reflection is visible from the Lickbank seam, but the quality of the section is too poor to determine whether the old mineworkings were indeed terminated by a small fault.

Chapter VI

A comparison of different migration operators for cross-hole seismic data

6.1 Introduction

A comparison of the effectiveness of different migration techniques for cross-hole data is discussed in this chapter. The migration operators of a variety of techniques were considered: Finite difference migration in the (x,z,t) domain, diffraction stack summation, Kirchhoff migration, Generalised Kirchhoff (GK) Migration and Generalised Radon Transform (GRT) migration.

6.2 Migration operators

The migration operators for the various techniques are presented in figures 6.1-6.5. The operators were calculated assuming a single source-receiver pairing in a homogeneous, isotropic medium. The source and receiver were both located at a depth of 20 metres with an offset of 40 metres. The velocity field was assumed to be 3000m/s and a band-limited spike with a travel time of 20 ms was used as input to a single trace migration operation. In all cases a regular grid with sampling at 0.5m intervals in the x and z directions was employed and the source and receiver were assumed to be elements in vertical source and receiver arrays.

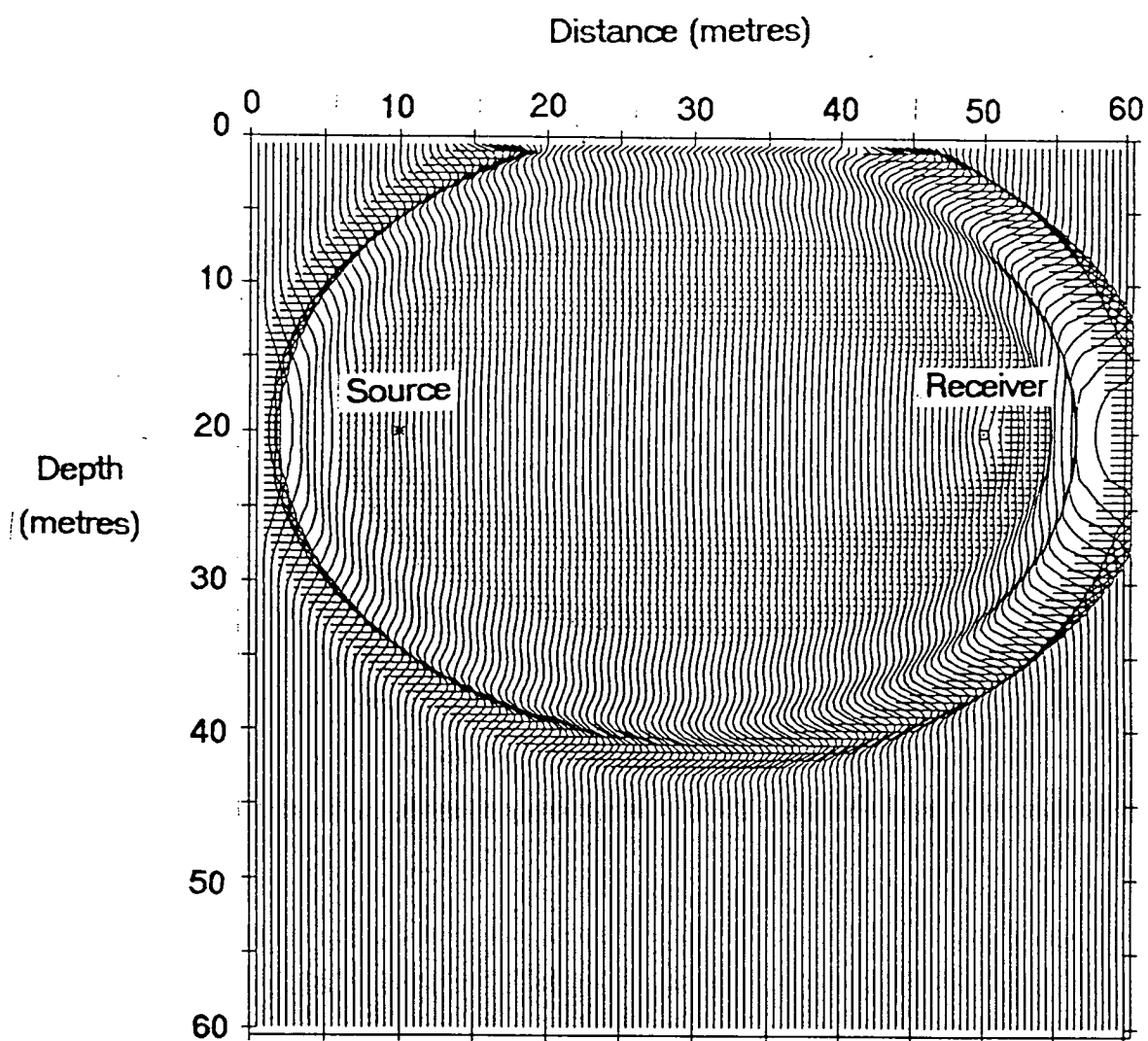


Figure 6.1 Finite-difference migration operator in a homogeneous, isotropic velocity field.

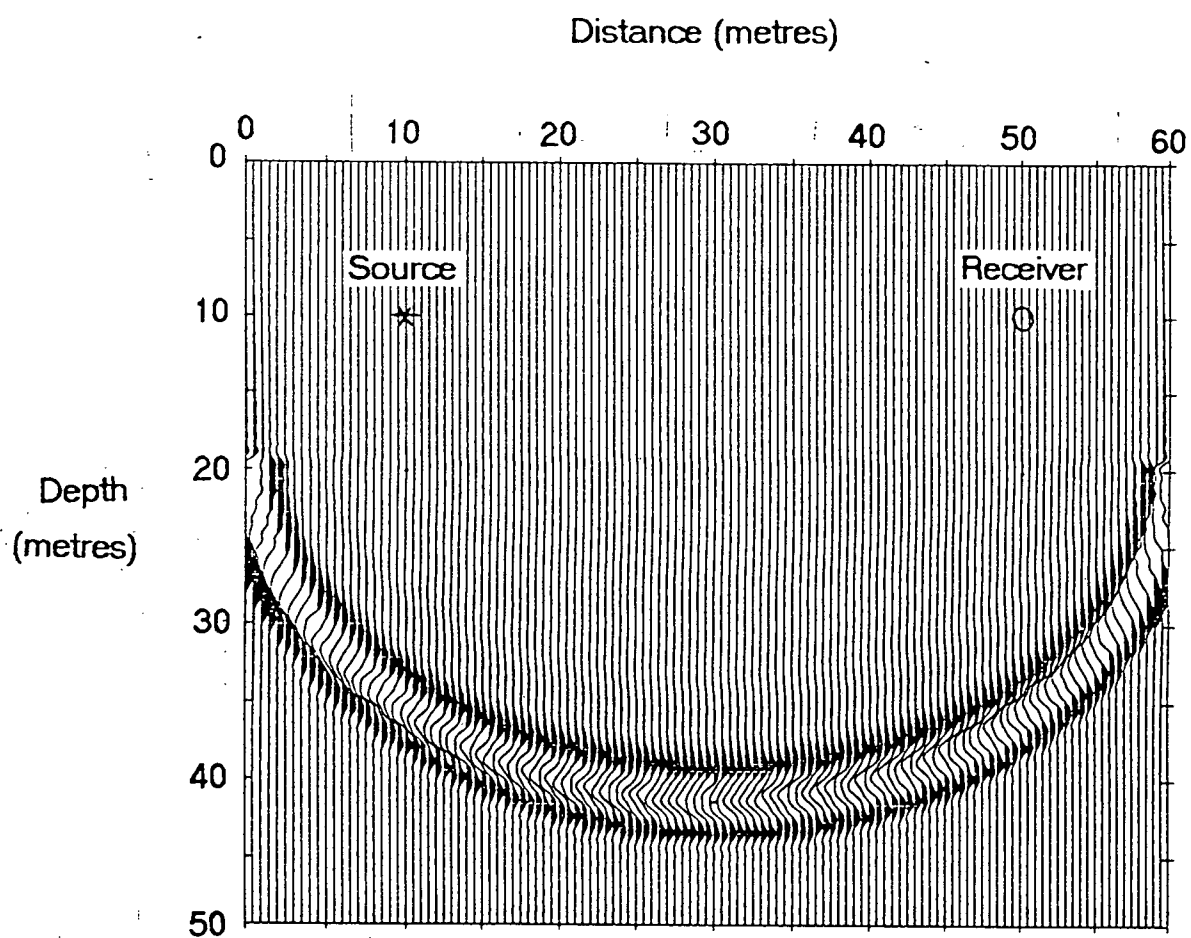


Figure 6.2 Diffraction stack migration operator in a homogeneous, isotropic velocity field.

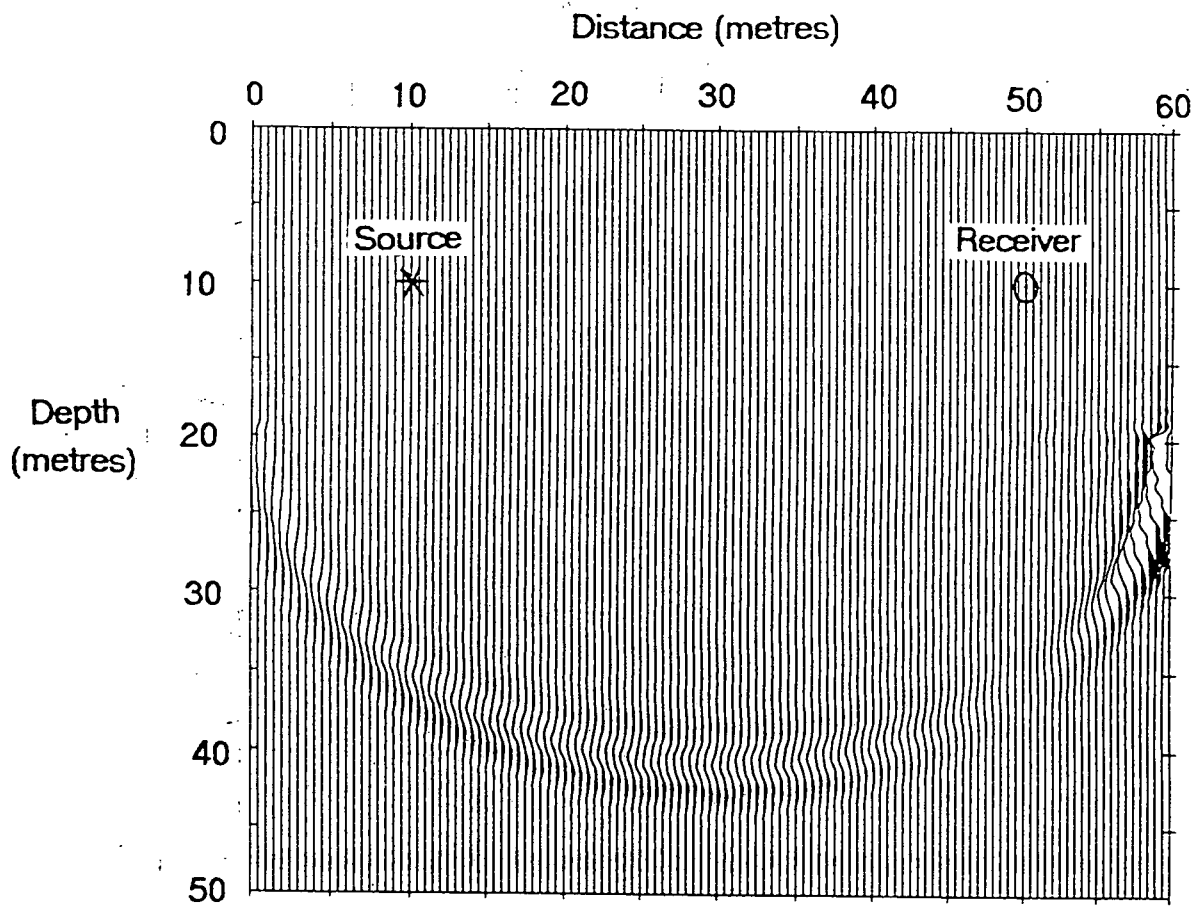


Figure 6.3 Kirchhoff migration operator in a homogeneous, isotropic velocity field.

The finite difference migration is shown in figure 6.1. Absorbing boundary conditions were used at the grid edges. The impulse response is an ellipse with the source and receiver at the focal points and is symmetrical with respect to the source and receiver. This ellipse may be thought of as the locus of possible locations for a diffracting point to produce an arrival corresponding to the input spike at 20 ms. Reverberations within the ellipse are caused by grid dispersion effects (see chapter IV). The finite difference method does not implicitly differentiate between upgoing and downgoing events and so a complete ellipse has been generated.

The result of diffraction stack migration (i.e. without an amplitude and phase correction) is shown in figure 6.2. Again, the impulse response is an ellipse (in this case a half-ellipse as only upgoing wavefields have been considered by the operator) and it is also symmetrical with respect to exchange of the source and receiver. There are fewer reverberations than were obtained by the finite-difference method, although the wavelet of the ellipse is less smooth.

The Kirchhoff migration operator is shown in figure 6.3. In this case the Newman phase and amplitude correction has been applied, but the ellipse is not symmetrical to the exchange of source and receiver positions because of a term proportional to $\sqrt{\frac{R_s}{R_r}}$ (see equation 4.10 of chapter IV) which magnifies the reflectivity near the receiver position.

The Generalised Kirchhoff migration operator is shown in figure 6.4. This operator is similar to the Kirchhoff operator but is symmetrical to the exchange of source and receiver positions because it contains the term $\sqrt{\frac{R_s}{R_r}}$ for integration over the receiver array as well as the term $\sqrt{\frac{R_r}{R_s}}$ for integration over the source array

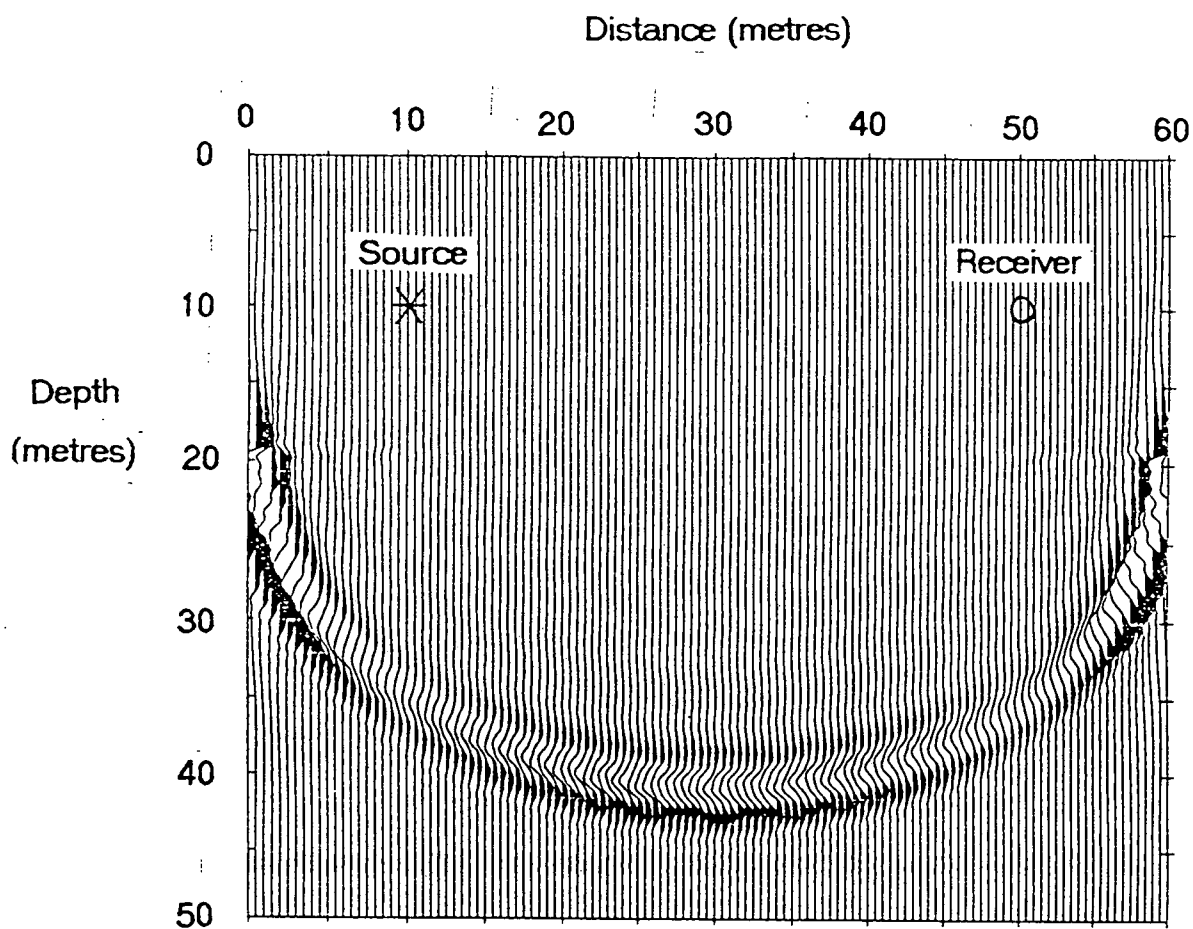


Figure 6.4 Generalised Kirchhoff migration operator in a homogeneous, isotropic velocity field.

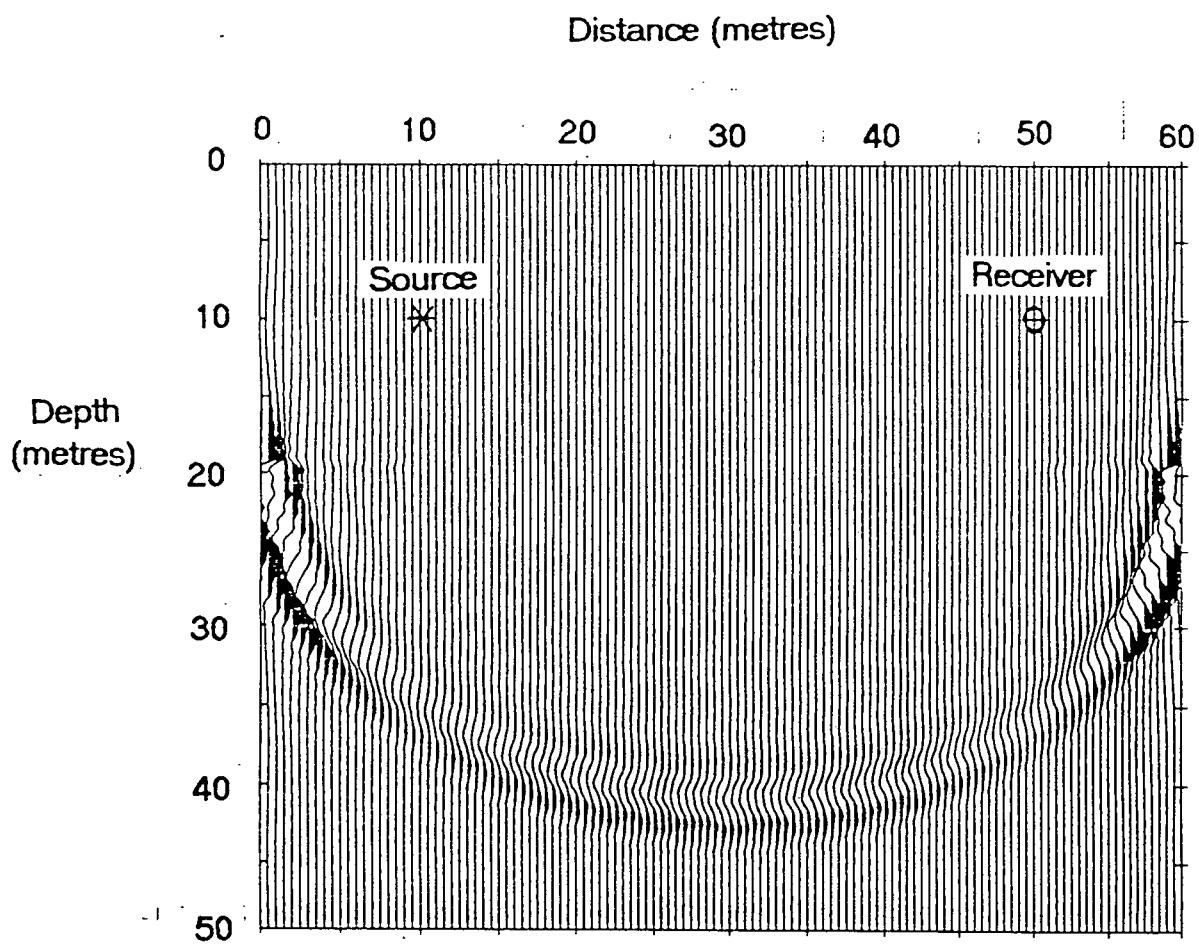


Figure 6.5 Generalised Radon Transform migration operator in a homogeneous, isotropic velocity field.

(see equation 4.11 of chapter IV). The high amplitudes to the left of the source and to the right of the receiver may be removed by a migration dip aperture taper in areas where the geology is known to be reasonably horizontal.

The Generalised Radon Transform (GRT) migration operator (Miller et al., 1987) is shown in figure 6.5. The integral is similar to that of the G-K operator but includes an additional term which depends on the scattering angle ϕ between the source and receiver raypaths at the scattering point. The term is proportional to $\cos^2 \frac{\phi}{2}$ which results in an almost constant amplitude in the part of the ellipse which lies between the source and the receiver. Again there are high reflectivities for the steeply dipping events beyond the source and receiver positions.

6.2.1 Suitability of impulse response functions for cross-hole data

Clearly the Kirchhoff migration method is unsuitable for the migration of cross-hole data due to the large variations in reflectivity produced between the source and receiver boreholes. The finite difference method is very slow and results in reverberations caused by grid dispersion effects. The diffraction stack operator does not include the theoretically correct amplitude and phase corrections provided by Kirchhoff migration. There is little to choose between the G-K and GRT migration operators. Both produce near-uniform reflectivities in the region of interest between the source and the receiver. In this work the G-K migration operator was used along with suitable controls for the dip aperture to enhance reflections from shallow-dipping strata.

Chapter VII

Conclusions and suggestions for future work

A field method for acquiring cross-hole seismic reflection data in shallow strata has been developed. The technique as it stands allows two or three cross-hole surveys to be acquired by a field crew of two or three personnel in one day. Further improvements in acquisition rates may be obtained by the use of a borehole sparker or piezoelectric source instead of a dynamite source.

Cross-hole direct arrival data may be processed using tomographic imaging software as a supplement to the cross-hole reflection method. The velocity field obtained by a tomographic survey, whilst not being successful at imaging faults or old mineworkings between boreholes (Kragh, 1990), is useful in providing a starting model for migration of the cross-hole seismic reflection data.

The cross-hole reflection method provides high-resolution seismic reflection sections of shallow Coal Measures strata. The processing of several trial surveys acquired at British Coal Opencast's exploration sites in the north of England has shown that wavelengths less than 4m are typical for these surveys with recorded reflection data frequencies lying in the bandwidth 100-500 Hz. The most pronounced reflections come from the coal seams.

It is envisaged that, for a practical exploration technique, the cross-hole reflection method should be used in conjunction with hole-to-surface surveys (Kragh, 1990). The zones of reflector coverage of the two types of survey are complementary (see figure 7.1). The cross-hole method provides better imaging in the region

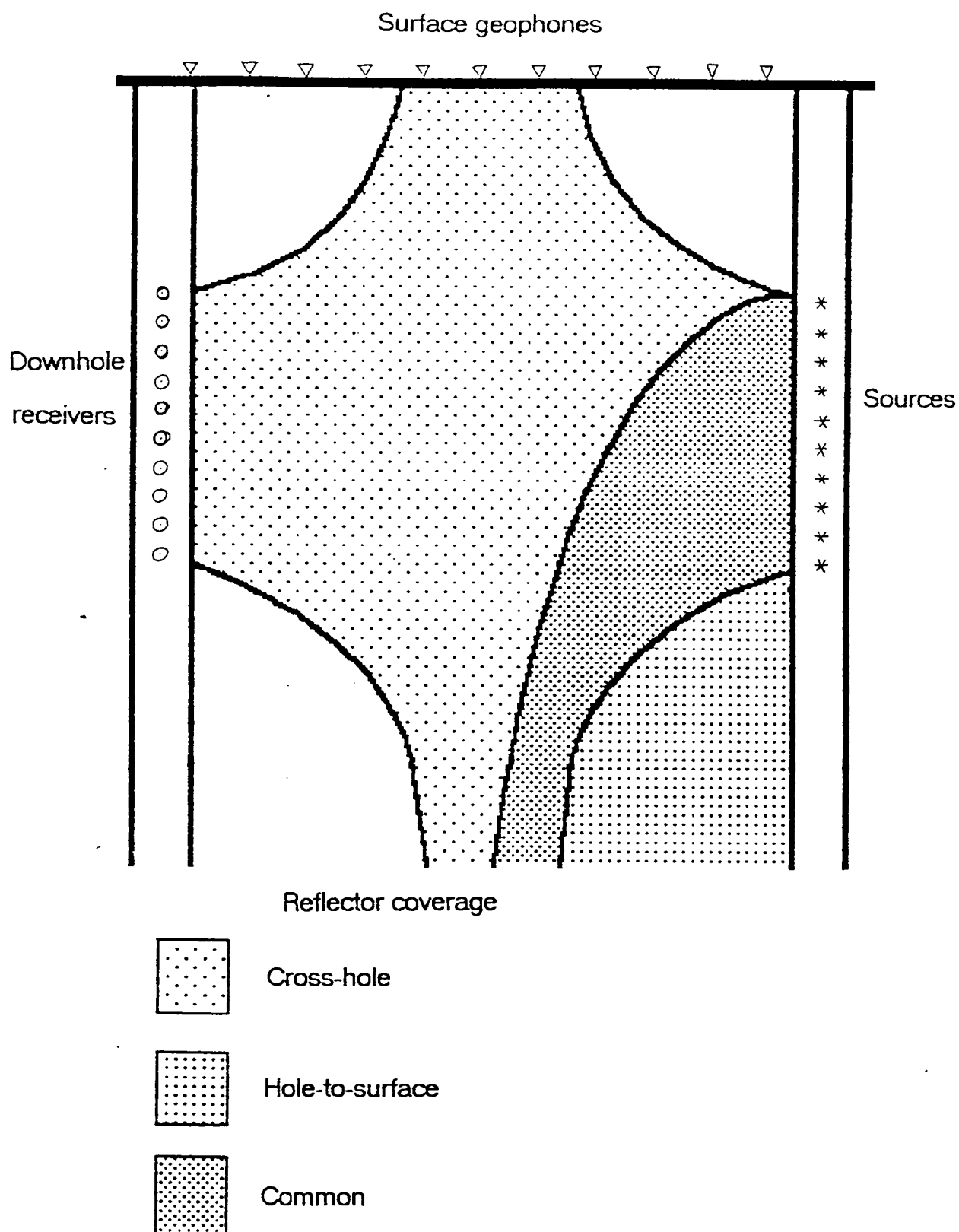


Figure 7.1 Zones of coverage of cross-hole and hole-to-surface reflection surveys with the shallowest source and down-hole receivers located near the water table.

between the boreholes, but the image quality deteriorates towards the boreholes. The hole-to-surface method provides lower resolution sections (wavelengths are around double that of the cross-hole method) but better imaging near the boreholes. The cross-hole image quality nearer the boreholes should be improved by the inclusion of a near-field correction term to the Kirchhoff integral (Dillon, 1988).

Possible uses of the cross-hole reflection method include the detailed imaging of regions of particular interest such as small-scale faulting at reservoir depths where borehole logging, conventional seismic and VSP methods cannot provide sufficient resolution. It should also be possible to acquire data from surveys beneath obstructions such as buildings, roads and water courses where surface sources and receivers cannot be used.

In addition to imaging "conventional" upgoing reflections, the cross-hole method also gives reflector coverage above the top source and receiver positions. An important characteristic of the cross-hole method is that the frequency content of the data is not effected by the depth of the region of interest as is the case with surface surveys and VSPs, but rather by the separation of the boreholes and therefore we might expect images of deeper strata to be every bit as good as those of the shallow strata included in this work. In fact the greater compaction of rocks at depth should result in better quality data.

Further important development work could include the use of elastic imaging by means of a downhole three-component geophone string. The 24-channel seismograph in use at Durham University would allow a string of eight such geophone units to be employed. This would necessitate further software development in order to implement elastic wave Generalised Kirchhoff migration. The consequential increase in computing costs would be offset by an improvement in imaging,

particularly as tube wave arrivals and other noise events should be more easily suppressed. Beattie's (1990) comparative study showed that tube waves are a bigger problem for geophones than for hydrophones. However, this might be overcome by modifying the geophone design. Also, with a string of downhole geophones, the top and bottom geophones might act as baffles, thus reducing the amplitudes of tube waves on the other geophones.

Borehole blockages restrict the coverage of sources and receivers, especially in those areas of particular interest where small faults and old mineworkings exist. This problem may best be solved by the use of a plastic casing along the full length of the borehole. Plastic is most suitable for ease of use and because the casing must be expendable since there is a high probability that the casing will become stuck in the borehole. However, it is also essential that the tubing is of sufficient strength to withstand borehole collapses and explosions from small seismic sources and also flexible enough to follow the curvature of the borehole.

The restriction in source and receiver coverage imposed by the depth of the water table might be removed by the development of a borehole "packer" which could block the borehole at a shallow depth and allow water to be poured in to fill the borehole.

Finally, more test surveys will be needed to evaluate the above suggestions and, in particular to obtain images of old mineworkings and small faults.

References

- Alford R.M., Kelly K.R., and Boore D.M. "Accuracy of finite-difference modeling of the acoustic wave equation." *Geophysics* 39,834-842, 1974.
- Balch A.H., Lee M.W., Miller J.J. and Ryder R.T. "The use of vertical seismic profiles in seismic investigations of the earth." *Geophysics* 47, 906-918, 1982.
- Balch A.H., Chang H., Hofland G. and Ranzinger K. "The use of forward and back-scattered P-, S- and converted waves in cross-borehole imaging." Presented at the 52nd Annual Meeting of the EAEG, Copenhagen, 1990.
- Beattie K.E. "A comparison of receiver types in crosshole seismics." *M.Sc. thesis, University of Durham*, 1990.
- Bishop I. and Styles P. "Seismic tomographic imaging of a buried concrete target." *Geophysical Prospecting* 38, pp169-188, 1990.
- Brabham P.J. "The application of seismic surveys to the evaluation of shallow coal deposits." *Ph.D. Thesis, University of Durham*, 1986.
- Bredewout J.W. and Goult N.R. "Some shallow seismic reflections." *First Break* 4(12) pp15-23, 1986.
- Buchanan D.J. "In-seam seismology: A method for detecting faults in coal seams." *Developments in Geophysical Exploration Methods-5 Chapter 1, Editor A.A. Fitch, Applied Science Publishers*, pp 1-34, 1983.
- Chang and McMechan, "Reverse-time migration of offset vertical seismic profiling data using the excitation-time imaging condition." *Geophysics* 51, pp67-84, 1986.

Christie P.A.F., Hughes V.J. and Kennett B.L.N. "Velocity filtering of seismic reflection data." *First Break* 1(3) pp9-21, 1983.

Claerbout J.F. "Towards a unified theory of reflector mapping." *Geophysics* 36 pp467-481, 1971

Claerbout J.F. "Fundamentals of Geophysical Data Processing: with Applications to Petroleum Prospecting." *McGraw-Hill*, 1976.

Clayton R. and Engquist B. "Absorbing boundary conditions for acoustic and elastic wave equations." *Bulletin of the Seismological Society of America*, 67, 1529-1540, 1977.

Cooley J.W. and Tukey J.W. "An algorithm for the machine calculation of complex Fourier series." *Maths computations* 19, pp297-301, 1965.

Cottin J.-F., Deletie P., Jacquet-Francillon H., Lakshmanan J., Lemoine Y., Sanchez M. "Curved ray seismic tomography: application to the Grand Etang Dam (Reunion Island)." *First Break* 4(7), 25-30, 1986.

Devaney A.J. "Geophysical diffraction tomography." *IEEE Transactions on Geoscience and Remote Sensing* GE-22, 3-13, 1984.

Dillon P.B. "Vertical seismic profile migration using the Kirchhoff integral." *Geophysics* 53, 786-799, 1988.

Dillon P.B. "A comparison between Kirchhoff and GRT migration on VSP data." *Geophysical prospecting* 38, 757-777, 1990.

Dillon P.B. and Thomson R.C. "Offset source VSP surveys and their image reconstruction." *Geophysical Prospecting* 32, 790-811, 1984.

Dines K.A. and Lytle R.J. "Computerised Geophysical Tomography." *Proceedings of the IEEE* 67,7, July 1979.

Dyer B.C. "Seismic reflection and transmission tomography." *Ph.D. dissertation, University of London*, 1988

Embree P., Burg J.P. and Backus M.M. "Wide-band velocity filtering - The pie-slice process." *Geophysics* 28 ,948-974, 1963.

Findlay M.J. "A curved ray iterative technique for tomographic inversion of cross-borehole data." *M.Sc. Thesis, University of Durham*, 1987.

French W.S. "Computer migration of oblique seismic reflection profiles." *Geophysics* 40, 961-980, 1975.

Goulty N.R., Thatcher J.S., Findlay M.J., Kragh J.E. and Jackson P.D. "Experimental investigation of crosshole seismic techniques for shallow coal exploration." *Quarterly Journal of Engineering Geology* 23, 217-228, 1990.

Hardage B.A. "Vertical Seismic Profiling part A: Principles." *Geophysical Press*, 1983.

Howson M. and Sides E.J. "Borehole desurvey calculation." *Computers and Geosciences* 12 ,97-104, 1986.

Hu L.-Z., McMechan G.A. and Harris J.M. "Acoustic prestack migration of cross-hole data." *Geophysics* 53, 1015-1023, 1988.

Jackson P. "Horizontal seismic in coal seams: its use in the UK coal industry." *First Break* 3(11), 15-24, 1985.

Jackson P.J., Onions K.R. and Westerman A.J. "Use of inverted VSP to enhance the exploration value of boreholes." *First Break* 7, 233-246, 1989.

Justice J.H., Vassilou A.A., Singh S., Logel J.D., Hansen P.A., Hall B.R., Hutt P.R. and Solanki J.J. "Acoustic tomography for monitoring enhanced oil recovery." *The Leading Edge* 8(2), 12-19, 1989.

Kohler K. and Koenig M. "Reconstruction of reflecting structures from vertical seismic profiles with a moving source." *Geophysics* 51, 1923-1938, 1986.

Kragh J.E. "Borehole seismic methods for opencast coal exploration." *Ph.D. Thesis, University of Durham*, 1990.

Lee W.H.K and Stewart S.W. Principles and applications of microearthquake networks. *Pub. Academic Press Inc.*, 1981.

Levin F.K. "The reflection, refraction, and diffraction of waves in media with an elliptical velocity dependence." *Geophysics* 43, 528-537, 1978.

McMechan G.A. "Synthetic finite-offset vertical seismic profiles for laterally varying media." *Geophysics* 50, 627-636, 1985.

Macrides C.G., Kanasevich E.R. and Bharatha S. "Multiborehole seismic imaging in steam injection heavy oil recovery projects." *Geophysics* 53, 65-75, 1988.

March D.W. and Bailey A.D. "A review of the two-dimensional transform and its use in signal processing." *First Break* 1(1), 9-21, 1983.

Miller D., Oristaglio M., and Beylkin G. "A new slant on seismic imaging: migration and integral geometry." *Geophysics* 52, 943-964, 1987.

Mitchell A.R. Computational methods in partial differential equations. *Pub. John Wiley and Sons, Inc.*, 1969.

Pratt R.G. and Goult N.R. "Combining wave equation imaging with traveltime tomography to form high resolution images from crosshole data." *Geophysics* 56, 208-224, 1991.

Pratt R.G. and Worthington M.H. "The application of diffraction tomography to cross-hole seismic data." *Geophysics* 53, 1284-1294, 1988.

Renaut R.A. and Petersen J. "Stability of wide-angle absorbing boundary conditions for the wave equation." *Geophysics* 54, 1153-1163, 1989.

Robinson E.A. and Treitel S., *Geophysical Signal Analysis, Prentice-Hall, New Jersey*, 1985.

Schneider W.A. "Integral formulation for migration in two and three dimensions." *Geophysics* 43(1), 49-76, 1978.

Schoenberg M. "Fluid and solid motion in the neighbourhood of a fluid-filled borehole due to the passage of a low-frequency elastic plane wave." *Geophysics* 51, 1191-1205, 1986.

Stewart R.R. "VSP interval velocities from traveltime inversion." *Geophysical Prospecting* 32, 608-628, 1984.

Sun R. and McMechan G.A. "Pre-stack reverse-time migration for elastic waves with application to synthetic offset vertical seismic profiles" *Proc. IEEE*, 74, 457-465, 1986.

Suprajitno M. and Greenhalgh S.A. "Separation of upgoing and downgoing waves in vertical seismic profiling by contour slice filtering." *Geophysics* 50, 950-962, 1985.

Van der Poel N.J. and Cassell B.R. "Borehole seismic surveys for fault delineation in the Dutch North Sea." *Geophysics*, 54, 1091-1100, 1989.

Worthington M.H. "An introduction to geophysical tomography." *First Break* 2(11), 20-25, 1984.

Wye R. "An investigation into the accuracy of algebraic reconstruction techniques for tomographic inversion." *M.Sc. Thesis, University of Durham*, 1986.

Zhu X. and McMechan G.A. "Acoustic modeling and migration of stacked cross-hole data" *Geophysics* 53, 492-500, 1988.

Ziolkowski A.M. "Seismic profiling for coal on land." in *Developments in Geophysical Exploration Methods* edited by A.A.Fitch vol.1, *Applied Science Publishers Ltd., London*, 1979.

Ziolkowski A.M., Lerwill W.E., March D.W. and Peardon L.G. "Wavelet deconvolution using a source scaling law." *Geophysical Prospecting* 28, 872-901, 1980.



Appendix A

Computer software

A number of computer programs were written during the course of this research. These include software for data-processing, migration and plotting.

The programs are all written in IBM-Fortran77 to run on the University of Durham Amdahl 470/V8 mainframe computer. Program listings are included in the following pages of this appendix. External subroutines used include the Culham Laboratory GHOST*80 graphical subroutine library as well as a library of time series analysis subroutines (TSAR4_L in user-identifier GLK4) taken from a variety of different sources. Michigan Terminal System (MTS) routines are used for handling data stored on magnetic tape.



Appendix A.1

Data processing software

The data processing program is menu driven with a main menu providing access to a series of submenus which carry out operations such as data input/output, data plotting, Fourier transforms, filtering, 2D Fourier transforms, waveshaping deconvolution, first break picking and direct arrival muting. Data may be plotted interactively on screen or sent to plotfiles.

```

C      PROGRAM XHR1
C
C      Written by M J Findlay 1987-1990
C
C      Reads seismic data from tape or disc in variety of formats
C      Dataprocessing software all Menu driven
C      Uses subroutines libraries
C      TSAR4_L (Time series analysis -- Robinson, Claerbout et al.)
C      XHR_LIB (Dataprocessing, display and file I/O -- Findlay)
C      *ghost80 (graphics library)
C      *nag (graphics and other routines)
C      Requires defaults file XHR.DEFAULT to be attached to unit0 at runtime.
C
C
C      PARAMETER (NSAMR=2048,NCHR=65)
C      CHARACTER ANSMR*1, OPFORM*6, IPDISC*17, OPDISC*17
C      CHARACTER IPSOUR*4, INPORT*3, TITOFF*3, CHNSPL*26, POLART*7,
C      1 VARE*3
C      CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECLOC,
C      1 COMSHT
C      CHARACTER*20 FILREC, A(9)
C      CHARACTER*25 PROC(20)
C      INTEGER INPCHN, NSHOT, LEN, NFILES, NFILE(2), NRECS, NCR(2)
C      INTEGER NFIRST(NCHR), IDPROC(5,NCHR), NPLST, NPLST, IOPT, NPROCS
C      REAL SORPOS, RECDEP(NCHR), DT, R4DAT(NSAMR,NCHR), DBGAIN(NCHR)
C      REAL GCMSC(1,NCHR), FLOWCT, FHICUT, FLOWFF, FHIOFF, DBLOW, DBHIGH
C      INTEGER LEDGE, LENFIL
C      EQUIVALENCE (LOCATE,A(1)), (DATE,A(2)), (DEVICE,A(3)),
C      1 (SORTYP,A(4)), (SORLOC,A(5)), (RECTYP,A(6)),
C      2 (RECLOC,A(7)), (COMSHT,A(8)), (FILREC,A(9))
C
C      Set up initial default values
C      READ IN FROM FILE ATTACHED TO UNIT 0
C
C      READ (0,10) LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECLOC,
C      1 COMSHT, FILREC, INPORT, IPDISC, OPDISC, OPFORM, IPSOUR
C      10 FORMAT (A20, 8(/A20)/A3/A17/A17/A6/A4)
C      READ (0,30) NSAM1, NCH1, NFILES, NFILE(1), NFILE(2), INPCHN,
C      1 NCR(1), NCR(2), SORPOS, TOPREC, RECSEP, DT, DZ, SCAL
C      IF (NSAM1.NE. NSAMR .OR. NCH1.NE. NCHR) THEN
C      1 WRITE (6,*) 'ARRAY SIZE INCOMPATIBILITY NSAM1,NCH1 =', NSAM1,
C      1 NCH1
C      1 WRITE (6,*) '
C      1 NCHR
C      1 NSAMR,NCHR =', NSAMR,
C      1 NCHR
C      1 WRITE (6,*) '
C      1 WRITE (6,*)
C      1 WRITE (6,*) MUST EDIT FILES XHR.1_U XHR.2_U & XHR.3_U.'
C      1 WRITE (6,*) CHANGE PARAMETER STATEMENTS IN SUBROUTINES '
C      1 WRITE (6,*) XHR.1 (XHR.1_U)
C      1 WRITE (6,*) XHR.2 (XHR.2_U)
C      1 WRITE (6,*) R0FILE,MENPL1,MEDFIL (XHR.2_U)
C      1 WRITE (6,*) MENUD,MENUE,MENUF,WVSHAP,GNCOMP (XHR.3_U)
C      1 WRITE (6,*)
C      1 WRITE (6,*)
C      1 STOP
C
C      END IF
C      20 FORMAT (I4/I2/I1/I3/I2/I2/I2/F6.2/F6.2/F6.2/F6.2/F6.2/F6.2)
C      READ (0,40) TITOFF, CHNSPL, POLART, VARE, NPLST, NPLST, IOPT,
C      1 FLOWCT, LEDGE, LENFIL, TITLE, XLB, YLB
C      40 FORMAT (A3/A26/A7/A3/I4/I4/I2/F6.2/I2/I2/A30/A30/A30)
C      RECDEP(1) = TOPREC
C      DO 50 I = 2, NCHR
C      1 IF (I.EQ. NCR(2)) THEN
C      1 RECDEP(I) = RECDEP(I - 1)
C      1 ELSE
C      1 RECDEP(I) = RECDEP(I - 1) + RECSEP
C      1 END IF
C      50 CONTINUE
C      FHICUT = 0.5E6 / DT
C      IF (NFILES.EQ. 1) NSHOT = NFILE(1)
C      DO 60 I = 1, NCHR
C      1 NPROCS = 1
C      1 IDPROC(I,I) = 0
C      1 NFIRST(I) = 0
C      1 DBGAIN(I) = 0.0
C      1 GCMSC(I) = 1.0
C      60 CONTINUE
C      IDPROC(1,NPROCS) = 1
C
C      DT in microseconds, SAMFRQ in kHz
C
C      WRITE (6,*)
C      1 WRITE (6,*) NIMBUS TAPE DATA MANIPULATION PROGRAM'
C      1 WRITE (6,*)
C      1 WRITE (6,*)
C      1 CALL PAPER(1)
C      1 CALL PSPACE(0.10, 0.9, 0.1, 0.75)
C      1 CALL MAP(0.0, 1.0, 0.0, 1.0)
C      1 CALL J06XAF
C      1 CALL J06XAF(0.0, 1.0, 0.0, 1.0)
C      1 CALL J06XFF(2)
C      1 CALL J06XGF(1.0, 2.0)
C      1 CALL J06AHF('XHR')
C      1 CALL J06WZF
C      1 CALL GREND
C
C      ~~~~~
C      ~ M J FINDLAY 1/JUNE/88 ~
C      ~~~~~
C      Verify that I/O parameters are OK
C
C      CALL MENUIO(NSAMR, NCHR, NRECS, NSHOT, LEN, NFILES, NFILE, IPDISC,
C      1 IPSOUR, INPORT, OPDISC, OPFORM, INPCHN)
C      Read in data from files NFILE(1&2) to array R4DAT(1024,NCHR)
C
C      CALL R0FILE(NSAMR, NCHR, NRECS, R4DAT, NFIRST, RECDEP, DBGAIN,
C      1 GCMSC, NFILES, NFILE, NCR, NSHOT, DT, LEN, SORPOS, NPROCS,
C      2 IDPROC, IPDISC, IPSOUR, INPORT, INPCHN, A, PROC)
C      WRITE (6,*) NPROCS
C
C      Main menu
C
C      CALL POWER2(NSAMR, NSAMP2)

```

```

70 WRITE (6,*)
   WRITE (6,*)'          MASTER MENU          (PRESS 0 TO EXIT)'
   WRITE (6,*)'          ~~~~~          '
   WRITE (6,*)'
1  Read data from file(s)          '
   WRITE (6,*)'
2  Write contents of R4DAT to file  '
   WRITE (6,*)'
3  Plot contents of R4DAT          '
   WRITE (6,*)'
4  FFT traces and plot spectra     '
   WRITE (6,*)'
5  Apply filters to traces         '
   WRITE (6,*)'
6  F-K transform traces           '
   WRITE (6,*)'
7  First breaks, ramps, AGC, tapers etc '
   WRITE (6,*)'
17 Interactively interpolate saturations '
   WRITE (6,*)'
8  Apply wavelet shaping filter    '
   WRITE (6,*)'
9  Adjust I/O parameters          '
   WRITE (6,*)'
10 Adjust display parameters      '
   WRITE (6,*)'
11 Adjust processing record       '
   WRITE (6,*)'
12 Output first break times      '
   WRITE (6,*)'
   READ (5,*) IANSWR
   IF (NPROCS .GT. 20 .OR. NPROCS .LT. 0) NPROCS = 0
   IF (IANSWR.EQ. 1) CALL RDFILE(NSAMR, NCHR, NRECS, RADAT, NFRST,
1  RECDP, DBGAIN, GCMSC, NFILES, NFILE, NCR, NSHOT, DT, LEN,
2  SORPOS, NPROCS, IDPROC, IPSOUR, INPOPT, INPCHN, A,
3  PROC)
   WRITE (6,*) NPROCS, A
   IF (IANSWR.EQ. 2) CALL WRFILE(NSAMR, NCHR, RADAT, NFRST, OPFORM,
1  OPDISC, NSHOT, LEN, LOCATE, DATE, DEVICE, SORTYP, SORLOC,
2  RECTYP, RECLOC, COMSHT, FILREC, SORPOS, NRECS, RECDP,
3  DBGAIN, GCMSC, NCR, NPROCS, IDPROC, DT, .OPCHN, PROC)
   IF (IANSWR.EQ. 3) WRITE (6,*)'NSAMR,NCHANS =', NSAMR, NRECS
   IF (IANSWR.EQ. 3) CALL MENUP2(NSAMR, NRECS, RADAT, NFRST,
1  TIOFF, CHNSPL, POLART, VARARE, NPLFST, NPLST, IOPT, SORPOS,
2  RECDP, NCR, DT, LOCATE, DATE, DEVICE, SORTYP, SORLOC,
3  RECTYP, RECLOC, COMSHT, FILREC, NSHOT, NPROCS, IDPROC, TITLE,
4  XLB, YLB, SCAL, PROC)
   IF (IANSWR.EQ. 4) CALL MENUD(NSAMR, NCHR, RADAT, NTRACE, DT,
1  NSHOT)
   IF (IANSWR.EQ. 5) CALL MENUE(NSAMR, NCHR, RADAT, DT, FLOWCT,
1  FHCUT, FLOOFF, FHCOFF, DBLOW, DBHIGH, FILTAP, BLHPAS,
2  LENFIL, NPROCS, IDPROC)
   IF (IANSWR.EQ. 6) CALL MENUF(NSAMR, NCHR, RADAT, DZ, DT, NCR)
   IF (IANSWR.EQ. 7) CALL GNCOMP(NSAMR, NCHR, RADAT, NFRST, DBGAIN,
1  GCMSC, RECDP, NCR, IDPROC, NPROCS)
   IF (IANSWR.EQ. 17) CALL GRAINT(NSAMR, NRECS, NSHOT)
   IF (IANSWR.EQ. 8) CALL WSHAP(NSAMR, NCHR, RADAT, NFRST, NCR,
1  DT)
   IF (IANSWR.EQ. 9) CALL MENUIO(NSAMR, NCHR, NRECS, NSHOT, LEN,
1  NFILES, NFILE, IPDISC, IPSOUR, INPOPT, OPDISC, OPFORM,
2  INPCHN)
   IF (IANSWR.EQ. 10) CALL MENUDS(NCHR, SORPOS, RECDP, NCR, DT,
1  LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECLOC, COMSHT,
2  FILREC, TITLE, XLB, YLB)
   IF (IANSWR.EQ. 11) CALL PROCES(NPROCS, PROC)
   IF (IANSWR.EQ. 12) CALL OPFIRS(NCHR, NFRST, RECDP, LEN, IPDISC)
   IF (IANSWR.EQ. 0) CALL EXIT

```

```

DZ = ABS(RECDP(2) - RECDP(1))
IF (DZ.EQ. 0.0) DZ = 1.0
GO TO 70
END

C
SUBROUTINE MENUIO(NSAMR, NCHR, NRECS, NSHOT, LEN, NFILES, NFILE,
1  IPDISC, IPSOUR, INPOPT, OPDISC, OPFORM, INPCHN)
C
C Menu for adjustment of I/O parameters
C
DIMENSION NFILE(2)
CHARACTER IPDISC*17, OPDISC*17, OPFORM*6, IPSOUR*4, INPOPT*3
10 WRITE (6,*)
   WRITE (6,*)'          I/O Setup Menu          '
   WRITE (6,*)'          ~~~~~          '
   WRITE (6,20) IPSOUR
20 FORMAT (/) 1 Input Source (Tape/Disc) :', A4)
   WRITE (6,30) IPDISC
30 FORMAT (/) 2 Input Disc Name :', A17)
   WRITE (6,40) OPDISC
40 FORMAT (/) 3 Output Disc Name :', A17)
   WRITE (6,50) OPFORM
50 FORMAT (/) 4 Output Format :', A6)
   WRITE (6,60)
60 FORMAT (/) 14 SEG-Y output to tape :')
   WRITE (6,70) NSAMR
70 FORMAT (/) 5 Record Length (samples) :', I4)
   (4 bytes / sample)
   LEN = (NSAMR + 2) * 4
   WRITE (6,80) NCHR
80 FORMAT (/) 6 Number of channels :', I3)
   WRITE (6,90) NSHOT
90 FORMAT (/) 7 Shot I.D. number :', I3)
   WRITE (6,100) INPOPT
100 FORMAT (/) 8 Input method :', A3)
   IF (INPOPT.NE.'NEW') THEN
   WRITE (6,110) (NFILE(I), I=1, NFILES)
110 FORMAT (/) 9 I.D. nos. of files :', 1X, I2, 1X,
1  I2)
   END IF
   WRITE (6,120)
120 FORMAT (/) 0 Return to MAIN MENU :')
   READ (5,*) IANSWR
130 FORMAT (A17)
   IF (IANSWR.EQ. 1) THEN
   IF (IPSOUR.EQ.'TAPE')
   IPSOUR = 'DISC'
   INPCHN = 1
   ELSE
   IPSOUR = 'DISC'
   INPCHN = 10
   END IF
   END IF
   IF (IANSWR.EQ. 2) THEN
   WRITE (6,*)
   READ (5,130) IPDISC
   Input Disc name :')
   END IF
   IF (IANSWR.EQ. 3) THEN
   WRITE (6,*)

```

```

WRITE (6,*) '      Output Disc name      '
READ (5,130) OPDISC
END IF
IF (IANSWR.EQ. 4) THEN
  IF (OPFORM.EQ. 'DIRECT') THEN
    OPFORM = 'SEQUEN'
  ELSE
    OPFORM = 'DIRECT'
  END IF
END IF
IF (IANSWR.EQ. 14) THEN
  OPFORM = 'SEG-Y'
END IF
IF (IANSWR.EQ. 5) THEN
  WRITE (6,*)
  WRITE (6,*) '      Number of samples      '
  READ (5,*) NSAMR
END IF
IF (IANSWR.EQ. 6) THEN
  WRITE (6,*)
  WRITE (6,*) '      Number of channels      '
  READ (5,*) NCHR
END IF
IF (IANSWR.EQ. 7) THEN
  WRITE (6,*) '      Shot I.D.      '
  READ (5,*) NSHOT
END IF
IF (IANSWR.EQ. 8) THEN
  WRITE (6,*) '1      DIRECT ACCESS FILE      '
  WRITE (6,*) '2      SEQUENTIAL SEG-D TAPE FILES      '
  WRITE (6,*) '3      1 SEQUENTIAL SEG-Y TAPE FILE      '
  READ (5,*) IREPLY
  IF (IREPLY.EQ. 1) THEN
    INOPT = 'NEW'
  ELSE IF (IREPLY.EQ. 2) THEN
    INOPT = 'OLD'
  ELSE
    INOPT = 'SGY'
  END IF
END IF
IF (IANSWR.EQ. 9 .AND. INOPT.EQ. 'OLD') THEN
  WRITE (6,*) '      No. of files      '
  READ (5,*) NFILES
  IF (NFILES.LT. 1 .OR. NFILES.GT. 2) GO TO 140
  DO 150 I = 1, NFILES
    WRITE (6,*) 'I.D. File #', I, ' : '
    READ (5,*) NFILE(I)
  150 CONTINUE
END IF
IF (INOPT.NE. 'OLD') NFILES = 1
NRECS = NFILES * NCHR
IF (IANSWR.NE. 0) GO TO 10
RETURN
END

C
1 SUBROUTINE MENUDS(NRECS, SORDEP, RECDER, NCR, DT, LOCATE, DATE,
2   DEVICE, SORTYP, SORLOC, RECTYP, RECLC, COMSHT, FILREC,
C Menu for adjustment of Display parameters

C CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECLC,
1 COMSHT
CHARACTER*20 FILREC
CHARACTER*30 TITLE, XLB, YLB
DIMENSION RECDER(NRECS), NCR(2)
10 WRITE (6,*)
  WRITE (6,*) '      Display Setup Menu (0 to EXIT)      '
  WRITE (6,*) '      *****      '
  WRITE (6,20) LOCATE
20 FORMAT (//, 1 Location
30 FORMAT (//, 2 Acquisition date
30 FORMAT (//, 2 Acquisition date
40 FORMAT (//, 3 Recording instrument
40 FORMAT (//, 3 Recording instrument
50 FORMAT (//, 4 Source hole I.D.
50 FORMAT (//, 4 Source hole I.D.
60 FORMAT (//, 5 Source type
60 FORMAT (//, 5 Source type
70 FORMAT (//, 6 Source depth (metres)
70 FORMAT (//, 6 Source depth (metres)
80 FORMAT (//, 7 Receiver hole I.D.
80 FORMAT (//, 7 Receiver hole I.D.
90 FORMAT (//, 8 Receiver type
90 FORMAT (//, 8 Receiver type
100 FORMAT (//, 9 Receiver depths (metres)
100 FORMAT (//, 9 Receiver depths (metres)
110 FORMAT (//, 10 Common shot traces
110 FORMAT (//, 10 Common shot traces
120 FORMAT (//, 11 Sampling rate (microsecs)
120 FORMAT (//, 11 Sampling rate (microsecs)
130 FORMAT (//, 12 Analogue filters
130 FORMAT (//, 12 Analogue filters
140 FORMAT (//, 13 Common Receiver Channels
140 FORMAT (//, 13 Common Receiver Channels
150 FORMAT (//, 14 Plot-title and axes labels
150 FORMAT (//, 14 Plot-title and axes labels
1 READ (5,*) IANSWR
160 FORMAT (A20)
170 FORMAT (A32)
IF (IANSWR.EQ. 1) THEN
  WRITE (6,*) '      Location      '
  READ (5,160) LOCATE
ELSE IF (IANSWR.EQ. 2) THEN
  WRITE (6,*) '      Date      '
  READ (5,160) DATE
ELSE IF (IANSWR.EQ. 3) THEN
  WRITE (6,*) '      Seismograph      '
  READ (5,160) DEVICE
ELSE IF (IANSWR.EQ. 4) THEN
  WRITE (6,*) '      Source hole      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 5) THEN
  WRITE (6,*) '      Source type      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 6) THEN
  WRITE (6,*) '      Source depth (metres)      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 7) THEN
  WRITE (6,*) '      Receiver hole I.D.      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 8) THEN
  WRITE (6,*) '      Receiver type      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 9) THEN
  WRITE (6,*) '      Receiver depths (metres)      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 10) THEN
  WRITE (6,*) '      Common shot traces      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 11) THEN
  WRITE (6,*) '      Sampling rate (microsecs)      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 12) THEN
  WRITE (6,*) '      Analogue filters      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 13) THEN
  WRITE (6,*) '      Common Receiver Channels      '
  READ (5,160) SORLOC
ELSE IF (IANSWR.EQ. 14) THEN
  WRITE (6,*) '      Plot-title and axes labels      '
  READ (5,160) SORLOC

```

```

WRITE (6,*)
WRITE (6,*) ' Source type : '
READ (5,160) SORTYP
ELSE IF (IANSWR.EQ. 6) THEN
WRITE (6,*)
WRITE (6,*) ' Source depth : '
READ (5,*) SORDEP
ELSE IF (IANSWR.EQ. 7) THEN
WRITE (6,*)
WRITE (6,*) ' Receiver hole: '
READ (5,160) RECLOC
ELSE IF (IANSWR.EQ. 8) THEN
WRITE (6,*)
WRITE (6,*) ' Receiver type: '
READ (5,160) RECTYP
ELSE IF (IANSWR.EQ. 9) THEN
WRITE (6,*)
WRITE (6,*) ' Rec depths : '
READ (5,*) (RECDEP(I),I=1,NRECS)
ELSE IF (IANSWR.EQ. 10) THEN
WRITE (6,*)
WRITE (6,*) ' Common shot traces : '
READ (5,160) COMSHT
ELSE IF (IANSWR.EQ. 11) THEN
WRITE (6,*)
WRITE (6,*) ' Sampling rate (microsecs) : '
READ (5,*) DT
ELSE IF (IANSWR.EQ. 12) THEN
WRITE (6,*)
WRITE (6,*) ' Analogue filters : '
READ (5,160) FILREC
ELSE IF (IANSWR.EQ. 13) THEN
WRITE (6,*)
WRITE (6,*) ' Common receiver traces : '
READ (5,*) NCR(1), NCR(2)
ELSE IF (IANSWR.EQ. 14) THEN
WRITE (6,*)
WRITE (6,*) ' Enter title and axes labels : '
READ (5,180) TITLE
READ (5,180) XLB
READ (5,180) YLB
FORMAT (A30)
180 END IF
IF (IANSWR.NE. 0) GO TO 10
RETURN
END

```

```

C
C Subroutine to amend/update processing record
C (All processing info is stored in data header)
C

```

```

SUBROUTINE PROCES(NPROCS, PROCS)
CHARACTER*25 PROCS(20)
10 WRITE (6,*)
WRITE (6,*) ' Process record editor '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,*) ' 1 CLEAR all process records '
WRITE (6,*) ' 2 Display all process records '

```

```

WRITE (6,*)
WRITE (6,*) ' 3 Display particular process record '
WRITE (6,*) ' 4 CHANGE particular process record '
WRITE (6,*) ' 5 ADD new process record '
WRITE (6,*) ' 0 EXIT '
WRITE (6,*)
READ (5,*) IANS
IF (IANS.EQ. 1) THEN
DO 20 II = 1, 20
PROCS(II) = ' '
20 CONTINUE
NPROCS = 0
ELSE IF (IANS.EQ. 2) THEN
DO 30 II = 1, NPROCS
WRITE (6,40) II, PROCS(II)
30 CONTINUE
40 FORMAT (/10X, I3, 4X, A25)
ELSE IF (IANS.EQ. 3) THEN
WRITE (6,*) 'Enter number of process record to display : '
READ (5,*) NPR
WRITE (6,40) NPR, PROCS(NPR)
ELSE IF (IANS.EQ. 4) THEN
WRITE (6,*) 'Enter number of process record to ALTER : '
READ (5,*) NPR
WRITE (6,40) NPR, PROCS(NPR)
WRITE (6,*) 'Enter text (<25 characters) to replace above : '
READ (5,50) PROCS(NPR)
50 FORMAT (A25)
ELSE IF (IANS.EQ. 5) THEN
NPROCS = NPROCS + 1
WRITE (6,*) 'Enter text (<25 characters for record ', NPROCS
READ (5,50) PROCS(NPROCS)
END IF
IF (IANS.NE. 0) GO TO 10
RETURN
END

```

```

C
C . SUBROUTINE TO WRITE OUT FIRST BREAK PICKS.
C
C .

```

```

SUBROUTINE OPFIRS(NCHR, NFIRST, RECDEP, LEN, IPDISC)
REAL, RECODEP(NCHR)
INTEGER NCHR, NFIRST(NCHR), LEN, IDOP(60)
CHARACTER IPDISC*17, A*180, OPNAME*17, TITLE*60, B*480
XS = 0.0
OPNAME = '-OUT'
10 WRITE (6,*)
WRITE (6,*) ' FIRST BREAK OUTPUT MENU '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,*) ' 1 OUTPUT ALL RECORDS IN FILE '
WRITE (6,*) ' 2 OUTPUT SELECTED RECORDS IN FILE '
WRITE (6,*) '
WRITE (6,20) OPNAME
20 FORMAT ( ' 3 OUTPUT FILENAME : ', A17)
WRITE (6,*)

```

```

WRITE (6,*)' 0 EXIT'
READ (5,*) IANS
IF (IANS.EQ.1) THEN
  WRITE (6,*)'Enter total number of records on disc : '
  READ (5,*) NOP
  DO 30 II = 1, NOP
    IDOP(II) = II
  CONTINUE
30 ELSE IF (IANS.EQ.2) THEN
  WRITE (6,*)'Enter total no. of records to be output : '
  READ (5,*) NOP
  DO 40 II = 1, NOP
    WRITE (6,*)'Enter l.d. of record #', II, ' : '
    READ (5,*) IDOP(II)
  CONTINUE
40 ELSE IF (IANS.EQ.3) THEN
  WRITE (6,*)'Enter name of new output file : '
  READ (5,50) OPNAME
  FORMAT (A17)
50 END IF
IF (IANS.EQ.1.OR.IANS.EQ.2) THEN
  WRITE (6,*)'Enter title (<60 characters) : '
  READ (5,60) TITLE
  WRITE (6,*)'Enter XREC in metres : '
  READ (5,*) XR
  FORMAT (A60)
  RNOP = REAL(NOP)
  RNCHR = REAL(NCHR)
  OPEN (10,FILE=IPDISC,STATUS='OLD',ACCESS='DIRECT',RECL=LEN)
  OPEN (18,FILE=OPNAME,STATUS='UNKNOWN',FORM='FORMATTED')
  WRITE (18,60) TITLE
  WRITE (18,*)'~~~~~'
1' WRITE (18,*)'No. of shots          No. of receivers
1'
  WRITE (18,70) RNOP, RNCHR
  FORMAT (F5.1, 30X, F5.1)
  WRITE (18,*)'XSHOT  ZSHOT  XREC  ZREC  TTIME'
  DO 100 II = 1, NOP
    IDCODE = (IDOP(II) - 1) * (NCHR + 1) + 1
    READ (10,REC=IDCODE) A, SORPOS, NRECS, RECDEP, RECDEP,
      RECDEP, NFIRST, A(1:12), B, DT
    DO 80 JJ = 1, NCHR
      TT = NFIRST(JJ) * DT / 1E6
      WRITE (18,90) XS, SORPOS, XR, RECDEP(JJ), TT,
        NFIRST(JJ)
    CONTINUE
80
90 FORMAT (F5.1, F9.1, F11.2, F8.1, F14.8, I12)
  WRITE (6,*)'Done for shot #', IDOP(II)
  CONTINUE
100 CLOSE (18)
CLOSE (10)
END IF
IF (IANS.NE.0) GO TO 10
RETURN
END

```

```

C Further subroutines for XHR programs
C
C Written by M J Findlay 1987-1989
C
C SUBROUTINE MENUD(NSAMR, NCHR, R4DAT, NTRACE, DT, NSHOT)
C
C Subroutine to plot FFT of individual traces in current array
C
  1  PARAMETER (NL=2048)
    DIMENSION R4DAT(NSAMR,NCHR)
    DIMENSION P(NL), Y(NL), S(NL), G(NL)
    CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECLOC,
    COMSHT, FILREC
  2  COMPLEX CSX(NL)
    CALL POWER2(NSAMR, NSAMP2)
    NTRACE = 12
  10  WRITE (6,*)
    WRITE (6,*) '          Trace Spectral Plot Menu          '
    WRITE (6,*) '          ~~~~~~          ~~~~~~          '
    WRITE (6,20) NTRACE
    20  FORMAT (/, 1 Trace number           :', I2)
    WRITE (6,30)
    30  FORMAT (/, 2 Amplitude and phase spectra ')
    WRITE (6,40)
    40  FORMAT (/, 3 Amplitude spectrum only  ')
    WRITE (6,50)
    50  FORMAT (/, 4 Power spectrum           ')
    WRITE (6,60)
    60  FORMAT (/, 5 FFT ALL traces OP to R4dat ')
    WRITE (6,70)
    70  FORMAT (/, 6 Put Autocorrelograms of traces into R4dat')
    WRITE (6,80)
    80  FORMAT (/, 7 Cross correlate to find optimum lag ')
    WRITE (6,90)
    90  FORMAT (/, 0 Return to MAIN MENU      ')
    READ (5,*) IOPT
    IF (IOPT.EQ.0) RETURN
    IF (IOPT.EQ.1) THEN
      WRITE (6,*) 'Enter trace number : '
      READ (5,*) NTRACE
    END IF
    IF (NTRACE.LT.1.OR.NTRACE.GT.NCHR) GO TO 10
    IF (IOPT.EQ.2.OR.IOPT.EQ.3) THEN
      DO 100 I = 1, NSAMR
        CSX(I) = CMPLX(R4DAT(I,NTRACE),0.0)
      100 CONTINUE
      DO 110 I = NSAMR + 1, NSAMP2
        CSX(I) = 0.0
      110 CONTINUE
      KEY = 0
      NPR = 0
      IRES1 = 0
      IRES2 = 0
      CALL TIME(KEY, NPR, IRES)
      CALL FORK(NSAMP2, CSX, -1.0)
      KEY = 15
      NPR = 0
      IRES1 = 0
      IRES2 = 0
      CALL TIME(KEY, NPR, IRES2)
      PRINT *, 'FFT TIME IN MICROSECS =', IRES1, IRES2
      IDSH = 473
      IDRH = 475
      TNAME = 'SWT014'
      TITLE = LOCATE // DATE
      CALL SPECPL(NSAMP2, CSX, DT, TITLE, TNAME, NSHOT, NTRACE, IDSH,
      IDRH)
  1  END IF
    IF (IOPT.EQ.4) THEN
      DO 120 I = 1, NSAMR
        G(I) = R4DAT(I,NTRACE)
      120 CONTINUE
      DO 130 I = NSAMR + 1, NSAMP2
        G(I) = 0.0
      130 CONTINUE
      CALL POWERS(G, P, NSAMP2, Y, S)
      FMIN = 0.0
      FMAX = 0.5E6 / DT
      CALL MINSN(NSAMP2/2 + 1, P, PMIN, II)
      CALL MAXSN(NSAMP2/2 + 1, P, PMAX, II)
      PRINT *, 'FMIN,FMAX,PMIN,PMAX', FMIN, FMAX, PMIN, PMAX
      CALL PAPER(1)
      CALL PSPACE(0.1, 0.9, 0.1, 0.9)
      CALL MAP(FMIN, FMAX, PMIN, PMAX)
      CALL SCALES
      CALL BORDER
      DF = 2.0 * FMAX / REAL(NSAMP2)
      DO 140 I = 1, NSAMP2
        Y(I) = REAL(I - 1) * DF
      140 CONTINUE
      CALL PTOJIN(Y, P, 1, NSAMP2/2 + 1, 1)
      CALL GREND
    END IF
    IF (IOPT.EQ.5) THEN
      DO 150 II = 1, NCHR
        CSX(II) = CMPLX(R4DAT(II,II),0.0)
      150 CONTINUE
      DO 160 II = NSAMR + 1, NSAMP2
        CSX(II) = 0.0
      160 CONTINUE
      CALL FORK(NSAMP2, CSX, -1.0)
      DO 170 II = 1, NSAMR
        R4DAT(II,II) = CABS(CSX(II))
      170 CONTINUE
      DO 180 II = NSAMR + 1, NSAMP2
        R4DAT(II,II) = 0.0
      180 CONTINUE
    END IF
    IF (IOPT.EQ.6) THEN
      WRITE (6,*) 'Enter no. of lags to be calculated : '
      READ (5,*) NLAGS
      NORM = 1
      DO 200 JJ = 1, NCHR
        CALL ACF(R4DAT(1,JJ), S, NSAMR, NLAGS, NORM, I2ZC)
        CALL ZERO(NSAMR, R4DAT(1,JJ))
        DO 190 II = 1, NLAGS
          R4DAT(II,JJ) = S(II)
        190 CONTINUE
      200 CONTINUE
    END IF
  
```

```

200 CONTINUE
END IF
IF (IOPT.EQ. 7) THEN
  WRITE (6,*)'ENTER NO. OF LAGS TO BE CALCULATED : '
  READ (5,*) LG
  WRITE (6,*)'ENTER TRACE NUMBERS TO COMPARE : '
  READ (5,*) ID1, ID2
  CALL CROSS(NSAMR, R4DAT(1,ID1), NSAMR, R4DAT(1,ID2), LG, G)
  CALL MAXSN(LG, G, GMAX, II)
  CALL CROSS(NSAMR, R4DAT(1,ID2), NSAMR, R4DAT(1,ID1), LG, G)
  CALL MAXSN(LG, G, GMAX1, I11)
  WRITE (6,*)'OPTIMUM LAG = ', GMAX, ' AT ', II - 1
  WRITE (6,*)'OPTIMUM LEAD = ', GMAX1, ' AT ', I11 - 1
  WRITE (6,*)'LAG implies should increase time on 2nd trace.'
  WRITE (7,*)'SHOT NO. : ', NSHOT
  WRITE (7,*)'OPTIMUM LAG = ', GMAX, ' AT ', II - 1
  WRITE (7,*)'OPTIMUM LEAD = ', GMAX1, ' AT ', I11 - 1
  WRITE (7,*)'LAG implies should increase time on 2nd trace.'
END IF
IF (IOPT.NE. 0) GO TO 10
RETURN
END

C
1 SUBROUTINE MENU(NSAMR, NCHR, R4DAT, DT, FLOWCT, FHCUT, FLOOFF,
  FHIUFF, DBLOW, DBHIGH, FILTAP, BLHPAS, LENFIL, NPROCS,
2 IDPROC)
C
C Subroutine to apply High pass, Low pass, and Band pass filters to
C data
C And Median filter
C
PARAMETER (NL1=2048,NL2=128,N1=2048,N2=651
  DIMENSION R4DAT(NSAMR,NCHR), IDPROC(5,NCHR)
  REAL S(NL1), Y(NL1), FL, FH, R(NL2), G(NL2), A(NL2), F(NL2),
  1 C(NL1)
  REAL DUM(NL1), DUM1(NL1)
  COMPLEX CX(NL1)
  CHARACTER FILTAP*9, BLHPAS*9, ANS*1
  CALL POWER2(NSAMR, NSAMP2)
  NRECS = NCHR
  FILTAP = 'Cosine'
  BLHPAS = 'Band pass'
  FNYQ = .5E6 / DT
  FLOOFF = 0.0
  FHIUFF = FNYQ
  DBLOW = 6.0
  DBHIGH = 6.0
  10 WRITE (6,*)
  WRITE (6,*)' Filter Menu
  WRITE (6,*)' ~~~~~
  WRITE (6,20) FLOWCT
20 FORMAT (/' 1 Low cut frequency (Hz) :', F6.1)
  WRITE (6,30) FHCUT
30 FORMAT (/' 2 High cut frequency (Hz) :', F6.1)
  WRITE (6,40) FLOOFF
40 FORMAT (/' 3 Cosine low reject freq (Hz) :', F6.1)
  WRITE (6,50) FHIUFF
50 FORMAT (/' 4 Cosine high reject freq (Hz):', F6.1)
  WRITE (6,60) DBLOW

60 FORMAT (/' 5 Low dB/Octave slope :', F6.1)
  WRITE (6,70) DBHIGH
70 FORMAT (/' 6 High dB/Octave slope :', F6.1)
  WRITE (6,80) FILTAP, BLHPAS
80 FORMAT (/' 7 Filter type :', 1X, A9, 1X, A9)
  WRITE (6,90)
90 FORMAT (/' 8 Apply BANDPASS filter
  WRITE (6,100)
100 FORMAT (/' 9 Apply MEDIAN filter
  WRITE (6,110)
110 FORMAT (/' 10 Apply SPLIKING decon
  WRITE (6,120)
120 FORMAT (/' 11 APPL NEWMAN FILTER
  WRITE (6,130)
130 FORMAT (/' 0 Return to MAIN MENU
  READ (5,*) IANSWR
  IF (IANSWR.EQ. 1) THEN
    WRITE (6,*)'Enter low cutoff frequency (Hz) : '
    READ (5,*) FLOWCT
  ELSE IF (IANSWR.EQ. 2) THEN
    WRITE (6,*)'Enter high cutoff frequency (Hz) : '
    READ (5,*) FHCUT
  ELSE IF (IANSWR.EQ. 3) THEN
    WRITE (6,*)'Enter low reject frequency (Hz) : '
    READ (5,*) FLOOFF
  ELSE IF (IANSWR.EQ. 4) THEN
    WRITE (6,*)'Enter high reject frequency (Hz) : '
    READ (5,*) FHIUFF
  ELSE IF (IANSWR.EQ. 5) THEN
    WRITE (6,*)'Enter low cutoff slope (dB/Octave) : '
    READ (5,*) DBLOW
  ELSE IF (IANSWR.EQ. 6) THEN
    WRITE (6,*)'Enter high cutoff slope (dB/Octave) : '
    READ (5,*) DBHIGH
  ELSE IF (IANSWR.EQ. 7) THEN
    WRITE (6,*)'Select taper function (Cosine or Db/octave) : '
    READ (5,140) ANS
    140 FORMAT (A1)
    IF (ANS.EQ. 'C'.OR. ANS.EQ. 'c') THEN
      FILTAP = 'Cosine'
    ELSE
      FILTAP = 'DB/Octave'
    END IF
    WRITE (6,*)'Select pass function (Low / High / Band) : '
    READ (5,140) ANS
    IF (ANS.EQ. 'L'.OR. ANS.EQ. '1') THEN
      FLOWCT = 0.0
    ELSE IF (ANS.EQ. 'H'.OR. ANS.EQ. 'h') THEN
      FHCUT = FNYQ
    ELSE IF (ANS.EQ. 'B'.OR. ANS.EQ. 'b') THEN
      BLHPAS = 'Low pass'
    ELSE IF (ANS.EQ. 'H'.OR. ANS.EQ. 'h') THEN
      BLHPAS = 'High pass'
    ELSE
      BLHPAS = 'Band pass'
    END IF
  ELSE IF (IANSWR.EQ. 8) THEN
    DO 180 J = 1, NCHR

```

```

150 DO 150 I = 1, NSAMR
150   DUM(I) = RADAT(I,J)
150   CONTINUE
150   DO 160 I = NSAMR + 1, NSAMP2
150     DUM(I) = 0.0
150     CONTINUE
150     WRITE (6,*) 'FILTERING CHANNEL #', J
150     IF (FILTRP.EQ.'Cosine') THEN
150       CALL BPSCON(DUM, NSAMP2, CX, FLOWCT, FHICUT, FLOOFF, FHIOFF,
150         FNYQ)
150     ELSE
150       CALL BPDSBS(DUM, NSAMP2, CX, FLOWCT, FHICUT, DBLOW, DBHIGH,
150         FNYQ)
150     END IF
150     DO 170 I = 1, NSAMR
150       RADAT(I,J) = DUM(I)
150       CONTINUE
150       CONTINUE
150       NPROCS = NPROCS + 1
150       WRITE (6,*) 'NPROCS = ', NPROCS
150       IF (FILTRP.EQ.'Cosine') THEN
150         IDPROC(1,NPROCS) = 4
150         IDPROC(2,NPROCS) = NINT(FLOOFF)
150         IDPROC(3,NPROCS) = NINT(FLOWCT)
150         IDPROC(4,NPROCS) = NINT(FHICUT)
150         IDPROC(5,NPROCS) = NINT(FHIOFF)
150       ELSE
150         IDPROC(1,NPROCS) = 11
150         IDPROC(2,NPROCS) = NINT(FLOWCT)
150         IDPROC(3,NPROCS) = NINT(DBLOW)
150         IDPROC(4,NPROCS) = NINT(FHICUT)
150         IDPROC(5,NPROCS) = NINT(DBHIGH)
150       END IF
150       ELSE IF (IANSWR.EQ. 9) THEN
150         WRITE (6,*) 'Enter length of median filter (traces) : '
150         READ (5,*) LENFIL
150         PRINT *, NFIRST
150         CALL MEDFIL(NSAMR, NCHR, RADAT, NFIRST, LENFIL)
150         NPROCS = NPROCS + 1
150         IDPROC(1,NPROCS) = 8
150         IDPROC(2,NPROCS) = LENFIL
150         PRINT *, NFIRST
150       ELSE IF (IANSWR.EQ. 10) THEN
150         WRITE (6,*) 'Apply triangular weighting to ACF (l=y 0=n) : '
150         READ (5,*) IFJ
150         WRITE (6,*) 'Enter %ge white noise to add : '
150         READ (5,*) RFC
150         WRITE (6,*) 'Enter filter length in samples : '
150         READ (5,*) LF
150         WRITE (6,*) 'Enter prediction distance in samples : '
150         READ (5,*) MM
150         WRITE (6,*) 'WRITE FILTER TO ARRAY RADAT l=y 0=APPLY FILTER ?'
150         READ (5,*) IRESP
150         DO 230 JF = 1, NCHR
150           DO 190 I = 1, NSAMR
150             DUM1(I) = RADAT(I,JF)
150             CONTINUE
150             DO 200 I = NSAMR + 1, NSAMP2
150               DUM1(I) = 0.0
150               CONTINUE
150               IF (MM.EQ. 0) THEN
150                 CALL ACF(DUM1, A, NSAMR, 100, 1, I22C)
150                 M = I22C
150               ELSE
150                 M = MM
150                 END IF
150                 IF (M.GT. LF) GO TO 230
150                 CALL PDSCON(NSAMP2, DUM1, M, LF, R, G, A, F, RFC, IFJ)
150                 IF (IRESP.EQ. 1) THEN
150                   CALL ZERO(NSAMR, RADAT(1,JF))
150                   DO 210 I = 1, LF
150                     RADAT(I,JF) = F(I)
150                   CONTINUE
150                 ELSE
150                   CALL ZERO(NSAMR, C)
150                   DO 220 I = 1, LF
150                     C(I) = F(I)
150                   CONTINUE
150                   CALL FRECON(NSAMP2, DUM1, DUM, C, Y, S)
150                 END IF
150                 END IF
150                 IF (IANSWR.EQ. 11) THEN
150                   DO 270 J = 1, NCHR
150                     NLC = 2 * NLI
150                     CALL ZERO(NLC, CX)
150                     DO 240 I = 1, NSAMR
150                       CX(I) = CMLPX(RADAT(I,J), 0.0)
150                     CONTINUE
150                     CALL FORK(NSAMR, CX, -1.0)
150                     CX(1) = 0.0
150                     CX(NSAMR/2 + 1) = 0.0
150                     DO 250 I = 2, NSAMR / 2
150                       CX(I) = CX(I) * SORT(REAL(I - 1))
150                       CX(I) = CX(I) * (1., 1.)
150                       CX(NSAMR + 2 - I) = CONJG(CX(I))
150                     CONTINUE
150                     CALL FORK(NSAMR, CX, +1.0)
150                     DO 260 I = 1, NSAMR
150                       RADAT(I,J) = REAL(CX(I))
150                     CONTINUE
150                     CONTINUE
150                     END IF
150                     IF (IANSWR.NE. 0) GO TO 10
150                     RETURN
150                   SUBROUTINE MENUF(NSAM, NCHR, RADAT, DZ, DT, NCR)
150                     SUBroutine to obtain F-K spectrum of recorded data
150                     PARAMETER (M1=513, N2=64, N3=5120, N4=1024, N5=65)
150                     DIMENSION RADAT(NSAM, NCHR), NCR(2)
150                     CHARACTER AMS*1, LOGYN*1, ANSMER*1, FNAME*10
150                     COMPLEX CP(M4, N2), CW(N2), CTEMP
150                     REAL KNYQ, FTLSPC(N1, N5)
150                     DOUBLE PRECISION AMEPC(N5, N1), TEMF, CHTS(8), DECHTS(8), XM
150                     LOGICAL SOLID
150                     CALL POWER2(NSAM, NSAMR)
150

```

```

      CHTS(1) = 20.0
      CHTS(2) = 50.0
      CHTS(3) = 100.0
      CHTS(4) = 150.0
      DBCHTS(1) = 0.01
      DBCHTS(2) = 3.0
      DBCHTS(3) = 6.0
      DBCHTS(4) = 20.0
      DBCHTS(5) = 40.0
      NCHTS = 4
      NCHTDB = 5
      NF = N1
      FNYQ = .5E6 / DT
      KNYQ = .5 / DZ
      PRINT *, 'KNYQ = ', KNYQ, ' FNYQ = ', FNYQ
      ISAMP = NINT(FNYQ*2)
10  WRITE (6,*)
      WRITE (6,*) ' F-K Spectra Menu
      WRITE (6,*) ' ~~~ ~~~~~ ~~~
      WRITE (6,*)
      WRITE (6,20)
20  FORMAT (/) 1 Calculate 2DFFT
      WRITE (6,30)
30  FORMAT (/) 2 Output complex array to file
      WRITE (6,40)
40  FORMAT (/) 3 Input complex array from file
      WRITE (6,50)
50  FORMAT (/) 4 Apply filters to array contents
      WRITE (6,60)
60  FORMAT (/) 5 Plot F-K amplitude spectrum
      WRITE (6,70) (CHTS(I), I=1, NCHTS)
70  FORMAT (/) 6 Contours for F-K spectrum : ', 8(F4.0, 1X)
      WRITE (6,80)
80  FORMAT (/) 7 dB plot of F-K amplitude spectrum
      WRITE (6,90) (DBCHTS(I), I=1, NCHTDB)
90  FORMAT (/) 8 dB contours for F-K spectrum : ', 8(F4.0, 1X)
      WRITE (6,100)
100 FORMAT (/) 9 Isometric projection plot of spectrum
      WRITE (6,110)
110 FORMAT (/) 0 Return to MAIN MENU
      READ (5,*) IANSWR
      IF (IANSWR.EQ. 0) RETURN
      IF (IANSWR.EQ. 1) THEN
        WRITE (6,*) 'To or From F-K space (T/F) : '
        READ (5,120) ANS
        FORMAT (A1)
120  NCEN = NSAMR / 2 + 1
        KNCN = N2 / 2 + 1
        IF (ANS.EQ. 'F'.OR. ANS.EQ. 'f') THEN
          SIGNA = +1.0
          SIGNB = +1.0
          DO 130 J = 1, N2
            CP(NCEN,J) = CONJG(CP(1,J))
          DO 130 I = 1, NSAMR / 2 - 1
            CP(NCEN + I, J) = CONJG(CP(NCEN - I, J))
130  CONTINUE
          SWAP AROUND TWO UPPER QUADRANTS
          DO 150 J = 1, N2 / 2 - 1
            DO 140 I = NCEN + 1, NSAMR
              CTEMP = CP(I, J + KNCN)
              CP(I, KNCN + J) = CP(I, KNCN - J)
              CP(I, KNCN - J) = CTEMP
140  CONTINUE
            ELSE
              WRITE (6,*) 'RESPONSE OF FILTER REQUIRED ?'
              READ (5,160) ANSWER
              FORMAT (A1)
              IF (ANSWER.EQ. 'Y'.OR. ANSWER.EQ. 'y') THEN
                NCP = NSAM * NCHR
                CALL ZERO(NCP, RADAT(1,1))
                RADAT(512,12) = 1.0
              END IF
              SIGNA = -1.0
              SIGNB = -1.0
              NZERO = N2 * N4 * 2
              CALL ZERO(NZERO, CP(1,1))
              DO 180 J = 1, NCHR
                DO 170 I = 1, NSAM
                  CP(I, J) = CMPLX(RADAT(I, J), 0.0)
170  CONTINUE
180  CONTINUE
190  FORMAT (4(I4, 1X, F7.3, 1X, F7.3))
          C
          C Remove common receiver channel
          IF (NCR(1).NE. 0) THEN
            PRINT *, 'Enter common receiver channel to be ignored : '
            READ (5,*) NCRIGN
            IF (NCRIGN.EQ. 0) GO TO 210
            DO 210 J = NCRIGN, NCHR
              DO 200 II = 1, NSAMR
                CP(II, J) = CP(II, J + 1)
200  CONTINUE
210  CONTINUE
            END IF
            END IF
            KEY = 0
            NPR = 0
            IRES1 = 0
            IRES2 = 0
            CALL TIME(KEY, NPR, IRES)
            CALL FT2D(N4, N2, CP, SIGNA, SIGNB, CW)
            KEY = 15
            NPR = 0
            IRES1 = 0
            IRES2 = 0
            CALL TIME(KEY, NPR, IRES2)
            PRINT *, 'FFT TIME IN MICROSECS =', IRES1, IRES2
            KEY = 1
            NPR = 0
            IRES1 = 0
            CALL TIME(KEY, NPR, IRES1)
            PRINT *, 'FFT TIME IN MILLISECS =', IRES1
            IF (ANS.EQ. 'F'.OR. ANS.EQ. 'f') THEN
              PRINT *, 'WRITING CP TO RADAT'
              DO 230 J = 1, NCHR
                DO 230 I = 1, NSAM

```

```

230      RADAT(I,J) = REAL(CP(I,J))
      CONTINUE
      END IF
      ELSE IF (IANSWR.EQ. 4) THEN
C PUT filter response into array FILSPC
      NF = NSAMR / 2 + 1
      NK = N2 + 1
      DK = 2 * KNYQ / N2
      DF = 2 * FNYQ / NSAMR
      PRINT *, 'NF,NK,DK,DF', NF, NK, DK, DF
      WRITE (6,*) 'PASS or REJECT filter (P/R)?'
      READ (5,160) ANS
      CALL PIE(NK, NF, FILSPC, DK, DF, ANS)
      IF (ANS.EQ. 'R') THEN
        DO 250 I = 1, N2
          DO 240 J = 1, NF
            FILSPC(J,I) = 1.0 - FILSPC(J,I)
          CONTINUE
        CONTINUE
      END IF
C APPLY FILTER FILSPC
      DO 270 I = 1, N2 / 2
        DO 260 J = 1, NF
          CP(J,I) = CP(J,I) * FILSPC(J,I + N2/2)
        CONTINUE
      CONTINUE
      DO 290 I = N2 / 2 + 1, N2
        DO 280 J = 1, NF
          CP(J,I) = CP(J,I) * FILSPC(J,I - N2/2)
        CONTINUE
      CONTINUE
      ELSE IF (IANSWR.EQ. 2) THEN
        DO 300 J = 1, N2
          DO 300 I = 1, NF
            AMPSPC(J,I) = DBLE(CABS(CP(I,J)))
          CONTINUE
        CONTINUE
      CONTINUE
C rearrange array so that K=0 axis lies at sample N2/2+1
C
      DO 310 I = 1, N2 / 2
        DO 310 J = 1, NF
          TEMP = AMPSPC(I,J)
          AMPSPC(I,J) = AMPSPC(I + N2/2,J)
          AMPSPC(I + N2/2,J) = TEMP
        CONTINUE
      CONTINUE
      WRITE (6,*) 'Enter name of file to write to : '
      READ (5,320) FNAME
      FORMAT (A10)
      OPEN (4,FILE=FNAME,STATUS='NEW',FORM='UNFORMATTED')
      WRITE (4) ((REAL(AMPSPC(J,I)),J=1,N2),I=1,NF)
      CLOSE (4)
      ELSE IF (IANSWR.EQ. 5 .OR. IANSWR.EQ. 7 .OR. IANSWR.EQ. 9)
1      THEN
        PRINT *, 'SELECT MAX FREQ FOR PLOT '
        READ (5,*) FMAX
C
C write amplitude spectrum to correctly orientated array AMPSPC
C
      DO 340 J = 1, N2
        DO 330 I = 1, NF
          AMPSPC(J,I) = DBLE(CABS(CP(I,J)))
        CONTINUE
      CONTINUE
      ISOLOG = 0
      IF (IANSWR.EQ. 9) THEN
        WRITE (6,*) 'Log plot required (Y/N) ?'
        READ (5,120) ANS
        IF (ANS.EQ. 'Y' .OR. ANS.EQ. 'y') ISOLOG = 1
      END IF
      IF (IANSWR.EQ. 7 .OR. ISOLOG.EQ. 1) THEN
        XM = 0.000
        DO 360 J = 1, NF
          DO 350 I = 1, N2
            IF (AMPSPC(I,J).GT. XM) XM = AMPSPC(I,J)
          CONTINUE
        CONTINUE
        PRINT *, 'MAX AMPLITUDE = ', XM
        DO 380 I = 1, NF
          DO 370 J = 1, N2
            IF (AMPSPC(J,I).LE. 1.0D-8) THEN
              AMPSPC(J,I) = 1000.0
            ELSE
              AMPSPC(J,I) = -20 * LOG10(AMPSPC(J,I)/XM)
            END IF
          CONTINUE
        CONTINUE
      END IF
C Now rearrange K space so that 0.0,0.0 is in centre of K axis
C
      NF = NSAMR / 2 + 1
      NK = N2 + 1
      DO 400 I = 1, N2 / 2
        DO 390 J = 1, NF
          TEMP = AMPSPC(I,J)
          AMPSPC(I,J) = AMPSPC(I + N2/2,J)
          AMPSPC(I + N2/2,J) = TEMP
        CONTINUE
      CONTINUE
      DO 410 J = 1, NF
        AMPSPC(NK,J) = AMPSPC(1,J)
      CONTINUE
      C Now call plotting routine
      C
      LOGYN = 'N'
      IF (IANSWR.EQ. 7) THEN
        LOGYN = 'Y'
      CALL FKPLLOT(NK, NF, AMPSPC, DT, DZ, FMAX, NCHTDB, DBCHTS,
1      LOGYN)
      ELSE IF (IANSWR.EQ. 5) THEN
        LOGYN = 'N'
      CALL FKPLLOT(NK, NF, AMPSPC, DT, DZ, FMAX, NCHTS, CHTS, LOGYN)
      ELSE
        CALL PAPER(1)
        CALL J06WAF
C

```

```

CALL J06WBF(0.0, 1.0, 0.0, 1.0, 1)
CALL J06WCF(0.0, 1.0, 0.0, 1.0)
CALL J06AHF('FK Amplitude Spectrum ')
IROT = 0
IFAIL = 0
HF = 1.000
SOLID = .TRUE.
IBASE = +1
ZBASE = 0.000
JSECT = 0
ZMIN = 0.0
ZMAX = 0.0
NFMAX = NINT(FMAX*2.0E-6*DT*NF)
PRINT *, 'NFMAX =', NFMAX
NRMS = 5 * NFMAX
CALL J06HBF(AMPSPC, NK, NFMAX, NK, HF, IROT, SOLID, IBASE,
1      ZBASE, JSECT, ZMIN, ZMAX, RMS, NRMS, IFAIL)
END IF
ELSE IF (IANSWR.EQ. 6) THEN
PRINT *, ' No. of contours required (<=8) : '
READ (5,*) NCHTS
DO 420 J = 1, NCHTS
WRITE (6,*) ' Contour level ', J, ' : '
READ (5,*) CHTS(J)
420 CONTINUE
ELSE IF (IANSWR.EQ. 8) THEN
PRINT *, ' No. of contours required (<=8) : '
READ (5,*) NCHTDB
DO 430 J = 1, NCHTDB
WRITE (6,*) ' Contour level ', J, ' : '
READ (5,*) DBCHTS(J)
430 CONTINUE
END IF
IF (IANSWR.NE. 0) GO TO 10
RETURN
END

C
C SUBROUTINE WSHAP(NSAMR, NCHR, RADAT, NFIRST, NCR, DT)
C
C Routine to apply waveshaping filters to dataset.
C Version 1.0; Version 2.0 is MUCH better (in XHR,2_U)
C
PARAMETER (NS1=1024, NS2=2048, NS3=512)
DIMENSION RADAT(NSAMR, NCHR), NFIRST(NCHR), NCR(2)
REAL S1(NS1), S2(NS1), S3(NS1), A(NS1,3), B(NS1), RADAT(NS1)
REAL C(NS2), ER(NS3), REF(NS1), XPOSNS(NS3), DSERIS(NS1)
REAL BSERIS(NS1)
COMPLEX CMPB(NS1)
INTEGER NOPF(6)
CHARACTER WEIGHT*3, ANS*1, REPLY*3
LOGICAL FILTER
CALL POWERS(NSAMR, NSAMP2)

C
C DEFAULTS
C
FILTER = .FALSE.
NFILOTS = 1
LA = 64

NL = 5
LAG1 = -2
RFC = 0.01
WEIGHT = 'NO'
IFJ = 0
IDD = NCR(1)
IDB = NCR(2)
REPLY = 'YES'

C
10 WRITE (6,*)
WRITE (6,*) ' WAVESHAPING FILTER MENU (0 to EXIT) '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,*) '
WRITE (6,20) IDD, IDB
20 FORMAT (// 1 Nos. of channels for shaping (REF 1st) :', I2, IX,
1      I2)
WRITE (6,30) NFILOTS
30 FORMAT (// 2 No. of filters to be calculated (1-3) :', I2)
WRITE (6,40) LA
40 FORMAT (// 3 Length of first filter to be calculated :', I4)
WRITE (6,50) NL
50 FORMAT (// 4 No. of lags to be considered :', I3)
WRITE (6,60) LAG1
60 FORMAT (// 5 First lag for shaping :', I3)
WRITE (6,70) RFC
70 FORMAT (// 6 Amount of white noise to be added to ACF :', F8.6)
WRITE (6,80) WEIGHT
80 FORMAT (// 7 Weighting for ACF ? :', IX, A3)
WRITE (6,90)
90 FORMAT (// 8 Calculate filter coefficients ')
WRITE (6,100) REPLY
100 FORMAT (// 9 Shorten IP & REF wavelets :', A3)
IF (FILTER) THEN
WRITE (6,110)
110 FORMAT (// 10 Apply filter to alternate traces ')
END IF
READ (5,*) IANS
IF (IANS.EQ. 1) THEN
WRITE (6,*) 'Nos. of channels for shaping (Reference first) : '
READ (5,*) IDD, IDB
ELSE IF (IANS.EQ. 2) THEN
WRITE (6,*) 'No. of filters to be calculated : '
READ (5,*) NFILOTS
ELSE IF (IANS.EQ. 3) THEN
WRITE (6,*) 'Length of first filter required : '
READ (5,*) LA
ELSE IF (IANS.EQ. 4) THEN
WRITE (6,*) 'No. of lags for each filter : '
WRITE (6,*) 'No. of lags for first filter : '
READ (5,*) NL
ELSE IF (IANS.EQ. 5) THEN
WRITE (6,*) 'First lag for waveshaping : '
READ (5,*) LAG1
ELSE IF (IANS.EQ. 6) THEN
WRITE (6,*) 'Fraction of white noise to add to zero lag ACF : '
READ (5,*) RFC
ELSE IF (IANS.EQ. 7) THEN
IF (WEIGHT.EQ. 'YES') THEN
WEIGHT = 'NO'
IFJ = 0
ELSE

```

```

      WEIGHT = 'YES'
      IF J = 1
      END IF
    ELSE IF (IANS.EQ. 10) THEN
      WRITE (6,*) 'Select required filter (1-3) : '
      READ (5,*) IDFILT
      IF (IDFILT.GT. NFILTS .OR. IDFILT.LE. 0) GO TO 10
      IODDEV = MOD(IDD,2) + 1
      DO 140 J = IODDEV, NRECS, 2
        DO 120 I = 1, NSAMR
          B(I) = A(I, IDFILT)
120        CONTINUE
          CALL ZERO(NSAMR, S1)
          CALL ZERO(NSAMR, S2)
          CALL ZERO(NSAMR, S3)
          CALL FRECON(NSAMR, R4DAT(1,J), S1, B, S2, S3)
          LAG = NOPR(IDFILT)
          IF (LAG.GT. 0) THEN
            NXST = 1
            NXST = 1024 - LAG
            ELSE IF (LAG.LT. 0) THEN
              NXST = 1
              NXST = 1024 - LAG
            ELSE
              NXSTEP = -1
            GO TO 140
          END IF
        C
        C Shift output trace by optimum lag
        C
        LAG = NOPR(IDFILT)
        DO 130 I = NXST, NXSTEP
          IF (LAG.GT. 0) THEN
            R4DAT(I,J) = R4DAT(I + LAG,J)
          ELSE
            R4DAT(I + LAG,J) = R4DAT(I,J)
          END IF
130        CONTINUE
          IF (LAG.GT. 0) CALL ZERO(LAG, R4DAT(NSAMR - LAG + 1,J))
          IF (LAG.LT. 0) CALL ZERO(LAG, R4DAT(1,J))
140        CONTINUE
          ELSE IF (IANS.EQ. 9) THEN
            IF (REPLY.EQ. 'YES') THEN
              REPLY = 'NO'
            ELSE
              REPLY = 'YES'
            END IF
          ELSE IF (IANS.EQ. 0) THEN
            RETURN
          END IF
          IF (IANS.NE. 8) THEN
            C
            C STARTS FILTER CALCULATION
            C
            FILTER = .FALSE.
            GO TO 10
          END IF
        C
        DO 150 I = 1, 512

```

```

      XPOSNS(I) = REAL(I)
150    CONTINUE
      CALL PAPER(1)
      LA = IA / 2
      DO 160 I = 1, NSAMR
        DSERIS(I) = R4DAT(1,IDD)
        BSERIS(I) = R4DAT(1,IDB)
160    CONTINUE
      IF (REPLY.EQ. 'YES') THEN
        IFIRST = MIN(NFIRST(IDB), NFIRST(IDD))
        PRINT *, ' INPUT LAST SAMPLE NO. FOR REFERENCE TRACE : '
        READ (5,*) NLAST1
        PRINT *, ' INPUT LAST SAMPLE NO. FOR INPUT TRACE : '
        READ (5,*) NLAST2
        PRINT *, ' TAPER OR MUTE (T/M) ? '
        READ (5,170) ANS
170      FORMAT (A1)
        DO 180 I = 1, NSAMR - IFIRST + 1
          DSERIS(I) = DSERIS(I + IFIRST - 1)
          BSERIS(I) = BSERIS(I + IFIRST - 1)
180      CONTINUE
          LASTD = NLAST1 - IFIRST + 1
          LASTB = NLAST2 - IFIRST + 1
          PRINT *, LASTD, LASTB
          IF (ANS.EQ. 'T' .OR. ANS.EQ. 't') THEN
            LEND = (LASTD) * 2
            LENB = (LASTB) * 2
            PRINT *, LEND, LENB
            CALL FEJER(DSERIS(LASTD), LEND)
            CALL FEJER(BSERIS(LASTB), LENB)
            NDZ = NSAMR - LEND
            PRINT *, NDZ
            CALL ZERO(NDZ, DSERIS(LEND + 1))
            NDZ = NSAMR - LENB
            PRINT *, NDZ
            CALL ZERO(NDZ, BSERIS(LENB + 1))
          ELSE
            NDZ = NSAMR - LASTD
            CALL ZERO(NDZ, DSERIS(LASTD + 1))
            NDZ = NSAMR - LASTB
            CALL ZERO(NDZ, BSERIS(LASTB + 1))
          END IF
        C
        DO 270 ICOUNT = 1, NFILTS
          LA = IA * 2
          DO 190 I = 1, NSAMR
            REF(I) = DSERIS(I)
190          CONTINUE
            DO 200 I = 1, NSAMR
              R4AT(I) = BSERIS(I)
200            CONTINUE
              LCD = NSAMR * 2 + IA - 2
              CALL ZERO(NSAMR, A(1, ICOUNT))
              LB = NSAMR
              LD = NSAMR
              LL = LD + NL / 2 + 1
              LC = LA + LB - 1
              CALL ZERO(2048, C)
              CALL ZERO(1024, S1)
              CALL ZERO(1024, S2)

```

```

CALL ZERO(1024, S3)
CALL ZERO(1024, A(1,ICOUNT))
PRINT *, 'LB,LD,LA,LC,LL,NL,LAG1,REC,IFJ', LB, LD, LA, LC, LL,
      NL, LAG1, REC, IFJ
CALL SHAPE(LB, RDAT, LD, REF, LA, A(1,ICOUNT), LC, C, LL, NL,
1     LAG1, IND, ER, S1, S2, S3, RFC, IFJ)
PRINT *, 'OPTIMUM LAG : ', IND
WRITE (6,*) (ER(I),I=1,NL)
NOPT(ICOUNT) = LAG1 + IND - 1
IF (NOPT(ICOUNT).GT. 0) THEN
  NXST = NOPT(ICOUNT) + 1
  NXST = 1
ELSE IF (NOPT(ICOUNT) .LT. 0) THEN
  NXST = 1 - NOPT(ICOUNT)
ELSE
  GO TO 210
END IF

C Shift output trace by optimum lag
C
210 CALL MOVE(1024, 1024, C, NXST, 1024, C, NXST)
C ERR = ER(IND)
C
C Plot Reference Traces and filtered output traces together
C
  XMIN = 0.01
  XMAX = 0.48
  YMIN = 0.95 - 0.3 * ICOUNT
  YMAX = YMIN + 0.25
C
C NORMALISE OUTPUT TRACE C(LC)
C
  FNORM = REF(1)
  DO 220 I = 2, NSAMR
    FNORM = AMAX1(REF(I),FNORM)
  CONTINUE
  DO 230 I = 1, NSAMR
    REF(I) = REF(I) / FNORM
    RDAT(I) = BSERIS(I) / FNORM
  CONTINUE
  ISIZE = MAX(LB,LD)
  IF (REPLY.EQ.'YES') ISIZE = MAX(LASTB,LASTD)
  IF (ISIZE.GT. 512) ISIZE = 512
  SIZE = REAL(ISIZE)
  CALL PSPACE(XMIN, XMAX, YMIN, YMAX)
  CALL BORDER
  CALL MAP(1.0, SIZE, -7.0, 1.0)
  CALL PTJOIN(XPOSNS, RDAT, 1, ISIZE, 1)
  CALL MAP(1.0, SIZE, -5.0, 3.0)
  CALL PTJOIN(XPOSNS, REF, 1, ISIZE, 1)
  CALL MAP(1.0, SIZE, -3.0, 5.0)
  CALL PTJOIN(XPOSNS, C, 1, ISIZE, 1)
  ERMAX = 0.0
  DO 240 I = 1, 512
    C(I) = REF(I) - C(I)
    ERMAX = AMAX1(ERMAX,ABS(C(I)))
  CONTINUE
240

ERRMAX = ERMAX * 100.0
CALL NORMAN(512, C)
CALL MAP(1., SIZE, -1.0, 7.0)
CALL PTJOIN(XPOSNS, C, 1, ISIZE, 1)
CALL CTMAG(10)
CALL PLOTS(SIZE/2, 0.5, 'MAX ERROR (% of peak ref):')
CALL TYPEF(ERMAX, 2)
Y0 = YMIN - 0.05
CALL PSPACE(XMIN, XMAX, Y0, YMIN)
CALL MAP(0.0, 1.0, 0.0, 1.0)
CALL PLOTS(0.01, 0.5, 'LENGTH = ')
CALL TYPEF(LA)
CALL PLOTS(0.26, 0.5, 'MEAN SQ. ERR = ')
CALL TYPEF(ERR, 3)
CALL PLOTS(0.75, 0.5, 'OPT LAG = ')
CALL TYPEF(NOPT(ICOUNT))

C
C PLOT FILTER AND ITS AMPLITUDE SPECTRUM
C
  RLA = REAL(LA)
  DX = 0.23
  XMIN = 0.52
  XMAX = XMIN + DX
  YMIN = 0.95 - 0.3 * (ICOUNT)
  YMAX = YMIN + 0.25
  CALL MAXSN(LA, A(1,ICOUNT), AMAX, II)
  CALL PSPACE(XMIN, XMAX, YMIN, YMAX)
  CALL BORDER
  CALL MAP(1., RLA, -AMAX, AMAX)
  CALL XAXIS
  CALL PTJOIN(XPOSNS, A(1,ICOUNT), 1, LA, 1)
  XMIN = XMAX + 0.02
  XMAX = XMIN + DX
  DO 250 I = 1, LA
    CMPB(I) = CMPLX(A(I,ICOUNT),0.0)
  CONTINUE
  CALL FORK(LA, CMPB, 1.0)
  DO 260 I = 1, LA
    B(I) = CABS(CMPB(I))
  CONTINUE
  CALL MAXSN(LA, B, BMAX, II)
  CALL PSPACE(XMIN, XMAX, YMIN, YMAX)
  RLAL = REAL(LA/2) + 1
  LAL = LA / 2 + 1
  FNYQ = .5E3 / DT
  CALL MAP(0.0, FNYQ, 0.0, BMAX)
  CALL AXES
  CALL MAP(1., RLAL, 0.0, BMAX)
  CALL BORDER
  CALL PTJOIN(XPOSNS, B, 1, LAL, 1)
270 CONTINUE
  CALL GREND
  FILTER = .TRUE.
  GO TO 10
END

C Subroutine to apply gain compensation across array of traces
C
SUBROUTINE GNCMP(NSAMR, NCHR, R4DAT, NFRST, DBGAIN, GCMSC,
1     RECDEP, NCR, IDPROC, NPROC)
  PARAMETER (NS1=1024, NS2=60)

```

[illegible]

```

IF (ANS.EQ.'R'.OR.ANS.EQ.'r') THEN
  DO 280 J = 1, NCHR
    DO 270 I = 1, NSAMR
      RADAT(I,J) = RADAT(I,J) * (I - 1) ** 2 / 10000
    CONTINUE
  CONTINUE
  WRITE (6,*) 'Taper (Y/N) '
  READ (5,490) ANS
  IF (ANS.EQ.'Y'.OR.ANS.EQ.'y') THEN
    WRITE (6,*) 'Enter length of taper in samples : '
    READ (5,*) LENTAP
    NTAP0 = NSAMR - LENTAP
    DO 300 J = 1, NCHR
      DO 290 I = NTAP0, NSAMR
        TAP = 1.0 - REAL(I - NTAP0) / REAL(LENTAP)
        IF (I.GT.NTAP0 + LENTAP) TAP = 0.0
        RADAT(I,J) = RADAT(I,J) * TAP
      CONTINUE
    CONTINUE
  END IF
ELSE IF (ANS.EQ.'D'.OR.ANS.EQ.'d') THEN
  WRITE (6,*) 'ENTER SPIKE (1), FLAT(2), T-RAMP(3), T1/2-RAMP(4) '
  WRITE (6,*) 'T-.5 RAMP(5) '
  READ (5,*) IOP
  IF (IOP.EQ.1) THEN
    DO 310 J = 1, NCHR
      CALL ZERO(NSAMR, RADAT(1,J))
      RADAT(NSAMR/2 + 1,J) = 1.0
    CONTINUE
  ELSE IF (IOP.EQ.2) THEN
    DO 330 J = 1, NCHR
      DO 320 I = 1, NSAMR
        RADAT(I,J) = 1.0
      CONTINUE
    CONTINUE
  ELSE IF (IOP.EQ.3) THEN
    DO 350 J = 1, NCHR
      DO 340 I = 1, NSAMR
        RADAT(I,J) = REAL(I - 1)
      CONTINUE
    CONTINUE
  ELSE IF (IOP.EQ.4) THEN
    DO 370 J = 1, NCHR
      DO 360 I = 1, NSAMR
        RADAT(I,J) = SORT(REAL(I - 1))
      CONTINUE
    CONTINUE
  ELSE IF (IOP.EQ.5) THEN
    DO 390 J = 1, NCHR
      DO 380 I = 1, NSAMR
        RADAT(I,J) = 1. / SORT(REAL(I))
      CONTINUE
    CONTINUE
  END IF
ELSE
  WRITE (6,*) 'Enter length of AGC window in samples : '
  READ (5,*) LENAGC
  IDPROC(2,NPROCS) = LENAGC
  DO 400 J = 1, NCHR
    CALL AGC(NSAMR, 1, RADAT(1,J), LENAGC)
  CONTINUE
CONTINUE
END IF

```

```

400 CONTINUE
END IF
ELSE IF (IOPT.EQ.5) THEN
  WRITE (6,*) 'Length of taper to be applied in traces' //
  ' (affects L-1 traces) : '
  READ (5,*) LTAPEX
  WRITE (6,*) 'Enter first and last traces with significant data : '
  READ (5,*) NTRACO, NTRACL
  PI = 3.1415926535
  WRITE (6,*) 'Apply / Remove Taper (A/R) : '
  READ (5,490) ANS
  DO 420 J = 1, LTAPEX
    THETA = PI * REAL(J) / REAL(LTAPEX)
    FAC = 0.5 * (1.0 - COS(THETA))
    IF (ANS.EQ.'R'.OR.ANS.EQ.'r') FAC = 1.0 / FAC
    NTR0 = NTRACO - 1 + J
    NTR1 = NTRACL + 1 - J
    IF (NTR0.LE.0 .OR. NTR1.GT.NCHR) GO TO 420
    DO 410 I = 1, NSAMR
      RADAT(I,NTR0) = RADAT(I,NTR0) * FAC
      RADAT(I,NTR1) = RADAT(I,NTR1) * FAC
    CONTINUE
  CONTINUE
ELSE IF (IOPT.EQ.6) THEN
  WRITE (6,*) 'Enter no. of channel to be removed : '
  READ (5,*) NCHNLO
  DO 440 J = NCHNLO, NCHR - 1
    NFIRST(J) = NFIRST(J + 1)
    RECDP(J) = RECDP(J + 1)
    DO 430 I = 1, NSAMR
      RADAT(I,J) = RADAT(I,J + 1)
    CONTINUE
  CONTINUE
  CALL ZERO(NSAMR, RADAT(1,NCHR))
  NFIRST(NCHR) = 0
  RECDP(NCHR) = RECDP(2) - RECDP(1) + RECDP(NCHR - 1)
  ELSE IF (IOPT.EQ.7) THEN
    WRITE (6,*) 'Remove Spike or D.C. (S/D) ? '
    READ (5,490) ANS
    IF (ANS.EQ.'S'.OR.ANS.EQ.'s'.OR.ANS.EQ.'M')
      THEN
      WRITE (6,*) 'ENTER TRACE NUMBER : '
      READ (5,*) NTRSPK
      WRITE (6,*) 'ENTER SAMPLE RANGE : '
      READ (5,*) NSMSPK, LSMSPK
      DO 450 I = NSMSPK, LSMSPK
        IF (ANS.EQ.'M') THEN
          ALPHA = 0.0
        ELSE
          ALPHA = .5 * (RADAT(I - 1, NTRSPK) + RADAT(I + 1, NTRSPK))
        END IF
        RADAT(I, NTRSPK) = ALPHA
      CONTINUE
    ELSE
      DO 480 J = 1, NCHR
        DC = 0.0
        DO 460 I = 1, NSAMR
          DC = DC + RADAT(I,J)
        CONTINUE
        DC = DC / REAL(NSAMR)

```

```

PRINT *, ' TRACE ', J, ' DC = ', DC
DO 470 I = 1, NSAMR
  RADAT(I,J) = RADAT(I,J) - DC
CONTINUE
470 CONTINUE
480 END IF
ELSE IF (IOPT.EQ.8) THEN
  WRITE (6,*) 'CALCULATE COMMON TRACE MISALIGNMENT (1=Y)?'
  READ (5,*) ICORR
  IF (ICORR.EQ.1) THEN
    WRITE (6,*) 'ENTER CHANNELS #1, #2 :'
    READ (5,*) I1, I2
    WRITE (6,*) 'ENTER MAX LAG :'
    READ (5,*) LG
    CALL CROSS(NSAMR, RADAT(1,I1), NSAMR, RADAT(1,I2), LG, TEMP)
    CALL MAXSN(LG, TEMP, GMAX, I1)
    CALL CROSS(NSAMR, RADAT(1,I2), NSAMR, RADAT(1,I1), LG, TEMP)
    CALL MAXSN(LG, TEMP, GMAX1, I11)
    WRITE (6,*) GMAX, I1, GMAX1, I11
    IF (GMAX1.GT. GMAX) THEN
      WRITE (6,*) 'TRACE 2 LEADS TRACE1 BY :', I11 - 1
    ELSE
      WRITE (6,*) 'TRACE 2 LAGS TRACE1 BY:', I1 - 1
    END IF
    WRITE (6,*) 'LAG implies INCREASE times on trace2.'
    WRITE (6,*) 'ENTER 0 TO EXIT 1 TO CONTINUE:'
    READ (5,*) IGO
    IF (IGO.NE.1) GO TO 570
  END IF
  WRITE (6,*) 'Shift single or range of traces (S/R) ?'
  READ (5,490) ANS
  FORMAT (A1)
  IF (ANS.EQ. 'S'.OR. ANS.EQ. 'S') THEN
    WRITE (6,*) 'Enter no. of trace :'
    READ (5,*) NTRA
    WRITE (6,*) 'POSITIVE shift reduces travel time!'
    WRITE (6,*) 'Enter shift (rotational) to be applied :'
    READ (5,*) NCHANG
    WRITE (6,*) 'ENTER FIRST SAMPLE TO SHIFT :'
    READ (5,*) ITOSH
    DO 500 I = ITOSH, NSAMR - NCHANG
      IF (I.GT. NSAMR) GO TO 500
      IF ((I + NCHANG).LT. ITOSH) THEN
        TEMP(I) = 0.0
        GO TO 500
      ELSE
        TEMP(I) = RADAT(I + NCHANG, NTRA)
      END IF
    CONTINUE
    DO 510 I = NSAMR - NCHANG + 1, NSAMR
      IF (I.GT. NSAMR) GO TO 510
      TEMP(I) = RADAT(I - NSAMR + NCHANG, NTRA)
    CONTINUE
    DO 520 I = 1, NSAMR
      RADAT(I, NTRA) = TEMP(I)
    CONTINUE
    WRITE (6,*) 'Taper the trace (Y/N) ?'
    READ (5,490) ANS
    IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'Y') THEN
      WRITE (6,*) 'Enter start of taper in samples :'

```

```

580 CONTINUE
590 NPROCS = NPROCS + 1
IDPROC(1,NPROCS) = 1
IDPROC(2,NPROCS) = 0
WRITE (6,*) 'Equate rms amps of common receiver channels ?'
READ (5,490) ANS
IF (ANS .EQ. 'Y'.OR. ANS .EQ. 'y') THEN
CALL RMERR(NSAMR, RADAT(1, NCR(1)), SCALE1, INDAMP)
CALL RMERR(NSAMR, RADAT(1, NCR(2)), SCALE2, INDAMP)
SCALE = SCALE1 / SCALE2
C N.B. CHANGED NEXT LINE
IF (NCR(1) .EQ. 12 .AND. NCR(2) .GE. 13) THEN
C COMMON SHOT TRACES ARE TOP 12 AND BOTTOM 12 .
DO 610 J = 13, 24
DO 600 I = 1, NSAMR
RADAT(I, J) = RADAT(I, J) * SCALE
CONTINUE
600 CONTINUE
610 ELSE
C ASSUME NCR(1) IS HIGHER COMMON TRACE
C AND COMMON SHOT TRACES ARE ALTERNATE
IDTRAC = 2 - MOD(NCR(2), 2)
PRINT *, 'IDTRAC =', IDTRAC, ' SHOULD BE 1'
DO 630 J = IDTRAC, NCHR, 2
DO 620 I = 1, NSAMR
RADAT(I, J) = RADAT(I, J) * SCALE
CONTINUE
620 CONTINUE
630 IDPROC(2, NPROCS) = 1
END IF
ELSE
DO 640 I = 1, NCHR
WRITE (6,*) 'Enter gain in db for channel #', I
READ (5,*) DBGAIN(I)
CONTINUE
640 CONTINUE
END IF
ELSE IF (IOPT .EQ. 10) THEN
WRITE (6,*) 'AUTOMATIC PICK : 1=y 0=n ?'
READ (5,*) IREPLY
IF (IREPLY .EQ. 1) THEN
WRITE (6,*) 'ENTER FIRST SAMPLE TO BEGIN ESTIMATE:'
READ (5,*) NO
WRITE (6,*) 'ENTER significant no. of S.D.s:'
READ (5,*) SSDS
DO 680 J = 1, NCHR
DO 670 I = 1, NSAMR
RMEAN = 0.0
DO 650 I1I = 1, I - 1
RMEAN = RMEAN + RADAT(I1I, J)
CONTINUE
RMEAN = RMEAN / REAL(I - 1)
RMERR = 0.0
DO 660 I1I = 1, I - 1
DIF = RADAT(I1I, J) - RMEAN
RMERR = RMERR + DIF ** 2
CONTINUE
RMERR = RMERR / REAL(I - 1)
RMERR = SORT(RMERR)
DIF0 = ABS(RADAT(I, J) - RADAT(I - 1, J))
C Criterion that 1st difference is greater than NSSDS Standard
C Deviations from mean value to that sample.
IF (DIF0 .GT. SSDS*RMERR) THEN
NFIRST(J) = I
GO TO 680
END IF
CONTINUE
670 CONTINUE
680 ELSE
DO 690 I = 1, NCHR
WRITE (6,*) 'Enter 1st break sample for channel #', I
READ (5,*) NFIRST(I)
CONTINUE
690 END IF
ELSE IF (IOPT .EQ. 11) THEN
WRITE (6,*) NFIRST
WRITE (6,*) 'APPLY SAME MUTE TO ALL TRACES (1=y, 0=n) : '
READ (5,*) IOANS
IF (IOANS .EQ. 1) THEN
WRITE (6,*) 'Enter length of cosine taper in samps : '
READ (5,*) NTAPO0
IF (NTAPO0 .EQ. 0) GO TO 730
WRITE (6,*) 'Enter start of cosine taper (0=1st break) : '
READ (5,*) NTAPO0
DO 730 J = 1, NCHR
NTAPO = NFIRST(J) + NTAPO0
NTAP1 = NTAPO0 + NTAPO
IF (NTAP1 - 1 .GT. NSAMR .OR. NTAPO .LT. 1) THEN
WRITE (6,*) 'ERROR NTAPO, NTAP1 = ', NTAPO, NTAP1
GO TO 700
END IF
DO 710 I = NTAPO + 1, NTAP1 - 1
FAC = REAL(I - NTAPO) / REAL(NTAP1 - NTAPO)
FAC = FAC * 3.1415926535
FAC = 0.5 * (1.0 - COS(FAC))
RADAT(I, J) = RADAT(I, J) * FAC
CONTINUE
DO 720 I = 1, NTAPO
RADAT(I, J) = 0.0
CONTINUE
720 CONTINUE
730 ELSE
WRITE (6,*) 'Enter length of cosine taper (0=NO mute) : '
READ (5,*) NTAPO0
IF (NTAP1 .EQ. 0) GO TO 780
WRITE (6,*) 'Enter start of cosine taper (0=1st break) : '
READ (5,*) NTAPO0
WRITE (6,*) 'ENTER FIRST AND LAST TRACES TO APPLY MUTE'
READ (5,*) NGOO, NGOI
NTAPO = NFIRST(J) + NTAPO0
NTAP1 = NTAPO0 + NTAPO0

```

```

IF (NTAP1 - 1.GT. NSAMR .OR. NTAPO .LT. 1) THEN
  WRITE (6,*) 'ERROR NTAPO,NTAP1 = ', NTAPO, NTAP1
  GO TO 780
END IF
DO 760 I = NTAPO + 1, NTAP1 - 1
  FAC = REAL(I - NTAPO) / REAL(NTAP1 - NTAPO)
  FAC = FAC * 3.14159265
  FAC = 0.5 * (1.0 - COS(FAC))
  R4DAT(I,J) = R4DAT(I,J) * FAC
CONTINUE
DO 770 I = 1, NTAPO
  R4DAT(I,J) = 0.0
CONTINUE
770 CONTINUE
780 END IF
ELSE IF (IOPT.EQ. 0) THEN
  RETURN
END IF
GO TO 10
END

C
SUBROUTINE RMSEPR(NSAMR, X, RMS)
  REAL X(NSAMR)
  RMS = 0.0
  DO 10 I = 1, NSAMR
    RMS = RMS + X(I) ** 2
  10 CONTINUE
  RMS = RMS / NSAMR
  RMS = SORT(RMS)
  RETURN
END

C
SUBROUTINE MAXAMP(NSAMR, X, AMPMAX, INDAMP)
  REAL X(NSAMR)
  AMPMAX = 0.0
  DO 10 I = 1, NSAMR
    IF (ABS(X(I)) .GT. AMPMAX) THEN
      AMPMAX = ABS(X(I))
      INDAMP = I
    END IF
  10 CONTINUE
  RETURN
END

C
SUBROUTINE FT2D(N1, N2, CP, SIGNA, SIGNB, CW)
  INTEGER N1, N2
  COMPLEX CP(N1,N2), CW(N2)
  REAL SIGNA, SIGNB
  DO 10 I = 1, N2
    CALL FORK(N1, CP(I,I), SIGNA)
  10 CONTINUE
  DO 40 J = 1, N1
    DO 20 K = 1, N2
      CW(K) = CP(J,K)
    20 CONTINUE
  40 CALL FORK(N2, CW, SIGNB)
  DO 30 L = 1, N2

```

```

      CP(J,L) = CW(L)
30 CONTINUE
40 CONTINUE
  RETURN
END

C
C. Routine to plot F-K amp. spectrum contained in the array ARR(NK,NF)
C   N.B. Sample order in K ----> 1 = -knyq
C                                     nk/2 + 1 = 0.0
C                                     nk          = +knyq
C   NF, NK no. of frequencies, wavenumbers for plot
C   DT, DZ temporal and spatial sampling intervals in microseconds and metres
C   NCHTS no. of contour levels
C
C
SUBROUTINE FKPILOT(NK, NF, ARR, DT, DZ, FMAX, NCHTS, CHTS, LOGYN)
  INTEGER NK, NF
  REAL DT, DZ, KNYQ, KMIN, KMAX, FMIN, FMAXD
  DOUBLE PRECISION ARR(NK,NF), CHTS(8)
  CHARACTER LOGYN*1
  LOGICAL UNUSED(65,1024)
  EXTERNAL J06GBY, J06GBW
  FNYQ = .5E6 / DT
  KNYQ = .5 / DZ
  FMIN = 0.0
  KMIN = - (KNYQ)
  KMAX = -KMIN
  NLF = NINT(FMAX*NF/FNYQ)
  ICH = 1
  PRINT *, 'NK, NF =', NK, NF
  PRINT *, 'KMIN, KMAX =', KMIN, KMAX

C
C START PLOTTING ROUTINES
C
CALL X04AAF(1, 10)
CALL J00AAF(IFACE)
PRINT *, IFACE
CALL PAPER(1)
CALL GPSTOP(30)
CALL PSPACE(0.05, 0.75, 0.05, 0.9)
CALL MAP(0.0, 1.0, 0.0, 1.0)
CALL J06XAF
CALL J06XEF(0.0, 1.0, 0.0, 1.0)
CALL J06XEF(2)
CALL J06XGF(1.0, 2.0)
IF (LOGYN.EQ. 'Y') THEN
  CALL J06AHF('F-K Amplitude Spectrum (dB)')
ELSE
  CALL J06AHF('F-K Amplitude Spectrum (linear)')
END IF
IFAIL = 0
MA = 1
NA = 1
MB = NK
I1AB = 2
I1IGH = 0
IGRID = 0
PRINT *, 'NK,NF,NCHTS,CHTS,IFAIL,ICH', NK, NF, NCHTS, CHTS, IFAIL,
  ICH
CALL PSPACE(0.05, 0.75, 0.05, 0.9)
CALL MAP(0.0, 1.0, 0.0, 1.0)

```

```

CALL JO6XBF(0.0, 1.0, 0.0, 1.0)
CALL JO6XBF(2)
CALL JO6XGF(1.0, 2.0)
CALL JO6GBF(ARR, NK, MA, MB, NA, NLF, NCHTS, CHTS, ICH, JO6GBY,
1  ILAB, IHIGH, JO6GBM, IGRID, UNUSED, IFAIL)
PRINT *, 'IFAIL = ', IFAIL
CALL PSPACE(0.05, 0.75, 0.05, 0.9)
PRINT *, 'KMIN, KMAX, FMIN, FMAX = ', KMIN, KMAX, FMIN, FMAX
CALL MAP(KMIN, KMAX, FMIN, FMAX)
CALL BORDER
CALL AXORIG(KMIN, FMIN)
CALL CTMAG(15)
CALL AXES
PRINT *, 'IFAIL = ', IFAIL

C
C DRAW CONTOUR KEY
CALL PSPACE(0.77, 0.99, 0.8, 0.9)
CALL MAP(0.0, 1.0, 0.0, 1.0)
CALL BORDER
CALL CTMAG(20)
CALL PCSCEN(0.5, 0.3, 'CONTOUR KEY')
IF (LOGN.EQ.'Y') CALL PCSCEN(0.5, 0.8, 'dB scale')
CALL PSPACE(0.77, 0.99, 0.5, 0.78)
CALL BORDER
ysize = REAL(NCHTS) + 0.1
CALL MAP(0.0, 1.0, ysize, 0.0)
CALL CTMAG(12)
DO 10 I = 1, NCHTS
XI = REAL(I)
HEIGHT = REAL(CHTS(I))
CALL PLOTNI(0.1, XI, I)
CALL PLOTNF(0.35, XI, HEIGHT, 1)
10 CONTINUE
CALL GREND
RETURN
END

C
C Apply age of length LEN to data in array X
SUBROUTINE AGC(NSAMR, NCHR, X, LEN)
REAL X(NSAMR, NCHR), XPRIME(1024)
DO 30 J = 1, NCHR
CALL ZERO(NSAMR, XPRIME)
DO 10 ICEN = 1, NSAMR
L0 = LEN
IF (ICEN.LE. LEN/2) L0 = 2 * ICEN - 1
IF (ICEN.GT. NSAMR - LEN/2) L0 = (NSAMR - ICEN + 1) * 2 - 1
CALL RMERP(L0, X(ICEN - L0/2, J), RMS)
IF (RMS.EQ. 0.0) GO TO 10
XPRIME(ICEN) = X(ICEN, J) / RMS
10 CONTINUE
DO 20 I = 1, NSAMR
X(I, J) = XPRIME(I)
20 CONTINUE
30 CONTINUE
RETURN
END

C
C . subroutine to calculate pie slice filter for application to FK
C spectrum
C
C REMEMBER TO TRANSFORM TO CORRECT QUADRANTS IN MAIN ROUTINE
C sample ordering in K space is 1 = -KNQ
C nk/2+1 = 0
C nk = +KNYQ
C Note: must wrap around sample #1 to sample #nk before calling routine
C must also have data arranged in appropriate sequence
C require an odd no. of samples e.g 65, 129, 257
C
C
C SUBROUTINE PIE(NK, NF, PIEFLT, DK, DF, PORR)
REAL PIEFLT(NF, NK), DIST, RHISLO, RHICTS, RLOSLO, RLOCTS, RM3,
1 RM6, KNYQ
DOUBLE PRECISION HISLOP, HICTSL, LOSLOP, LOCTS, M3, M6
INTEGER KNYQ, FNYQ, KO, KL, FO, FL
CHARACTER*1 ANS, PORR
PI = 3.1415926535
KNYQ = NK / 2 + 1
KNYQ = NK / 2 * DK
WRITE (6,*) 'ZERO QUADRANT (L/R/No) ? -- (L = -Kspace) '
READ (5,10) ANS
10 FORMAT (A1)
FO = 1
FL = NF
KO = 1
KL = NK
DO 20 K = 1, NK
CALL ZERO(NF, PIEFLT(1, K))
20 CONTINUE
IF (ANS.EQ.'L'.OR.ANS.EQ.'1') THEN
DO 30 K = 1, KNYQ - 1
CALL ZERO(NF, PIEFLT(1, K))
30 CONTINUE
KO = KNYQ
ELSE IF (ANS.EQ.'R'.OR.ANS.EQ.'r') THEN
DO 40 K = KNYQ + 1, NK
CALL ZERO(NF, PIEFLT(1, K))
40 CONTINUE
KL = KNYQ
END IF
WRITE (6,*) 'TAPER INWARDS OR OUTWARDS FROM SLOPE (I/O) ?'
READ (5,10) ANS
WRITE (6,*) 'INPUT HIGH CUTOFF SLOPE (m/sec) (-VE for L quad) : '
READ (5,*) HISLOP
IF (ANS.EQ.'O'.OR.ANS.EQ.'o') THEN
WRITE (6,*) 'INPUT HIGH CUTOFF TAPER SLOPE (m/sec) : '
READ (5,*) HICTSL
IF (HISLOP + HICTSL.EQ. 0.0) THEN
M3 = 0.0
ELSE
M3 = (HISLOP + HICTSL - 1) - SQRT(HISLOP**2 + HICTSL**2 + (
1 HISLOP + HICTSL)**2 + 1) / (HISLOP + HICTSL)
END IF
PRINT *, 'HISLOP, HICTSL, M3', HISLOP, HICTSL, M3
MULT1 = 1
MULT2 = 1
IF (HICTSL.LT. 0.0) MULT1 = -1
IF (HISLOP.LT. 0.0.AND. HICTSL.GE. HISLOP) MULT2 = -1
END IF

```

```

WRITE (6,*)'INPUT LOW CUTOFF SLOPE (m/sec) (-VE for L quad) : '
READ (5,*) LOSLOP
IF (ANS.EQ.'0'.OR.ANS.EQ.'o') THEN
  WRITE (6,*)'INPUT LOW CUTOFF TAPER SLOPE (m/sec) : '
  READ (5,*) LOCTSL
  IF (LOSLOP + LOCTSL.EQ.0.0) THEN
    M6 = 0.0
  ELSE
    M6 = ((LOSLOP*LOCTSL - 1) - SQRT(LOSLOP**2 + LOCTSL**2 +
      1 LOSLOP*LOCTSL**2 + 1)) / (LOSLOP + LOCTSL)
  END IF
  MULT3 = 1
  MULT4 = 1
  IF (LOCTSL.LT.0.0) MULT3 = -1
  IF (LOSLOP.LT.0.0.AND.LOCTSL.LE.LOSLOP) MULT4 = -1
  PRINT *, 'LOSLOP,LOCTSL,M6', LOSLOP, LOCTSL, M6
END IF
WRITE (6,*)'Enter no. of wraparounds required '
READ (5,*) NWRAPS
IF (ANS.EQ.'1'.OR.ANS.EQ.'i') THEN
  WRITE (6,*)'INPUT DISTANCE IN SAMPLES FOR TAPER : '
  READ (5,*) NSAMP
END IF
WRITE (6,*)'INPUT LENGTH OF TAPER FOR K-SPACE (samples) : '
READ (5,*) LNTAPK
WRITE (6,*)'INPUT LENGTH OF TAPER FOR F-SPACE (samples) : '
READ (5,*) LNTAPF
RHISLO = REAL(HISLOP)
RHICTS = REAL(HICTSL)
RLOSLO = REAL(LOSLOP)
RLOCTS = REAL(LOCTSL)
RM3 = REAL(M3)
RM6 = REAL(M6)
IF (ANS.EQ.'0'.OR.ANS.EQ.'o') THEN
  WRITE (6,*)'Enter frequency axis intercepts Hicut, Hizero, ' //
  'Locut, Lozero : '
  READ (5,*) RHSLIN, RHCTIN, RLSLIN, RLCTIN
  WRITE (6,*)'RHICTS,RHCTIN,MULT1', RHICTS, RHCTIN, MULT1
  WRITE (6,*)'RHISLO,RHSLIN,MULT2', RHISLO, RHSLIN, MULT2
  DO 70 J = K0, KL
    JIND = 0
    IF (NWRAPS.EQ.0.AND.J.EQ.33) JIND = 1
    PRINT *, 'J = ', J
    X = DK * (J - KNYQ)
    DO 60 I = 1, NF
      Y = DF * (I - 1)
      IF (DIST(X,Y,RHICTS,RHCTIN,MULT1).GT.0.0) THEN
        RESP = 0.0
      ELSE IF (DIST(X,Y,RHISLO,RHSLIN,MULT2).GT.0.0)
        THEN
        D1 = (Y - HISLOP*X - RLSLIN) / (HISLOP - M3)
        D2 = (Y - HICTSL*X - RHCTIN) / (HICTSL - M3)
        D2 = ABS(D2)
      ELSE IF (DIST(X,Y,RLOSLO,RLSLIN,1).GT.D1) THEN
        RESP = 1.0
      ELSE IF (DIST(X,Y,RLOSLO,RLSLIN,1).GT.0.0) THEN
        THETA = DIST(X,Y,RLOSLO,RLSLIN,1) * PI / D1
        RESP = 0.5 * (1 - COS(THETA))
      ELSE
        THETA = PI * D2 / (D1 + D2)
        RESP = 0.5 * (1 - COS(THETA))
    END IF
  END IF
  PRINT *, 'INTERCEPTS : ', RHCTIN, RHSLIN, RLCTIN,
    GO TO 50
END IF
ELSE
  HISLOP = HISLOP / (DF/DK)
  LOSLOP = LOSLOP / (DF/DK)
  MULT = 1
  IF (HISLOP.LT.0.0.AND.LOSLOP.GT.0.0) MULT = -1
  DO = -D0
  PRINT *, 'HISLOP,LOSLOP,D0,D1', HISLOP, LOSLOP, D0, D1
  DO 90 J = K0, KL
    X = REAL(J - KNYQ)
    DO 80 I = 1, NF
      Y = REAL(I - 1)
      IF (DIST(X,Y,RHISLO,RHSLIN,MULT).GT.0.0) THEN
        RESP = 0.0
      ELSE IF (DIST(X,Y,RHISLO,RHSLIN,MULT).GT.D0.AND.DIST(X,
        Y,RLOSLO,RLSLIN,1).GT.0.0) THEN
        THETA = DIST(X,Y,RHISLO,RHSLIN,MULT) * PI / D0
        RESP = 0.5 * (1 - COS(THETA))
      IF (DIST(X,Y,RLOSLO,RLSLIN,1).LE.D1) THEN
        THETA = DIST(X,Y,RLOSLO,RLSLIN,1) * PI / D1
        RESP = RESP * 0.5 * (1 - COS(THETA))
      END IF
      ELSE IF (DIST(X,Y,RLOSLO,RLSLIN,1).GT.D1) THEN
        RESP = 1.0
      ELSE IF (DIST(X,Y,RLOSLO,RLSLIN,1).GT.0.0) THEN
        THETA = DIST(X,Y,RLOSLO,RLSLIN,1) * PI / D1
        RESP = 0.5 * (1 - COS(THETA))
      ELSE
        RESP = 0.0
    END IF
  END IF

```

```

      IF (PIEFLT(I,J) .GT. 0.0) THEN
        PIEFLT(I,J) = PIEFLT(I,J) * RESP
      ELSE
        PIEFLT(I,J) = RESP
      END IF
    80 CONTINUE
  80 CONTINUE
  END IF
  IF (LNTAPK .GT. 0) THEN
    DO 110 J = KL - LNTAPK, KL
      THETA = REAL(J - (KL - LNTAPK)) / LNTAPK * PI
      THETA = PI - THETA
      LOWLT = KL - J + K0
      FAC = 0.5 * (1.0 - COS(THETA))
      DO 100 I = 1, NF
        PIEFLT(I,J) = PIEFLT(I,J) * (FAC)
        PIEFLT(I,LOWLT) = PIEFLT(I,LOWLT) * (FAC)
      100 CONTINUE
    110 CONTINUE
  END IF
  IF (LNTAPF .GT. 0) THEN
    DO 130 J = 1, NK
      DO 120 I = 1, LNTAPF
        THETA = REAL(I - 1) / LNTAPF * PI
        FAC = 0.5 * (1.0 - COS(THETA))
        PIEFLT(I,J) = PIEFLT(I,J) * FAC
      120 CONTINUE
    130 CONTINUE
  END IF
  C
  C NOW SET LOW NYQUIST (-KNYQ) TO HIGH NYQUIST (+KNYQ)
  C
  DO 140 I = 1, NF
    PIEFLT(I,1) = PIEFLT(I,NK)
  140 CONTINUE
  RETURN
END
C
FUNCTION DIST(X, Y, M, INTERC, MULT)
  REAL DIST, M, INTERC
  DIST = ((Y - M*X - INTERC)/SQRT(1 + M**2)) * MULT
END
C
C Subroutine to bandpass filter data using linear dB/octave taper
C
SUBROUTINE BPSDBS(X, NSAMR, CX, FLOWCT, FHICUT, DBLOW, DBHIGH,
  1 FNYQ)
  REAL X(NSAMR)
  REAL ALPHA, DBDBLOW, DDBHIGH, F, FLOW, FOHIGH
  COMPLEX CX(NSAMR)
  DBDBLOW = (ABS(DBLOW))
  DDBHIGH = (ABS(DBHIGH))
  FLOW = 0.0D0
  IF (DBDBLOW .GT. 1E-3) FLOW = FLOWCT * 2.0E0 ** (3.0E0/DBDBLOW)
  FOHIGH = FNYQ
  IF (DDBHIGH .GT. 1E-3) FOHIGH = FHICUT * 2.0E0 ** (-3.0E0/DDBHIGH)
  DO 10 I = 1, NSAMR
    CX(I) = CMPLX(X(I),0.0)
  10 CONTINUE
  CALL FORK(NSAMR, CX, -1.0)

```

```

      IL = NSAMR / 2 + 1
      DF = 2.0 * FNYQ / NSAMR
      DO 20 I = 1, IL
        F = ((I - 1)*DF)
        IF (F .LT. 1E-2) THEN
          ALPHA = 0.0
        ELSE IF (F .LT. FLOW) THEN
          ALPHA = DBDBLOW / 20.0E0 * (LOG10(F/FLOW)) / (LOG10(2.0E0))
        ELSE IF (F .LT. FOHIGH) THEN
          ALPHA = 1.0E0
        ELSE
          ALPHA = DDBHIGH / 20.0E0 * (LOG10(FOHIGH/F)) / (LOG10(2.0E0))
        ALPHA = 10.0D0 ** ALPHA
      END IF
      CX(I) = CX(I) * REAL(ALPHA)
    20 CONTINUE
    DO 30 I = IL + 1, NSAMR
      CX(I) = CONJG(CX(NSAMR - I + 2))
    30 CONTINUE
    CALL FORK(NSAMR, CX, +1.0)
    DO 40 I = 1, NSAMR
      X(I) = REAL(CX(I))
    40 CONTINUE
  RETURN
END
C
C SUBROUTINE TO BANDPASS FILTER DATA USING COSINE TAPER
C
SUBROUTINE BPSCOS(X, NSAMR, CX, FLOWCT, FHICUT, FLOWF, FHIOFF,
  1 FNYQ)
  REAL X(NSAMR)
  COMPLEX CX(NSAMR)
  PI = 3.1415926535
  SAMHZ = 0.5 * NSAMR / FNYQ
  LOCUT = INT(FLOWCT*SAMHZ) + 1
  LOOFF = INT(FLOWF*SAMHZ) + 1
  IHICUT = INT(FHICUT*SAMHZ) + 1
  IHIOFF = INT(FHIOFF*SAMHZ) + 1
  DO 10 I = 1, NSAMR
    CX(I) = CMPLX(X(I),0.0)
  10 CONTINUE
  CALL FORK(NSAMR, CX, -1.0)
  IL = NSAMR / 2 + 1
  DO 20 I = 1, IL
    IF (I .LE. LOOFF) THEN
      CX(I) = 0.0
    ELSE IF (I .LE. LOCUT) THEN
      CX(I) = CX(I) * 0.5 * (1.0 + COS((LOCUT - I)*PI/(LOCUT -
  1 LOOFF)))
    ELSE IF (I .LE. IHICUT) THEN
      CX(I) = CX(I)
    ELSE IF (I .LE. IHIOFF) THEN
      CX(I) = CX(I) * 0.5 * (1.0 + COS((I - IHICUT)*PI/(IHIOFF -
  1 IHICUT)))
    ELSE
      CX(I) = 0.0
    END IF
  20 CONTINUE
  DO 30 I = IL + 1, NSAMR

```

```
      CX(I) = CONJ(CX(NSAMR - I + 2))
30 CONTINUE
   CALL FORK(NSAMR, CX, +1.0)
   DO 40 I = 1, NSAMR
      X(I) = REAL(CX(I))
40 CONTINUE
   RETURN
END
```

```

C
C Additional subroutines for use with XHR programs
C Written by M J Findlay 1987-1989
C
C
C SUBROUTINE RDBFILE(NSAMR, NCHR, NRECS, RADAT, NFIRST, RECDP,
1 DBGAIN, GCMSC1, NFILES, NF1LE, NCR, NSHOT, DT, LEN,
2 SORPOS, NPROCS, IDPROC, IPDISC, IPSOUR, INPOPT, INPCHN,
3 A, PROC)
C
C requires SUBROUTINE RDNIM1 in NIMRWSUB for TAPES : reads in arrays to
C RADAT
C
C PARAMETER (NS1=1024, NS2=12)
C DIMENSION RADAT(NSAMR, NCHR), NFIRST(NCHR), RECDP(NCHR),
1 DBGAIN(NCHR)
C DIMENSION GCMSC1(NCHR), NFILE(2), NCR(2), IDPROC(5, NCHR),
1 TEMP(NS1)
C CHARACTER IPDISC*17, IPSOUR*4, A*180, INPOPT*3, B(9)*20
C CHARACTER*25 PROC(20)
C INTEGER*2 I2DAT(NS1, NS2)
C CHARACTER*25 PROC(20)
C IF (INPOPT.EQ.'NEW') GO TO 70
C IF (INPOPT.EQ.'SGY') GO TO 160
C
C Open disc file if needed
C
C IF (IPSOUR.EQ.'DISC') THEN
C ILEN = 25000
C OPEN (10, FILE=IPDISC, STATUS='OLD', ACCESS='DIRECT', RECL=ILEN)
C
C END IF
C K24 = 1024 * NCHR
C CALL ZERO(K24, RADAT(1,1))
C IF (NFILES.EQ.1) THEN
C PRINT *, NCHR, NSAMR, NFILE(1), INPCHN
C CALL RDNIM1(I2DAT, NCHR, NSAMR, NFILE(1), INPCHN)
C DO 20 J = 1, NCHR
C DO 10 I = 1, NSAMR
C IVAL = I2DAT(I,J)
C RADAT(I,J) = REAL(IVAL)
C CONTINUE
C CONTINUE
C ELSE
C CALL RDNIM1(I2DAT, NCHR, NSAMR, NFILE(1), INPCHN)
C DO 40 J = 1, NCHR
C K = 2 * J - 1
C DO 30 I = 1, NSAMR
C IVAL = I2DAT(I,J)
C RADAT(I,K) = REAL(IVAL)
C CONTINUE
C CONTINUE
C CALL REMIND(INPCHN)
C CALL RDNIM1(I2DAT, NCHR, NSAMR, NFILE(2), INPCHN)
C DO 60 J = 1, NCHR
C K = 2 * J
C DO 50 I = 1, NSAMR
C IVAL = I2DAT(I,J)
C RADAT(I,K) = REAL(IVAL)
C CONTINUE
C CONTINUE
C END IF

```

```

C IF (INPCHN.EQ.1) CALL REMIND(INPCHN)
C IF (INPCHN.EQ.10) CLOSE (INPCHN)
C RETURN
C
C 70 CONTINUE
C
C Open disc file if needed
C
C NSHOT = NO. OF SHOT (1.d.)
C
C PRINT *, LEN
C IDCODE = (NSHOT - 1) * (NCHR + 1) + 1
C PRINT *, IDCODE, NSHOT
C WRITE (6,*) Stack or New (1 or 0 -1 == REVERSE POLARITY) :
C READ (5,*) ISTACK
C IF (IPSOUR.EQ.'DISC') THEN
C OPEN (10, FILE=IPDISC, STATUS='OLD', ACCESS='DIRECT', RECL=LEN)
C
C END IF
C K24 = 1024 * NCHR
C CALL ZERO(K24, RADAT(1,1))
C IF (ISTACK.EQ.0) CALL ZERO(K24, RADAT(1,1))
C
C READ IN HEADER PARAMETERS
C
C DO 90 J = 1, NCHR
C DO 80 I = 1, 4
C IDPROC(I,J) = 0
C CONTINUE
C 90 CONTINUE
C NPROCS = 0
C PRINT *, 'READING HEADERS .....
C READ (10, REC=IDCODE) A, SORPOS, NRECS, RECDP, DBGAIN, GCMSC1,
1 NFIRST, NCR, NPROCS, IDPROC, DT, PROC
C
C READ IN SEISMOGRAM RECORDS
C
C ISTACK = 0
C IF (GCMSC1(1).EQ.0.0) GCMSC1(1) = 1.0
C DO 110 J = 1, NCHR
C NREC = IDCODE + J
C READ (10, REC=NREC) (TEMP(I), I=1, NSAMR)
C DO 100 I = 1, NSAMR
C IF (ISTACK.EQ.0) THEN
C RADAT(I,J) = RADAT(I,J) + TEMP(I)
C ELSE
C RADAT(I,J) = RADAT(I,J) - TEMP(I)
C END IF
C END IF
C CONTINUE
C CALL MAXSN(NSAMR, RADAT(1,J), XMAX, IMAX)
C CALL MINSN(NSAMR, RADAT(1,J), XMIN, IMIN)
C IF (ABS(XMIN).GT.XMAX) THEN
C XMAX = -XMIN
C IMAX = IMIN
C END IF
C 110 CONTINUE
C IF (INPCHN.EQ.1) CALL REMIND(INPCHN)
C IF (INPCHN.EQ.10) CLOSE (INPCHN)
C PRINT *, 'Reset parameters to defaults (Y/N)?'
C READ (5,120) ANS
C 120 FORMAT (A1)

```

```

IF (ANS.EQ.'Y'.OR.ANS.EQ.'y') THEN
  REMIND (0)
  READ (0,130) B, INPORT, IPDISC, OPDISC, OPFORM, IPSOUR
130  FORMAT (A20, 8(/A20)/A3/A16/A16/A6/A4)
  A = B(1) // B(2) // B(3) // B(4) // B(5) // B(6) // B(7) // B(
1    8) // B(9)
  WRITE (6,*) B
  WRITE (6,*) A
  NRECS = NCHR
  READ (0,140) NA, NA, NA, NA, NA, NA, NCR(1), NCR(2), SA,
1    TOPREC, RECSEP, SA
140  FORMAT (I4/I2/I1/I3/I3/I2/I2/I2/I2/F6.2/F6.2/F6.2/F6.2)
  RECDEP(1) = TOPREC
  DO 150 I = 2, NRECS
    IF (I.EQ.NCR(2)) THEN
      RECDEP(I) = RECDEP(I - 1)
    ELSE
      RECDEP(I) = RECDEP(I - 1) + RECSEP
    END IF
150  CONTINUE
  END IF
  C Check for saturation -- code removed
  C
  C GO TO 170
  C
  C READ IN SEG Y TAPE
  C
160  CONTINUE
  INPCHN = 1
  CALL SEGYPD(RADAT, NRECS, NSAMR, NSHOT, INPCHN)
170  RETURN
  END
  C
  C Subroutine to write out data to disc file
  C
  SUBROUTINE WRFLA(NSAMR, NCHR, RADAT, NFIRST, OPFORM, OPDISC,
1    NSHOT, LEN, A1, A2, A3, A4, A5, A6, A7, A8, A9, SORPOS,
2    NRECS, RECDEP, DEGAIN, GCMSC1, NCR, NPROCS, IDPROC, DT,
3    IOPCHN, PROC)
  DIMENSION RADAT(NSAMR, NCHR), NFIRST(NCHR), RECDEP(NCHR), A(9)
  CHARACTER OPDISC*17, OPFORM*6, A*20
  CHARACTER*20 A1, A2, A3, A4, A5, A6, A7, A8, A9
  CHARACTER*25 PROC(20)
  A(1) = A1
  A(2) = A2
  A(3) = A3
  A(4) = A4
  A(5) = A5
  A(6) = A6
  A(7) = A7
  A(8) = A8
  A(9) = A9
  NRECS = NCHR
  IF (OPFORM.EQ.'SEQUEN') THEN
    PRINT *, 'OPTION NOT AVAILABLE YET'
  ELSE IF (OPFORM.EQ.'DIRECT') THEN
    IDCODE = (NSHOT - 1) * (NRECS + 1) + 1
    OPEN (2,FILE=OPDISC,STATUS='UNKNOWN',FORM='UNFORMATTED',

```

```

1    ACCESS='DIRECT', RECL=LEN)
  C
  C OUTPUT PROCESSING AND DISPLAY PARAMETERS
  C
  WRITE (2,REC=IDCODE) A, SORPOS, NRECS, RECDEP, DEGAIN, GCMSC1,
1    NFIRST, NCR, NPROCS, IDPROC, DT, PROC
  C
  C WRITE SEISMOGRAM RECORDS
  C
  DO 10 J = 1, NRECS
    NREC = IDCODE + J
    WRITE (2,REC=NREC) (RADAT(I,J), I=1, NSAMR)
10  CONTINUE
  ELSE
    WRITE (6,*) 'SEG-Y OUTPUT TO TAPE ATTACHED TO UNIT 19 : '
    IOPCHN = 19
    CALL SEGYPW(RADAT, NRECS, NSAMR, NSHOT, IOPCHN)
  END IF
  CLOSE (2)
  RETURN
  END
  C
  C ARRAY plotting subroutine for seismic data
  C
  C Plenty of interactive options included to obtain required scaled data.
  C
  C Can apply AGC etc to plot without affecting original data
  C
  C Allows manual (interactive) first break picking/confirmation.
  C
  SUBROUTINE MENPL(NSAMR, NCHANS, RADAT, NFIRST, TITOFF, CHNSPL,
1    POLART, VARARE, NPLST, NPLST, IOPT, SORPOS, RECDEP,
2    NCR, DT, LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP,
3    RECILOC, COMSHT, FILREC, NSHOT, NPROCS, IDPROC)
  C
  C Subroutine for graphics output of array RADAT
  C
  PARAMETER (NS1=2048, NS2=60)
  CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP, RECILOC,
1    COMSHT
  CHARACTER*20 FILREC
  CHARACTER TITLE*20, STR1*3, FLAGBR*3, COMMENT*40, ANS*1
  CHARACTER*4 RES(5)
  CHARACTER VARARE*3, TITOFF*3, CHNSPL*26, POLART*7, NONE*4, STRING*
1    10
  REAL RECDEP(NCHANS)
  REAL A(NS1), B(NS1), RADAT(NS1, NS2), RADAT(NSAMR, NCHANS)
  INTEGER NFIRST(NCHANS), NCR(2), IDPROC(5, NCHANS)
  CALL TIME(6, 0, RES)
  TITLE = 'X-hole Reflection'
  FLAGBR = 'OFF'
  COMMENT = ' '
  C
  C SET UP DUMMY PLOT ARRAY RADAT(NSAMR, NCHANS)
  C
10  DO 20 J = 1, NCHANS
  DO 20 I = 1, NSAMR
    RADAT(I, J) = RADAT(I, J)
  20 CONTINUE
  C
  C MENU FOR PLOT OPTIONS
  C
30  WRITE (6,*)

```



```

CALL PCSCEN(0.4, 0.790, 'Travel Time (milliseconds)')
CALL PCSCEN(0.4, 0.025, 'Sample Number')
CALL CTRORI(90.0)
CALL PCSCEN(0.028, 0.4, 'Receiver Depth (metres)')
CALL CTRORI(0.0)
CALL PSPACE(0.0, 1.0, 0.07, 0.75)
CALL MAP(0.0, 1.0, RNCHR, 0.0)
RI = 0.035
DO 170 J = 1, NCHR2
  RJ = REAL(J)
  CALL POSITN(RI, RJ)
  CALL TYPENF(RECDP(J), 1)
CONTINUE
CALL PSPACE(0.07, 0.70, 0.07, 0.75)
CALL BORDER
CALL MAP(TMIN, TMAX, RNCHR, 0.0)
CALL AXORIG(TMAX, 0.0)
CALL ANNOTP(1, 1)
CALL XAXIS
CALL YAXISI(1.0)
CALL XGRAT
CALL MAP(SAME, SAML, RNCHR, 0.0)
CALL AXORIG(SAME, RNCHR)
CALL ANNOTP(0, 0)
CALL XAXIS
XDIV = (SAML - SAME) / 10
CALL SAMNOR(NSAMR, NCHR2, RDAT, NPLFST, NPLIST)
IF (IOPT.EQ.1) THEN
  N1 = 1
  N2 = NCHR2
  NSTEP = 1
ELSE IF (IOPT.EQ.2) THEN
  N1 = 1
  N2 = NCHR2 - 1
  NSTEP = 2
ELSE IF (IOPT.EQ.3) THEN
  N1 = 2
  N2 = NCHR2
  NSTEP = 2
ELSE IF (IOPT.EQ.4) THEN
  N1 = 1
  N2 = NCHR2 - 1
  NSTEP = 1
END IF
DO 210 J = N1, N2, NSTEP
  DEP = REAL(J)
  IF (IOPT.EQ.4.AND.J.GT.NCR(1)) THEN
    DEP = DEP - 1.0
  END IF
  IF (IOPT.EQ.4.AND.J.EQ.NCR(1)) GO TO 210
  A(I) = FLOAT(I)
  IF (POLART.EQ.'REVERSE') THEN
    B(I) = DEP - RDAT(I,J)
    ISIGN = -1
  ELSE
    B(I) = RDAT(I,J) + DEP
    ISIGN = +1
  END IF
CONTINUE
180

CALL PJOIN(A, B, NPLFST, NPLIST, 1)
IF (FLAGBR.EQ.'OFF') GO TO 190
CALL CTRMAG(20)
ARROWD = DEP - 0.3
DO 190 I = NPLFST, NPLIST
  IF (I.EQ.NFIRST(J)) CALL PLOTNC(A(I), ARROWD, 187)
CONTINUE
190
IF (VARE.EQ.'OFF') GO TO 200
DO 200 I = NPLFST, NPLIST
  IF (RDAT(I,J)*ISIGN.GT.-0.01) GO TO 200
  CALL POSITN(A(I), DEP)
  CALL JOIN(A(I), B(I))
CONTINUE
200
CALL CTRMAG(10)
210
CALL GREN
GO TO 10
END IF
IF (IANSWR.EQ.7) THEN
  IF (FLAGBR.EQ.'OFF') THEN
    FLAGBR = 'ON'
  ELSE
    FLAGBR = 'OFF'
  END IF
ELSE IF (IANSWR.EQ.8) THEN
  PRINT *, 'COMMENT : '
  READ (5,220) COMMENT
  FORMAT (A40)
220
END IF
IF (IANSWR.EQ.9) THEN
  WRITE (6,*) 'No. of channel to be printed : '
  READ (5,*) IDCHAN
  WRITE (6,*) 'Range of samples to be printed : '
  READ (5,*) NSAM0, NSAM1
  WRITE (6,230) ((I,RDAT(I,IDCHAN)),I=NSAM0,NSAM1)
  FORMAT (4(I4,1X,F12.1,2X))
230
  WRITE (6,*) 'Select first break sample no. : '
  READ (5,*) NFIRST(IDCHAN)
END IF
240 FORMAT (A1)
IF (IANSWR.EQ.10) THEN
  WRITE (6,*) 'Ramp or AGC : '
  READ (5,240) ANS
  IF (ANS.EQ.'R'.OR.ANS.EQ.'r') THEN
    WRITE (6,*) 'r-Squared or Exponential (S or E) ? '
    READ (5,240) ANS
    IF (ANS.EQ.'S'.OR.ANS.EQ.'s') THEN
      DO 260 J = 1, NCHANS
        DO 250 I = 1, NSAMR
          RDAT(I,J) = RDAT(I,J) * (I - 1) ** 2
        CONTINUE
      CONTINUE
250
260
    ELSE
      WRITE (6,*) 'Enter Alpha (Exponential constant) : '
      READ (5,*) ALPHA
      DO 280 J = 1, NCHANS
        DO 270 I = 1, NSAMR
          TI = EXP(ALPHA*TI)
          RDAT(I,J) = RDAT(I,J) * TI * TI
        CONTINUE
      CONTINUE
180

```

```

270          CONTINUE
280          CONTINUE
          ELSE
            WRITE (6,*) 'Length of AGC to be applied : '
            READ (5,*) LENAGC
            CALL AGC(NSAMR, NCHANs, RDAT, LENAGC)
          END IF
        END IF
        IF (IANSWR.NE. 0) GO TO 30
        RETURN
      END
    C Subroutine to median filter seismic data
    C Really only useful for zero-offset VSP
  C
    SUBROUTINE MEDFIL(NSAMR, NCHR, R4DAT, NFIRST, LENFIL)
      PARAMETER (NS1=60, NS2=1024, NS3=60)
      DIMENSION R4DAT(NSAMR, NCHR), NFIRST(NCHR)
      REAL X(NS1), Y(NS1), DGDAT(NS2, NS3), UGDAT(NS2, NS3)
      CHARACTER*1 ANS
      NRECS = NCHR
  C
  C Shift the traces : FIRST BREAKS AT NALIGN SAMPLES
  C
    10 FORMAT (A1)
      LENDU = NS2 * NS3
      CALL ZERO(LENDU, UGDAT(1,1))
      PRINT *, 'REMOVE COMMON TRACE (Y/N) ? '
      READ (5,10) ANS
      IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') THEN
        WRITE (6,*) 'Enter no. of trace to be removed : '
        READ (5,*) NDEAD
        DO 30 J = NDEAD, NRECS - 1
          NFIRST(J) = NFIRST(J + 1)
          DO 20 I = 1, NSAMR
            R4DAT(I,J) = R4DAT(I,J + 1)
          CONTINUE
        CONTINUE
      CONTINUE
      CALL ZERO(NSAMR, R4DAT(1,NRECS))
      NFIRST(NRECS) = 100
    END IF
    DO 50 J = 1, NRECS
      DO 40 I = 1, NSAMR
        DGDAT(I,J) = R4DAT(I,J)
      CONTINUE
    CONTINUE
    IF (NFIRST(1).EQ. 0) THEN
      PRINT *, 'INPUT FIRST BREAK SAMPLES : '
      READ (5,*) NFIRST
    END IF
    PRINT *, 'ALIGNING TRACES...'
    NALIGN = 50
    DO 80 J = 1, NRECS
      NDT = NFIRST(J) - NALIGN
      DO 60 I = 1, NSAMR - NDT
        DGDAT(I,J) = DGDAT(I + NDT,J)
      CONTINUE
      DO 70 I = NSAMR - NDT + 1, NSAMR
        DGDAT(I,J) = 0.0
        CONTINUE
      IN = NALIGN - 5
      C MUTE before first arrivals
      DO 90 J = 1, NRECS
        CALL ZERO(IN, DGDAT(1,J))
      CONTINUE
      PRINT *, 'TRACES ALIGNED ... PLOT (Y/N) ? '
      READ (5,10) ANS
      IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') THEN
        CALL MENPL1(NSAMR, NRECS, DGDAT, NFIRST)
      END IF
    C
    C Apply median filter across traces
    C
    100 PRINT *, 'APPLYING MEDIAN FILTER ...'
      DO 140 I = 1N + 1, NSAMR
        DO 110 J = 1 - LENFIL / 2, NRECS + LENFIL / 2
          IF (J.LT. 1.OR. J.GT. NRECS) THEN
            X(J + LENFIL/2) = 0.0
          ELSE
            X(J + LENFIL/2) = DGDAT(I,J)
          END IF
        CONTINUE
        DO 130 J = 1, NRECS
          DO 120 JL = 1, 48
            Y(JL) = X(JL)
          CONTINUE
          CALL MDIAN1(Y(J), LENFIL, DGDAT(I,J))
        CONTINUE
      CONTINUE
    140 CONTINUE
    C DISPLAY
    C
    PRINT *, 'TRACES STILL ALIGNED ... PLOT (Y/N) ? '
    READ (5,10) ANS
    IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') THEN
      CALL MENPL1(NSAMR, NRECS, DGDAT, NFIRST)
    END IF
    PRINT *, 'RE-APPLY MEDIAN FILTER (Y/N) ? '
    READ (5,10) ANS
    IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') GO TO 100
    WRITE(7,129) ((I, DGDAT(I,J), I=1, NSAMR), J=1, NRECS)
    C 129 FORMAT(5(I4,1X,F5.2,2X))
  C
  C Shift back the traces
  C
    PRINT *, 'SHIFTING BACK TRACES ...'
    DO 170 J = 1, NRECS
      WRITE(7,*) NFIRST
      NDT = NFIRST(J) - NALIGN
      WRITE(7,*) NDT
      DO 150 I = NSAMR, NALIGN, -1
        DGDAT(I,J) = DGDAT(I - NDT,J)
      CONTINUE
    CONTINUE
    150

```

```

DO 160 I = 1, NALIGN - 1
  DGDAT(I,J) = 0.0
160 CONTINUE
170 CONTINUE
C
  PRINT *, 'TRACES SHIFTED BACK ... PLOT (Y/N)?'
  READ (5,10) ANS
  IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') THEN
    CALL MENPL1(NSAMR, NRECS, DGDAT, NFIRST)
  END IF
  PRINT *, 'DISPLAY UPGOING WAVEFIELD (Y/N)?'
  READ (5,10) ANS
  IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'y') THEN
    DO 190 J = 1, NRECS
      DO 180 I = 1, NSAMR
        UGDAT(I,J) = RADAT(I,J) - DGDAT(I,J)
      CONTINUE
    CONTINUE
  END IF
  CALL MENPL1(NSAMR, NRECS, UGDAT, NFIRST)
180 CONTINUE
190 CONTINUE
END IF
PRINT *, 'COPY WAVEFIELD TO RADAT (Up/Down/No)?'
READ (5,10) ANS
IF (ANS.EQ. 'U'.OR. ANS.EQ. 'u') THEN
  DO 210 J = 1, NRECS
    DO 200 I = 1, NSAMR
      RADAT(I,J) = UGDAT(I,J)
    CONTINUE
  CONTINUE
ELSE IF (ANS.EQ. 'D'.OR. ANS.EQ. 'd') THEN
  DO 230 J = 1, NRECS
    DO 220 I = 1, NSAMR
      RADAT(I,J) = DGDAT(I,J)
    CONTINUE
  CONTINUE
END IF
RETURN
END
C
C. Subroutine to obtain median value of array X(N) (XMED)
C The array X is left sorted on exiting
C SOURCE : Numerical recipes
C
  SUBROUTINE MDIAN1(X, N, XMED)
    DIMENSION X(N)
    CALL PIKSR1(N, X)
    N2 = N / 2
    IF (2*N2.EQ. N) THEN
      XMED = 0.5 * (X(N2) + X(N2 + 1))
    ELSE
      XMED = X(N2 + 1)
    END IF
  END IF
  RETURN
END
C
  SUBROUTINE PIKSR1(N, ARR)
C SOURCE : Numerical recipes
  DIMENSION ARR(N)
  DO 30 J = 2, N
    A = ARR(J)
    DO 10 I = J - 1, 1, -1
      IF (ARR(I) .LE. A) GO TO 20
      ARR(I + 1) = ARR(I)
    CONTINUE
    I = 0
    20 ARR(I + 1) = A
    30 CONTINUE
    RETURN
  END
C
  SUBROUTINE WWSHAP(NSAMR, NCHR, RADAT, NFIRST, NCR, DT)
C
C Version 2.0 -- Menu-driven waveshaping code
C. Routine to apply waveshaping filters to dataset.
C Displays input,desired output, filter and filter output as time series
C and as amplitude spectra.
C
  PARAMETER (NS1=1024,NS2=2048,NS3=513)
  DIMENSION RADAT(NSAMR,NCHR), NFIRST(NCHR), NCR(2)
  REAL S1(NS1), S2(NS1), S3(NS1), A(NS1), B(NS1), RADAT(NS1)
  REAL C(NS2), ER(NS3), REF(NS1), XPOSNS(NS1), DSERIS(NS1)
  REAL BSERIS(NS1), S4(NS1)
  REAL TEMP(NS1), X(NS1), ACF(NS1), TEMP2(NS1), BUTT(NS1),
    1 BUTT2(NS1)
  REAL D(NS1), P1(NS3), P2(NS3), P3(NS3), P4(NS3)
  COMPLEX CMPEB(NS1), CBUTT(NS1), C1(NS1), C2(NS1), C3(NS1), C4(NS1)
  COMPLEX CXA(NS1), CXB(NS1), CXC(NS1), CXD(NS1), CTEMP(NS1)
  INTEGER NOPT(6), NDO(64)
  CHARACTER WEIGHT*3, ANS*1, REPLY*3
  LOGICAL FILTER
  CALL POWER2(NSAMR, NSAMP2)
  NSAMS = NSAMP2
  FNYQ = .566 / DT
  DF = FNYQ / REAL(NSAMS/2)
  DO 10 I = 1, NSAMS
    XPOSNS(I) = REAL(I)
  10 CONTINUE
C
C DEFAULTS
C
  FILTER = .FALSE.
  NFILTS = 1
  LA = 64
  NL = 5
  LAG1 = -2
  RFC = 0.01
  WEIGHT = 'NO'
  IFJ = 0
  IDD = NCR(1)
  IDB = NCR(2)
  REPLY = 'YES'
C
  20 WRITE (6,*)
  WRITE (6,*) '
  WRITE (6,*) '
  WRITE (6,*) '
  WRITE (6,*) '
  WRITE (6,30)
  WWSHAPING FILTER MENU (0 to EXIT)'
  ~~~~~

```

```

30 FORMAT (/' 1   Select "input" wavelet')
   WRITE (6,40)
40 FORMAT (/' 2   Select "desired output" wavelet')
   WRITE (6,50)
50 FORMAT (/' 3   Calculate filter ')
   WRITE (6,60)
60 FORMAT (/' 4   Apply filter')
   WRITE (6,70)
70 FORMAT (/' 0   Exit ')
   READ (5,*) IANS
   IF (IANS.EQ.1) THEN
80    WRITE (6,*)
       WRITE (6,*)' "Input" wavelet selection'
       WRITE (6,*)' ~~~~~~'
90    WRITE (6,90)
       FORMAT (/' 1   Window of single trace ')
100   WRITE (6,100)
       FORMAT (/' 2   Extract Minimum phase wavelet ')
110   WRITE (6,110)
       FORMAT (/' 3   Plot input wavelet ')
120   WRITE (6,120)
       FORMAT (/' 0   Exit ')
       READ (5,*) IOPT
       IF (IOPT.EQ.1) THEN
           WRITE (6,*)' Enter trace number : '
           READ (5,*) NTRACE
           WRITE (6,*)' ENTER window on trace in samples : '
           READ (5,*) NWT1, NWT2
           NWTDF = NWT2 - NWT1 + 1
           CALL ZERO(NSAMS, ACF)
           DO 130 I = 1, NWTDF
               ACF(I + NSAMS/2) = R4DAT(I + NWT1 - 1, NTRACE)
           CONTINUE
130      CALL NORMAN(NSAMS, ACF)
           LB = NSAMS / 2 + NWTDF
           ELSE IF (IOPT.EQ.2) THEN
               WRITE (6,*)'Enter l= same window 0=selection for channels : '
               READ (5,*) IREP
               IF (IREP.EQ.1) THEN
                   WRITE (6,*)'ENTER window on traces : '
                   READ (5,*) IW1, IW2
                   WRITE (6,*)'ENTER NUMBER OF CHANNELS : '
                   READ (5,*) NAV
                   WRITE (6,*)'ENTER FIRST CHANNEL : '
                   READ (5,*) NCHANN
                   END IF
                   CALL ZERO(NSAMS, TEMP2)
                   IF (IREP.NE.1) THEN
                       WRITE (6,*)' Average over N traces ENTER N : '
                       READ (5,*) NAV
                   END IF
                   DO 160 J = 1, NAV
                       CALL ZERO(NSAMS, TEMP)
                       IF (IREP.NE.1) THEN
                           WRITE (6,*)'Enter channel number (', J, ') : '
                           READ (5,*) NCHANN
                           WRITE (6,*)'Enter window on trace : '
                           READ (5,*) IW1, IW2
                       END IF
140                      DO 140 I = IW1, IW2
                           TEMP(I) = R4DAT(I, NCHANN)
                           CONTINUE
                           CALL CROSS(NSAMS, TEMP, NSAMS, TEMP, NSAMS, ACF)
                           IF (ACF(1).NE.0.0) THEN
                               DO 150 I = 1, NSAMS
                                   TEMP2(I) = TEMP2(I) + (ACF(I)/ACF(1))
                               CONTINUE
150                      END IF
                           IF (IREP.EQ.1) NCHANN = NCHANN + 1
                           CONTINUE
160                      C Do minimum phase conversion.
                           C
                           CALL MINPH(TEMP2, NSAMS)
                           WRITE (6,*)'Enter taper start and end for wavelet : '
                           READ (5,*) NWT1, NWT2
                           LB = NWT2
                           CALL LINTAP(NSAMS, TEMP2, 0, 0, NWT1, NWT2)
                           C Shift so origin is at sample NSAMS/2
                           DO 170 I = 1, NSAMS / 2
                               ACF(I + NSAMS/2) = TEMP2(I)
                               ACF(I) = TEMP2(NSAMS/2 + I)
                           CONTINUE
170                      CONTINUE
                           CALL NORMAN(NSAMS, ACF)
                           LB = LB + NSAMS / 2
                           ELSE IF (IOPT.EQ.3) THEN
                               CALL MAXSN(NSAMS, ACF, ACFMAX, II)
                               CALL MINSN(NSAMS, ACF, ACFMIN, II)
                               SCALE = ACFMAX
                               IF (ABS(ACFMIN).GT. ACFMAX) SCALE = ABS(ACFMIN)
                               SCALE = SCALE * 2
                               CALL PSPACE(0.05, 0.95, 0.05, 0.95)
                               RX = REAL(NSAMS)
                               RX0 = RX / 2 - 2.0 * REAL(NWT2) + 1
                               RX1 = RX0 + 4.0 * REAL(NWT2)
                               CALL MAP(RX0, RX1, -SCALE, SCALE)
                               CALL BORDER
                               CALL AXORIG(513.0, -2.0)
                               CALL XGRAT(20.0)
                               CALL CTRMAG(20)
                               CALL YAXISI(20.0)
                               CALL YAXISI(1.0)
                               CALL CTRMAG(10)
                               CALL PTOIN(XPOSNS, ACF, 1, NSAMS, 1)
                               CALL FRAME
                               END IF
                               IF (IOPT.NE.0) THEN
                                   GO TO 80
                               END IF
180                      ELSE IF (IANS.EQ.2) THEN
                           WRITE (6,*)'
                               WRITE (6,*)' Desired output wavelet '
                               WRITE (6,190)
                               ~~~~~~
190                      FORMAT (/' 1   Butterworth wavelet ')
                               WRITE (6,200)
                               FORMAT (/' 2   Trace in record ')
200                      WRITE (6,210)
                               FORMAT (/' 3   Plot output wavelet ')
210

```

```

220 WRITE (6,220)
    FORMAT (/,' 4 MINIMUM PHASE extracted wavelet ')
    WRITE (6,230)
230 FORMAT (/,' 0 EXIT')
    READ (5,*) IOPT
    IF (IOPT.EQ.1) THEN
        WRITE (6,*) 'Enter low-cut, slope, high-cut, slope '
        READ (5,*) BUT1, BUT2, BUT3, BUT4
    C HIGH CUT
        RNL = ALOG10((2.*10.** (BUT4/10.)) - 1.0)
        WRITE (6,*) 'RNL = ', RNL
        RNL = RNL / (2.*ALOG10(2.))
        WRITE (6,*) 'RNL = ', RNL
        DO 240 J = 1, NSAMS / 2 + 1
            RFR = DF * REAL(J - 1)
            TEM = 1. / (1. + ((RFR/BUT3)**(2.*RNL)))
            BUTT(J) = SORT(TEM)
        CONTINUE
240 C LOW CUT
        WRITE (6,*) 'NOW LOW CUT..'
        RNL = ALOG10((2.*10.** (BUT2/10.)) - 1.0)
        WRITE (6,*) 'RNL = ', RNL
        RNL = RNL / (2.*ALOG10(2.))
        WRITE (6,*) 'RNL = ', RNL
        BUTT2(1) = 0.0
        DO 250 J = 2, NSAMS / 2 + 1
            RFR = DF * REAL(J - 1)
            TEM = 1. / (1. + ((BUT1/RFR)**(2.*RNL)))
            BUTT2(J) = SORT(TEM)
        CONTINUE
250 DO 260 J = 1, NSAMS / 2 + 1
        BUTT(J) = BUTT(J) * BUTT2(J)
        CONTINUE
260 C TRANSFORM TO TIME DOMAIN
        ISAM = 2
        DO 270 I = NSAMS, NSAMS / 2 + 2, -1
            BUTT(I) = BUTT(ISAM)
            ISAM = ISAM + 1
        CONTINUE
270 DO 280 I = 1, NSAMS
        CBUTT(I) = CMPLX(BUTT(I),0.0)
        CONTINUE
280 CALL FORK(NSAMS, CBUTT, 1.)
    C Shift origin to NSAMS/2
        DO 290 I = 1, NSAMS / 2
            BUTT2(I + NSAMS/2) = REAL(CBUTT(I))
            BUTT2(I) = REAL(CBUTT(I + NSAMS/2))
        CONTINUE
290 C Normalise output wavelet
        CALL NORMAN(NSAMS, BUTT2)
        LD = NSAMS
        ELSE IF (IOPT.EQ.2) THEN
            WRITE (6,*) 'Enter trace i.d. number : '
            READ (5,*) NTRACE
            WRITE (6,*) 'Enter taper points (4) : '
            READ (5,*) NTAP0, NTAP1, NTAP2, NTAP3
            DO 300 I = 1, NSAMS
                BUTT2(I) = R4DAT(I,NTRACE)
            CONTINUE
            CALL LINTAP(NSAMS, BUTT2, NTAP0, NTAP1, NTAP2, NTAP3)
300
310 DO 310 I = NTAP0, NTAP3
        BUTT2(I - NTAP0 + NSAMS/2 + 1) = BUTT2(I)
        CONTINUE
310 DO 320 I = 1, NSAMS / 2
        BUTT2(I) = 0.0
        CONTINUE
320 DO 330 I = NTAP3 + 1 + NSAMS / 2, NSAMS
        BUTT2(I) = 0.0
        CONTINUE
330 CALL NORMAN(NSAMS, BUTT2)
        WRITE (6,*) 'NORMALISED OK'
        ID = NSAMS
        ELSE IF (IOPT.EQ.3) THEN
            CALL MAXSN(NSAMS, BUTT2, BMAX, IT)
            CALL MINSN(NSAMS, BUTT2, BMIN, IT)
            SCALE = BMAX
            IF (ABS(BMIN).GT. BMAX) SCALE = ABS(BMIN)
            SCALE = SCALE * 2
            CALL PSPACE(0.05, 0.95, 0.05, 0.95)
            RX = REAL(NSAMS)
            CALL MAP(RX0, RX1, -SCALE, SCALE)
            CALL BORDER
            CALL AXORIG(513.0, 0.0)
            CALL XGRAT(20.0)
            CALL PTJOIN(XPOSNS, BUTT2, 1, NSAMS, 1)
            CALL FRAME
            ELSE IF (IOPT.EQ.4) THEN
                WRITE (6,*) 'Enter 1=same window 0=selection for channels : '
                READ (5,*) IREP
                IF (IREP.EQ.1) THEN
                    WRITE (6,*) 'ENTER window on traces : '
                    READ (5,*) IW1, IW2
                    WRITE (6,*) 'ENTER NUMBER OF CHANNELS : '
                    READ (5,*) NAV
                    WRITE (6,*) 'ENTER FIRST CHANNEL : '
                    READ (5,*) NCHANN
                END IF
                CALL ZERO(NSAMS, TEMP2)
                IF (IREP.NE.1) THEN
                    WRITE (6,*) 'Enter channel number (', J, ') : '
                    READ (5,*) NCHANN
                END IF
                WRITE (6,*) 'Enter window on trace : '
                READ (5,*) IW1, IW2
            END IF
            DO 340 I = 1, NAV
                TEMP(I) = IW1, IW2
            CONTINUE
            CALL CROSS(NSAMS, TEMP, NSAMS, TEMP, NSAMS, BUTT2)
            IF (BUTT2(1).NE. 0.0) THEN
                DO 350 I = 1, NSAMS
                    TEMP2(I) = TEMP2(I) + (BUTT2(I)/BUTT2(1))
                CONTINUE
            END IF
350

```

```

360         IF (IREP .EQ. 1) NCHANN = NCHANN + 1
C         CONTINUE
C Do minimum phase conversion.
C
C         CALL MINPH(TEMP2, NSAMS)
C         WRITE (6,*) 'Enter taper start and end for wavelet : '
C         READ (5,*) NWT1, NWT2
C         LD = NWT2
C         CALL LINTAP(NSAMS, TEMP2, 0, 0, NWT1, NWT2)
C         Shift so origin is at sample NSAMS/2
C         DO 370 I = 1, NSAMS / 2
C             BUTT2(I + NSAMS/2) = TEMP2(I)
C             BUTT2(I) = TEMP2(NSAMS/2 + I)
C         CONTINUE
370         CONTINUE
C         NORMALISE INPUT WAVELET
C         CALL NORMAN(NSAMS, BUTT2)
C         LD = LD + NSAMS / 2
C
C         END IF
C         IF (IOPT .NE. 0) GO TO 180
C         ELSE IF (IANS .EQ. 3) THEN
C             WRITE (6,*)
C             WRITE (6,*) ' Calculating filter '
C             WRITE (6,*) ' ~~~~~~ '
C         END IF
C
C         Calculating the filter coefficients
C
C         WRITE (6,*) 'LD = ', LD
C         WRITE (6,*) 'LB = ', LB
C         LB = 1024
C         LD = 1024
C         WRITE (6,*) ' Enter filter length in samples : '
C         READ (5,*) LA
C         WRITE (6,*) ' Enter no. of lags to try : '
C         READ (5,*) NL
C         LL = LD + NL / 2 + 1
C         LL = 2048
C         WRITE (6,*) ' Enter 1st lag to try (-ve == OP leads IP) : '
C         READ (5,*) LAG1
C         RFC = 0.02
C         IFJ = 0
C         CALL ZERO(NS2, C)
C         CALL ZERO(NSAMS, S1)
C         CALL ZERO(NSAMS, S2)
C         CALL ZERO(NSAMS, S3)
C         CALL ZERO(NSAMS, A)
C         DO 380 I = 1, NSAMS
C             B(I) = ACF(I)
C
C INPUT
C         D(I) = BUTT2(I)
C
C OUTPUT
380         CONTINUE
C
C Get ACF of input trace ---> S1
C         CALL CROSS(LB, B, LB, B, LA, S1)
C         S1(I) = S1(I) * (1. + RFC)
C
C Loop over specified range of lags
C
C         DO 480 L = 1, NL

```

```

LAG = LAG1 + (L - 1)
C         WRITE (6,*) 'LAG NO. : ', LAG
C
C Move shifted desired output to C
C
C         CALL ZERO(2048, C)
C         DO 390 J = 1, LAG
C             C(J) = D(LD - LAG + J)
C         CONTINUE
390         DO 400 J = LAG + 1, LD
C             C(J) = D(J - LAG)
C         CONTINUE
400
C GET XCF OF B & Desired OP
C         CALL ZERO(NS1, S2)
C         CALL CROSS(LL, C, LB, B, LA, S2)
C Solve Normal equations
C         CALL EUREKA(LA, S1, S2, A, S3)
C
C TRANSFORM FILTER AND INPUT
C         WRITE (6,*) 'LA = ', LA
C         DO 410 I = LA + 1, NS1
C             A(I) = 0.0
C         CONTINUE
410         LA1 = LA * 7 / 8
C         LA2 = LA / 8
C         DO 420 I = 1, LA2
C             A(NS1 - I + 1) = A(LA - I + 1)
C             A(LA - I + 1) = 0.0
C         CONTINUE
420
C TAPER FILTER
C         DO 430 I = 1, NS1
C             CXA(I) = CMPLX(A(I), 0.0)
C             CXB(I) = CMPLX(B(I), 0.0)
C             CXD(I) = CMPLX(D(I), 0.0)
C             CXC(I) = CMPLX(C(I), 0.0)
C         CONTINUE
430         CALL FORK(NS1, CXA, -1.0)
C         CALL FORK(NS1, CXB, -1.0)
C         CALL FORK(NS1, CXC, -1.0)
C         CALL FORK(NS1, CXD, -1.0)
C         DO 440 I = 1, NS1 / 2 + 1
C             S4(I) = CABS(CXA(I))
C             S2(I) = CABS(CXB(I))
C         CONTINUE
440
C Test filter
C         DO 450 I = 1, NS1
C             CTEMP(I) = CXA(I) * CXB(I)
C             S4(I) = CABS(CTEMP(I))
C         CONTINUE
450         CALL FORK(NS1, CTEMP, +1.0)
C         DO 460 I = 1, NS1
C             TEMP(I) = REAL(CTEMP(I))
C         CONTINUE
460
C Calculate ERROR
C         ER(I) = 0.0
C         CALL NORMAN(NS1, C)
C         CALL NORMAN(NS1, TEMP)

```

```

DO 470 I = 1, NS1
  ER(I) = ER(I) + ((C(I) - TEMP(I))**2)
470 CONTINUE
  ER(I) = SORT(ER(I)/REAL(NS1))
480 CONTINUE
  WRITE (6,*)'MEAN SQUARE ENERGIES :', (ER(I),I=1,NS1)
C
C Find optimum lag
  ERMIN = 1.E10
  DO 490 I = 1, NL
    IF (ER(I) .LE. ERMIN) THEN
      ERMIN = ER(I)
      IND = I
    END IF
490 CONTINUE
  NLAGOP = IND + LAG1 - 1
  WRITE (6,*)'OPTIMUM LAG :', NLAGOP, (' ', IND, ' ')
  WRITE (6,*)'Minimum energy :', ERMIN
C
C Now repeat filter calculation for a length of 1024 with optimum lag
C
C Move shifted desired output to C
  LA = 1024
  LL = 2048
  CALL ZERO(2048, C)
  DO 500 J = 1, NLAGOP
    C(J) = D(LD - NLAGOP + J)
    CONTINUE
  DO 510 J = NLAGOP + 1, LD
    C(J) = D(J - NLAGOP)
  510 CONTINUE
C
C Get ACF of B
  CALL ZERO(NS1, S1)
  CALL CROSS(LB, B, LB, B, LA, S1)
  S1(1) = S1(1) * (1. + RFC)
C
C GET XCF OF B & Desired OP
  CALL ZERO(NS1, S2)
  CALL CROSS(LL, C, LB, B, LA, S2)
C
C Solve Normal equations
  CALL EUREKA(LA, S1, S2, A, S3)
C
C TRANSFORM FILTER AND INPUT
C
C TAPER FILTER
  L1 = NS1 / 4
  L2 = 3 * NS1 / 4
  R1 = REAL(L1)
  R2 = REAL(L2)
  DO 520 I = 1, NS1
    CXA(I) = CMPLX(A(I), 0.0)
    CXB(I) = CMPLX(B(I), 0.0)
    CXD(I) = CMPLX(D(I), 0.0)
    CXC(I) = CMPLX(C(I), 0.0)
  520 CONTINUE
  CALL FORK(NS1, CXA, -1.0)
  CALL FORK(NS1, CXB, -1.0)
  CALL FORK(NS1, CXC, -1.0)
  CALL FORK(NS1, CXD, -1.0)

```

```

C
C Test filter
  DO 530 I = 1, NS1
    CTEMP(I) = CXA(I) * CXB(I)
  530 CONTINUE
  CALL FORK(NS1, CTEMP, +1.0)
  DO 540 I = 1, NS1
    TEMP(I) = REAL(CTEMP(I))
  540 CONTINUE
C
C Calculate ERROR
  ERMIN = 0.0
  CALL NORMAN(NS1, C)
  CALL NORMAN(NS1, TEMP)
  DO 550 I = 1, NS1
    ERMIN = ERMIN + ((C(I) - TEMP(I))**2)
  550 CONTINUE
  ERMIN = SORT(ERMIN/REAL(NS1))
  560 CONTINUE
  WRITE (6,*)'MEAN SQUARE ENERGY:', ERMIN
  RX = REAL(NSAMS)
  LA1 = LA * 7 / 8
  LA2 = LA / 8
C
C Shift filter so that t=0 is at sample 513
  DO 570 I = 1, NS1
    XPOSNS(I) = REAL(I)
  570 CONTINUE
C
C Rotate filter trace A
  CALL ZERO(NS1, S4)
  DO 580 I = 1, NS1 / 2
    S4(I + NS1/2) = A(I)
    S4(I) = A(I + NS1/2)
  580 CONTINUE
  WRITE (6,*)'NSAMS =', NSAMS
  L1 = NS1 / 3
  L2 = 3 * NS1 / 4
  R1 = REAL(L1)
  R2 = REAL(L2)
  CALL PSPACE(0.05, 0.95, 0.0, 1.0)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL AXORIG(513.0, 0.0)
  CALL XGRAT(64.0)
  CALL PSPACE(0.05, 0.95, 0.8, 0.99)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL PTJOIN(XPOSNS, B, L1, L2, 1)
  CALL PSPACE(0.05, 0.95, 0.60, 0.8)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL PTJOIN(XPOSNS, D, L1, L2, 1)
  CALL PSPACE(0.05, 0.95, 0.4, .60)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL PTJOIN(XPOSNS, S4, L1, L2, 1)
  CALL PSPACE(0.05, 0.95, 0.2, 0.4)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL PTJOIN(XPOSNS, C, L1, L2, 1)
  CALL PSPACE(0.05, 0.95, 0.0, 0.2)
  CALL MAP(R1, R2, -1.0, 1.0)
  CALL PTJOIN(XPOSNS, TEMP, L1, L2, 1)
  CALL FRAME

```

C Now plot transforms

```

DO 590 I = 1, NSAMS
  C1(I) = CMPLX(B(I),0.0)
  C2(I) = CMPLX(D(I),0.0)
  C3(I) = CMPLX(A(I),0.0)
  C4(I) = CMPLX(TEMP(I))
CONTINUE
590  CALL FORK(NSAMS, C1, -1.0)
    CALL FORK(NSAMS, C2, -1.0)
    CALL FORK(NSAMS, C3, -1.0)
    CALL FORK(NSAMS, C4, -1.0)
    NSFT = NSAMS / 2 + 1
    DO 600 I = 1, NSFT
      P1(I) = CABS(C1(I))
      P2(I) = CABS(C2(I))
      P3(I) = CABS(C3(I))
      P4(I) = CABS(C4(I))
CONTINUE
600  CALL NORMAN(NSFT, P1)
    CALL NORMAN(NSFT, P2)
    CALL NORMAN(NSFT, P3)
    CALL NORMAN(NSFT, P4)
    R1 = 0.0
    R2 = REAL(NSFT) * DF
    L1 = 1
    L2 = NSFT
    CALL PSPACE(0.05, 0.95, 0.09, 0.99)
    CALL MAP(R1, R2, 0.0, 1.0)
    CALL AXORIG(0.0, 0.0)
    CALL XGRATI(100.0)
    CALL CTRMAG(20)
    CALL XAXISI(200.0)
    CALL CTRMAG(10)
    R1 = 0.0
    R2 = REAL(NSFT)
    L1 = REAL(R1)
    L2 = REAL(L2)
    CALL PSPACE(0.05, 0.95, 0.765, 0.99)
    CALL MAP(R1, R2, 0.0, 1.0)
    CALL PTJOIN(XPOSNS, P1, L1, L2, 1)
    CALL PSPACE(0.05, 0.95, 0.54, 0.765)
    CALL MAP(R1, R2, 0.0, 1.0)
    CALL PTJOIN(XPOSNS, P2, L1, L2, 1)
    CALL PSPACE(0.05, 0.95, 0.315, .54)
    CALL MAP(R1, R2, 0.0, 1.0)
    CALL PTJOIN(XPOSNS, P3, L1, L2, 1)
    CALL PSPACE(0.05, 0.95, 0.09, 0.315)
    CALL MAP(R1, R2, 0.0, 1.0)
    CALL PTJOIN(XPOSNS, P4, L1, L2, 1)
    CALL FRAME
ELSE IF (IANS.EQ. 4) THEN
610  WRITE(6,*)
    WRITE(6,*)'      FILTER APPLICATION'
    WRITE(6,*)'      ~~~~~'
    WRITE(6,*)
    WRITE(6,*)' 1 Select channels to apply filter '
    WRITE(6,*)
    WRITE(6,*)' 2 Apply filter to channels '
    WRITE(6,*)
    WRITE(6,*)' 0 EXIT '
C
C
SUBROUTINE LINTAP(NX, X, N1, N2, N3, N4)
  READ(5,*) IOPT
  IF (IOPT.EQ. 1) THEN
    WRITE(6,*)'Enter "1" if you want to apply to all : '
    READ(5,*) IREPL
    IF (IREPL.EQ. 1) THEN
      NAPPLY = NCHR
      DO 620 I = 1, NAPPLY
        NDO(I) = I
      CONTINUE
    ELSE
      WRITE(6,*)' Enter number of channels to apply filter
1: '
      READ(5,*) NAPPLY
      DO 630 I = 1, NAPPLY
        WRITE(6,*)'Enter channel #', I
        READ(5,*) NDO(I)
      CONTINUE
    END IF
  ELSE IF (IOPT.EQ. 2) THEN
    DO 700 I = 1, NAPPLY
      II = NDO(I)
      CALL RMSERR(NSAMR, R4DAT(1,II), RMSO)
      DO 640 J = 1, NSAMR
        CXB(J) = CMPLX(R4DAT(J,II))
      CONTINUE
      DO 650 J = NSAMR + 1, NSAMS
        CXB(J) = 0.0
      CONTINUE
      CALL FORK(NSAMS, CXB, -1.0)
      DO 660 J = 1, NSAMS
        CXB(J) = CXB(J) * C3(J)
      CONTINUE
      CALL FORK(NSAMS, CXB, +1.0)
      DO 670 J = 1, NLAGOP
        IF (NSAMS - NLAGOP + J.GT. NSAMR) GO TO 670
        R4DAT(NSAMS - NLAGOP + J,II) = REAL(CXB(J))
      CONTINUE
      DO 680 J = NLAGOP + 1, NSAMS
        IF (J - NLAGOP.GT. NSAMR) GO TO 680
        R4DAT(J - NLAGOP,II) = REAL(CXB(J))
      CONTINUE
      CALL RMSERR(NSAMR, R4DAT(1,II), RMS1)
      SCALE = RMSO / RMS1
      DO 690 J = 1, NSAMR
        R4DAT(J,II) = R4DAT(J,II) * SCALE
      CONTINUE
    END IF
  END IF
  IF (IOPT.NE. 0) GO TO 610
END IF
IF (IANS.NE. 0) THEN
  GO TO 20
END IF
RETURN
END

```

C Applies linear taper to both sides of time series

```
      DIMENSION X(NX)
      DO 10 I = 1, N1
        X(I) = 0.0
      10 CONTINUE
      DO 20 I = N1 + 1, N2 - 1
        X(I) = REAL(I - N1) * X(I) / REAL(N2 - N1)
      20 CONTINUE
      DO 30 I = N3 + 1, N4 - 1
        X(I) = REAL(N4 - I) * X(I) / REAL(N4 - N3)
      30 CONTINUE
      DO 40 I = N4, NX
        X(I) = 0.0
      40 CONTINUE
      RETURN
      END
```

Appendix A.2

The reflection point loci program

REFLOC

The program REFLOC takes a given model with layer velocities and source and receiver coordinates and calculates the locus of reflection points for each source and receiver pairing. The loci are output to a file which is attached as unit 7 at run-time and these loci are then used by the VSP-CDP mapping program XHRMAP in Appendix A.3.

```

C
C
C Program to find loci of reflection points for a plane parallel
C homogeneous layered model with source at position XS,ZS ; receiver
C at XR(I),ZR(I) ; Locus for each source/receiver pair will be
C contained in the arrays X(I), Z(I) : output to disc for later
C interpolation.
C The method involves raytracing by iterating over different
C initial ray parameters from the source and repeating with the same
C ray parameter from the receiver. The intersection of the 2 raypaths
C yields the reflector locus.
C
C Subroutines called : MAXSN,MINSN,ZERO from REAL*4 TSAR4_1 Lib
C
C Routines from *ghost80
C
C Program output used by XHRMAP program
C N.B. May be necessary to play around with DTHETA (step angle)
C
C Written by M J Findlay 1988
C
C Modified to include anisotropy in raytracing by Ed Kragh 1989
C-----
C Modified to handle Hole-to-surface geometry as well as crosshole
C The following convention should be used: The borehole is at x=0.
C Space to the right is -ve x and space to the left is +ve x. If a
C single sided survey is being processed then -ve x must be used. If a
C two sided survey is being processed then -ve x comes first.
C-----
PROGRAM REFLOC
REAL XS,ZS,XR(24),ZR(24),X(52000),Z(52000),T(52000),
1 DIST(31)
REAL XINTR(31),XINTS(31),THICK(30),VV(30),VH(30),THET(31)
REAL DEPTH(31),TIME(31)
LOGICAL RECLAY
CHARACTER MODE*1,HSTYPE*1,WAVFLD*1,GATHER*1
PI = ACOS(-1.)
PI2 = PI / 2.0
KOUNT = 1
C
C Parameterise the velocity structure layer i ; thickness THICK(I),
C attached to unit 1 Output goes to unit 7
C Read in the survey type..... F=full two sided ..... else single
C sided
C
PRINT *, 'Enter F for full two sided survey else single sided'
READ (1,10) HSTYPE
10 FORMAT (A1)
PRINT *, 'Enter R for common receiver gathers or S for common shot
1s'
READ (1,10) GATHER
PRINT *, 'Enter TOTAL no of loci to be calculated '
READ (1,*) NLOCI
READ (1,*) NMAXX
IF (NMAXX.GT. 30) THEN
WRITE (6,*) ' MAXIMUM NO. OF LAYERS = 30 : PLEASE MODIFY UNIT 1 '
STOP
C
C
C-----
END IF
DEPTH(1) = 0.0
DO 20 I = 1, NMAXX
READ (1,*) THICK(I), VV(I)
DEPTH(I + 1) = DEPTH(I) + THICK(I)
20 CONTINUE
DEPMAX = DEPTH(NMAXX + 1)
READ (1,*) ANISO
DO 30 I = 1, NMAXX
VH(I) = VV(I) + VV(I) * ANISO
30 CONTINUE
READ (1,*) PHI1
PHI = angle of dip +ve for dip UP from borehole to RIGHT
IF (GATHER.EQ. 'S'.OR. GATHER.EQ. 's') PHI1 = -PHI1
PHI = PHI1
READ (1,10) WAVFLD
C
C wavfld is not used in this version of refloc
C Read in display mode (Full scale/ Optimal fill)
READ (1,10) MODE
C
C Read in stepping angle for raysearch
READ (1,*) DTHETA
C
C Read in max number of takeoff angle spreads & NUMBERS FOR
INCREASE IN DTHETA
READ (1,*) MAXIOP, NLP10, NLP100
40 CONTINUE
C
C Read in source and receiver positions
C The borehole is at x=0 and -ve values are to the left.
C If a full survey is being processed do the -ve half first.
IF (GATHER.EQ. 'R'.OR. GATHER.EQ. 'r') THEN
PRINT *, 'Borehole is at x=0 -ve is to the right'
PRINT *, 'If processing a full survey do the -ve half first'
PRINT *, 'Enter XREC ZREC'
READ (1,*) XS,ZS
IF (KOUNT.EQ. 2) XS = -XS
XSPLOT = XS
PRINT *, 'Enter the no of downhole shots'
READ (1,*) NRECS
PRINT *, 'Enter XSHOT ZSHOT for every shot'
DO 50 I = 1, NRECS
READ (1,*) XR(I), ZR(I)
50 CONTINUE
END IF
IF (GATHER.EQ. 'S'.OR. GATHER.EQ. 's') THEN
PRINT *, 'Borehole is at x=0 -ve is to the right'
PRINT *, 'If processing a full survey do the -ve half first'
PRINT *, 'Enter XSHOT ZSHOT'
READ (1,*) XS,ZS

```

```

PRINT *, 'Enter the no of surface receivers'
READ (1,*) NRECS
PRINT *, 'Enter XREC ZREC for every receiver'
DO 60 I = 1, NRECS
  READ (1,*) XR(I), ZR(I)
CONTINUE
DO 70 I = 1, NRECS
  IF (KOUNT.EQ. 1) XR(I) = -XR(I)
CONTINUE
DO 80 I = 1, NRECS
  PRINT *, '  XR(I) = ', XR(I)
CONTINUE
80  CALL MAXSN(NRECS, XR, XXXX, II)
  XSPLT = -XXXX
END IF

C
  Pick out maximum and minimum values of X & Z
  CALL MAXSN(NRECS, XR, XMAX, II)
  XMAX = AMAX1(XMAX, XS)
  CALL MINSN(NRECS, XR, XMIN, II)
  XMIN = AMIN1(XMIN, XS)
  CALL MAXSN(NRECS, ZR, ZMAX, II)
  ZMAX = AMAX1(ZMAX, ZS)
  CALL MINSN(NRECS, ZR, ZMIN, II)
  ZMIN = AMIN1(ZMIN, ZS)
  XMNP = XMIN
  XMAXP = XMAX

C
  Shift axes so that all Xs and Zs are >0
  XSHIFT = 0.0
  ZSHIFT = 0.0
90  IF (XMIN.LT. 0.0) THEN
    XS = XS - XMIN
    DO 100 J = 1, NRECS
      XR(J) = XR(J) - XMIN
    CONTINUE
    XSHIFT = XMIN
    XMIN = 0.0
  END IF
  IF (ZMIN.LT. 0.0) THEN
    ZS = ZS - ZMIN
    DO 110 J = 1, NRECS
      ZR(J) = ZR(J) - ZMIN
    CONTINUE
    ZSHIFT = ZMIN
    ZMIN = 0.0
  END IF

C
  Now apply rotational transformation of PHI to source&receiver
  C posns
  120 DISTS = SORT(XS**2 + ZS**2)
  IF (XS.EQ. 0.0) THEN
    ALPHA = PI/2
  ELSE
    ALPHA = ZS / XS
    ALPHA = ATAN(ALPHA)
  END IF
  DZS = DISTS * (SIN(ALPHA) - SIN(ALPHA - PHI))
  DXS = DISTS * (COS(ALPHA - PHI) - COS(ALPHA))

XS = XS + DXS
ZS = ZS - DZS
PRINT *, 'NEW SOURCE POSN = ', XS, ZS
DO 130 J = 1, NRECS
  DISTR = SORT(XR(J)**2 + ZR(J)**2)
  IF (XR(J).EQ. 0.0) THEN
    ALPHA = PI/2
  ELSE
    ALPHA = ZR(J) / XR(J)
    ALPHA = ATAN(ALPHA)
  END IF
  DZR = DISTR * (SIN(ALPHA) - SIN(ALPHA - PHI))
  DXR = DISTR * (COS(ALPHA - PHI) - COS(ALPHA))
  XR(J) = XR(J) + DXR
  ZR(J) = ZR(J) - DZR
  PRINT *, 'NEW RECEIVER POSN ', J, ' = ', XR(J), ZR(J)
130 CONTINUE

C
  If new source position is lt zero then shift source + all recs
  RESET = 0.0
  IF (XS.LT. 0.0) THEN
    RESET = XS
    DO 140 J = 1, NRECS
      XR(J) = XR(J) - RESET
    CONTINUE
  END IF
  PRINT *, ' Reset = ', RESET

C
  Pick out maximum and minimum values of X & Z IN SHIFTED
  C COORDINATES
  CALL MAXSN(NRECS, XR, X0, II)
  XMAX = AMAX1(XMAX, X0)
  CALL MINSN(NRECS, XR, X1, II)
  XMIN = AMIN1(XMIN, X1)
  CALL MAXSN(NRECS, ZR, Z0, II)
  ZMAX = AMAX1(ZMAX, Z0)
  CALL MINSN(NRECS, ZR, Z1, II)
  ZMIN = AMIN1(ZMIN, Z1)

C
  Find source layer number
  DO 150 I = 1, N1AMAX
    IF (DEPTH(I + 1).GE. ZS) THEN
      LAYSOR = I
      GO TO 160
    END IF
  150 CONTINUE
  160 CONTINUE
  PRINT *, 'SOURCE LAYER = ', LAYSOR

  IF (KOUNT.EQ. 1) THEN
    CALL PAPER(1)
  170  CALL PSPACE(0.1, 0.9, 0.1, 0.9)
    IF (HSTYPE.EQ. 'F'.OR. HSTYPE.EQ. 'F') THEN
      CALL MAP(-XSPLT, XSPLT, DEPMAX, 0.0)
    ELSE
      CALL MAP(0.0, XSPLT, DEPMAX, 0.0)
    END IF
  END IF

```

```

CALL SCALES
END IF
PRINT *, ' XMAXP = ', XMAXP

C *****
C Loop over each receiver ; THETA0 is initial takeoff angle to
C horizontal
C
      IF (KOUNT.EQ.1) WRITE (7,*) NLOC1
      DO 370 J = 1, NRECS
        PRINT *, ' RECEIVER no.: ', J, ' At depth ', ZR(J)
      END IF
      ; -ve ==> upwards

C Find receiver layer
      DO 180 NL = 1, NLMAX
        IF (DEPTH(NL + 1).GE. ZR(J)) THEN
          LAYREC = NL
          GO TO 190
        END IF
      180 CONTINUE
      190 CONTINUE

      IF (LAYREC.EQ. LAYSOR .OR. LAYREC.EQ. 1) THEN
        THETA0 = 0.0
        GO TO 200
      END IF
      IF (VH(LAYREC).LE. VH(LAYREC - 1)) THEN
        THETA0 = 0.0
      ELSE
        TEM1 = VH(LAYREC) ** 2 - VH(LAYREC - 1) ** 2
        TEM2 = TEM1 * VV(LAYREC - 1) ** 2
        TEM3 = VH(LAYREC - 1) ** 4
        THETA0 = ATAN(SQRT(TEM3/TEM2))
        THETA0 = PI2 - THETA0
      END IF
      DTHETA = ABS(DTHETA)
      DTHETA0 = DTHETA
      THETA0 = THETA0 - DTHETA + DTHETA / 100.

C Loop over takeoff angles (Lay parameters)
      LOOP = 1
      DO 320 I = 1, MAXLOP
        IF (I.GT. NLP10) DTHETA = 10. * DTHETA0
        IF (I.GT. NLP100) DTHETA = 100. * DTHETA0
        RECLAY = .FALSE.
        CALL ZERO(31, DIST)
        CALL ZERO(31, XINTR)
        DO 220 JJ = 1, 31
          XINTS(JJ) = -1000
        CONTINUE
        THETA0 = THETA0 + DTHETA
        IF (THETA0.EQ. 0.0) THETA0 = THETA0 + DTHETA
        IF (ABS(THETA0).GE. PI2) GO TO 320
        NLAYER = LAYSOR
        XPO = XS
        ZPO = ZS

C Now start raytracing loop
      230 CONTINUE
      C Ensure ray goes deeper than rec (FOR THETA>0 :: SHALLOWER FOR
      C THETA<0)
        FAC = (ZR(J) - ZPO)
        IF (NLAYER.EQ. LAYREC) THEN
          RECLAY = .TRUE.
          IF (ABS(TAN(THETA)).LT. ABS(FAC/(XR(J) - XPO)))
            GO TO 320
        END IF
        IF (THETA.LT. 0) THEN
          D2 = DEPTH(NLAYER) - ZPO
          ZPO = DEPTH(NLAYER)
        ELSE
          D2 = DEPTH(NLAYER + 1) - ZPO
          ZPO = DEPTH(NLAYER + 1)
        END IF
        IF (ABS(THETA).EQ. PI2) THEN
          DX = 0.0
        ELSE
          DX = D2 / TAN(THETA)
        END IF
        DIST(NLAYER) = DIST(NLAYER) + SQRT(DX**2 + D2**2)
        THET(NLAYER) = THETA
        TEMP1 = VV(NLAYER) ** 2
        TEMP2 = VH(NLAYER) ** 2
        TEMP3 = TEMP1 * COS(THETA) ** 2
        TEMP4 = TEMP2 * SIN(THETA) ** 2
        VTHETA = SQRT(TEMP1*TEMP2/(TEMP3 + TEMP4))
        TIME(NLAYER) = DIST(NLAYER) / VTHETA
        XPO = XPO + DX
        IF (THETA.LT. 0.0) THEN
          XINTS(NLAYER) = XPO
        ELSE
          XINTS(NLAYER + 1) = XPO
        END IF
      C Check for ray passing beyond receiver or out top
      C
        IF (XPO.GT. XR(J)) THEN
          IF (.NOT. RECLAY) THEN
            IF ((THETA.LT. 0.0 .AND. FAC.LT. 0.0) .OR. (THETA.GT.
              0.0 .AND. FAC.GT. 0.0)) GO TO 320
          END IF
          GO TO 250
        C ***** RAY BEYOND RECEIVER ==> STOP THIS RAY & START FROM
        C RECEIVER
          ELSE IF (THETA.LT. 0.0 .AND. NLAYER.EQ. 1) THEN
            GO TO 250
          C ***** RAY OUT TOP ==> STOP THIS RAY AND START FROM RECEIVER
          1 ELSE IF (THETA.GE. 0.0 .AND. NLAYER.EQ. NLMAX)
            THEN
              GO TO 250
          C ***** RAY OUT BOTTOM ==> STOP THIS RAY AND START RAY FRO
          C RECEIVER
            END IF
          C Check for critical reflection
          C

```

```

V0 = VH(NLAYER)
V00 = VV(NLAYER)
IF (THETA.LT. 0.0) THEN
  V1 = VH(NLAYER - 1)
ELSE
  V1 = VH(NLAYER + 1)
END IF
IF (V0 .GE. V1) GO TO 240
END IF

C
TEM = V00 ** 2 * (V1**2 - V0**2) / V0 ** 4
CRIT = ATAN(SORT(1./TEM))
CRIT = PI2 - CRIT
IF (ABS(THETA) .LT. CRIT) THEN
  PRINT *, 'SUPERCritical AT LAYER ', NLAYER
  GO TO 250
ELSE
  GO TO 320
END IF

C***** SUPERCRITICAL ==> STOP THIS RAY & START FROM RECEIVER
C***** APPLY SNEEL'S LAW, UPDATE LAYER NO. & RESTART RAYTRACE
ELSE
  NLAYER = NLAYER + NINT(SIGN(1.0, THETA))
  TEMA = VH(NLAYER) ** 4 * VV(NLAYER - 1) ** 2
  TEMB = VV(NLAYER) ** 2
  TEMC = VH(NLAYER - 1) ** 4 / (TAN(PI2 - ABS(THETA))**2)
  TEMD = VH(NLAYER - 1) ** 2 - VH(NLAYER) ** 2
  TEMD = VV(NLAYER - 1) ** 2
  TEME = TEMB * (TEMC + TEMD*TEMD)
  THETA2 = ATAN(SORT(TEMA/TEME))
  THETA2 = (PI2 - ABS(THETA2)) * SIGN(1.0, THETA)
  THETA = THETA2
  GO TO 230
END IF

C
C Code for tracing from receiver
250 IF (LAYREC .GE. LAYSOR) THETA = THET(LAYREC)
  IF (LAYREC.LT. LAYSOR) THEN
    V0 = VH(LAYREC)
    V00 = VV(LAYREC)
    V1 = VH(LAYSOR)
    IF (V0 .GE. V1) GO TO 260
    TEM = V00 ** 2 * (V1**2 - V0**2) / V0 ** 4
    CRIT = ATAN(SORT(1./TEM))
    CRIT = PI2 - CRIT
    No ray will reach sor layer
    GO TO 260
  END IF

C ***** SUPERCRITICAL ==> STOP THIS RAY & TRY NEXT ONE
260 CONTINUE
  TEMA = VH(LAYSOR) ** 4 * VV(LAYREC) ** 2
  TEMB = VV(LAYSOR) ** 2
  TEMC = (TAN(PI2 - ABS(THET(LAYSOR)))**2)
  TEMD = (1.0/TEMC) * TEMA / TEMB
  TEMD = VH(LAYREC) ** 2 - VH(LAYSOR) ** 2
  TEMD = VV(LAYREC) ** 2
  TEME = TEMC - TEMD * TEMD
  TEME = TEME / (VH(LAYREC)**4)
  IF (TEME.LT. 0.0) GO TO 320
  THETA = ATAN(SORT(1.0/TEME))
  THETA = (PI2 - ABS(THETA))

IF (ABS(THETA) .LT. CRIT) THEN
  PRINT *, ' CRIT RAY ... FOR REC CODE'
  GO TO 320
C ***** SUPERCRITICAL ==> STOP THIS RAY & TRY NEXT ONE
END IF
END IF
RECLAY = .FALSE.
IF (LAYREC .EQ. LAYSOR) THETA = THETA0
XPO = XR(J)
ZPO = ZR(J)
NLAYER = LAYREC

C
C Now start raytracing code for receiver
270 CONTINUE
  FAC = (ZS - ZPO)
  IF (NLAYER .EQ. LAYSOR) THEN
    IF (ABS(TAN(THETA)) .LT. ABS(FAC/(XS - XPO))) THEN
      GO TO 320
    END IF
  END IF
  IF (THETA.LT. 0.0) THEN
    DZ = DEPTH(NLAYER) - ZPO
    ZPO = DEPTH(NLAYER)
  ELSE
    DZ = DEPTH(NLAYER + 1) - ZPO
    ZPO = DEPTH(NLAYER + 1)
  END IF
  IF (ABS(THETA) .EQ. PI2) THEN
    DX = 0.0
  ELSE
    DX = DZ / TAN(THETA)
  END IF
  DIST(NLAYER) = DIST(NLAYER) + SORT(DX**2 + DZ**2)
  TEMP1 = VV(NLAYER) ** 2
  TEMP2 = VH(NLAYER) ** 2
  TEMP3 = TEMP1 * COS(THETA) ** 2
  TEMP4 = TEMP2 * SIN(THETA) ** 2
  VTHETA = SORT(TEMP1*TEMP2/(TEMP3 + TEMP4))
  TIME(NLAYER) = DIST(NLAYER) / VTHETA
  XPO = XPO - DX
  IF (THETA.LT. 0.0) THEN
    XINTR(NLAYER) = XPO
  ELSE
    XINTR(NLAYER + 1) = XPO
  END IF

C
C Check for raypath intersection
IF (THETA.LT. 0.0) THEN
  IF (XINTR(NLAYER) .LE. XINTS(NLAYER)) GO TO 290
ELSE
  IF (XINTR(NLAYER + 1) .LE. XINTS(NLAYER + 1)) GO TO 290
END IF

C
C Check for ray passing beyond source or out top
IF (XPO.LT. XMIN) THEN
  GO TO 320
C***** RAY BEYOND RECEIVER ==> STOP THIS RAY & TRY NEXT ONE

```

```

ELSE IF (THETA .LT. 0.0 .AND. N1LAYER .EQ. 1) THEN
  GO TO 320
C***** RAY OUT TOP =====> STOP THIS RAY AND TRY NEXT ONE
ELSE IF (THETA .GE. 0.0 .AND. N1LAYER .EQ. N1LMAX)
  THEN
    GO TO 320
C***** RAY OUT BOTTOM =====> STOP THIS RAY AND TRY NEXT ONE
END IF
C***** APPLY SNELL'S LAW, UPDATE LAYER NO. & RESTART RAYTRACE
V0 = VH(N1LAYER)
V00 = VV(N1LAYER)
IF (THETA .LT. 0.0) THEN
  V1 = VH(N1LAYER - 1)
ELSE
  V1 = VH(N1LAYER + 1)
END IF
IF (V0 .GE. V1) GO TO 280
  No crit Rays
TEM = V00 ** 2 * (V1**2 - V0**2) / V0 ** 4
CRIT = ATAN(SQRT(1./TEM))
CRIT = PI2 - CRIT
IF (ABS(THETA) .LT. CRIT) THEN
  GO TO 320
C ***** SUPERCRITICAL ==> STOP THIS RAY & TRY NEXT ONE
ELSE
  280 N1LAYER = N1LAYER + NINT(SIGN(1.0,THETA))
      TEMA = VH(N1LAYER) ** 4 * VV(N1LAYER - 1) ** 2
      TEMB = VV(N1LAYER) ** 2
      TEMC = VH(N1LAYER - 1) ** 4 / (TAN(PI2 - ABS(THETA))**2)
      TEMD = VH(N1LAYER - 1) ** 2 - VH(N1LAYER) ** 2
      TEMD0 = VV(N1LAYER - 1) ** 2
      TEME = TEMB * (TEMC + TEMD*TEMD)
      THETA2 = ATAN(SQRT(TEMA/TEME))
      THETA2 = (PI2 - ABS(THETA2)) * SIGN(1.0,THETA)
      THETA = THETA2
      GO TO 270
END IF
C
C Now establish point of intersection of the rays
C
290 IF (THETA .LT. 0.0) THEN
      X(LOOP) = 0.5 * (XINTS(N1LAYER) + XINTR(N1LAYER))
      IF (X(LOOP) .LT. 0.0) WRITE (6,*) 'XINTS(NL), XINTR(NL) = ',
1 XINTS(N1LAYER), XINTR(N1LAYER)
      DZ = (X(LOOP) - XINTS(N1LAYER)) * TAN(THETA)
      Z(LOOP) = DEPTH(N1LAYER) + DZ
      IF (N1LAYER .EQ. LAYSOR) THEN
        X1DASH = XS
        Z1DASH = ZS
      ELSE
        X1DASH = XINTS(N1LAYER + 1)
        Z1DASH = DEPTH(N1LAYER + 1)
      END IF
      IF (N1LAYER .EQ. LAYREC) THEN
        X2DASH = XR(J)
        Z2DASH = ZR(J)
      ELSE
        X2DASH = XINTR(N1LAYER + 1)
        Z2DASH = DEPTH(N1LAYER + 1)
      END IF

```

```

      TEMP1 = VV(N1LAYER) ** 2
      TEMP2 = VH(N1LAYER) ** 2
      TEMP3 = TEMP1 * COS(THETA) ** 2
      TEMP4 = TEMP2 * SIN(THETA) ** 2
      VTHETA = SQRT(TEMP1*TEMP2/(TEMP3 + TEMP4))
      T(LOOP) = SQRT((X2DASH - X1DASH)**2 + (2*Z(LOOP) - Z1DASH -
300 Z2DASH)**2) / VTHETA
      DO 300 M = N1LAYER + 1, N1LMAX, 1
        T(LOOP) = T(LOOP) + TIME(M)
      CONTINUE
ELSE
      X(LOOP) = 0.5 * (XINTS(N1LAYER + 1) + XINTR(N1LAYER + 1))
      IF (X(LOOP) .LT. 0.0) THEN
        WRITE (6,*) 'XINTS(NL+1), XINTR(NL+1) = ', XINTS(N1LAYER + 1),
1 XINTR(N1LAYER + 1)
        WRITE (6,*) 'XINTS = ', XINTS
        WRITE (6,*) 'N1LAYER = ', N1LAYER
        WRITE (6,*) 'THETA = ', THETA
        WRITE (6,*) 'RECEIVER = ', J
        WRITE (6,*) 'LAYREC = ', LAYREC
      END IF
      DZ = (X(LOOP) - XINTS(N1LAYER + 1)) * TAN(THETA)
      Z(LOOP) = DEPTH(N1LAYER + 1) + DZ
      IF (N1LAYER .EQ. LAYSOR) THEN
        X1DASH = XS
        Z1DASH = ZS
      ELSE
        X1DASH = XINTS(N1LAYER)
        Z1DASH = DEPTH(N1LAYER)
      END IF
      IF (N1LAYER .EQ. LAYREC) THEN
        X2DASH = XR(J)
        Z2DASH = ZR(J)
      ELSE
        X2DASH = XINTR(N1LAYER)
        Z2DASH = DEPTH(N1LAYER)
      END IF
      TEMP1 = VV(N1LAYER) ** 2
      TEMP2 = VH(N1LAYER) ** 2
      TEMP3 = TEMP1 * COS(THETA) ** 2
      TEMP4 = TEMP2 * SIN(THETA) ** 2
      VTHETA = SQRT(TEMP1*TEMP2/(TEMP3 + TEMP4))
      T(LOOP) = SQRT((X2DASH - X1DASH)**2 + (2*Z(LOOP) - Z1DASH -
310 Z2DASH)**2) / VTHETA
      DO 310 M = N1LAYER - 1, 1, -1
        T(LOOP) = T(LOOP) + TIME(M)
      CONTINUE
      END IF
      LOOP = LOOP + 1
    END IF
  C
  C Next ray
  C
  320 CONTINUE
      DTHETA = DTHETO
      IF (LOOP .LE. 1) GO TO 370
C
C Now map X,Z coordinates back to dipping space
C apply rotational transformation of minus PHI to source/receiver posns
C
330 PHI = -PHI

```

```

DO 340 K = 1, LOOP - 1
  DISTP = SORT(X(K)**2 + Z(K)**2)
  IF (X(K) .EQ. 0.0) THEN
    ALPHA = PI2
  ELSE
    ALPHA = Z(K) / X(K)
    ALPHA = ATAN(ALPHA)
  END IF
  First reset the x position before the rotation
  The x position was shifted if after the rotation the
  source had -ve x value
  IF (RESET .LT. 0.0) THEN
    X(K) = X(K) + RESET
  END IF
  DZP = DISTP * (SIN(ALPHA) - SIN(ALPHA - PHI))
  DXP = DISTP * (COS(ALPHA - PHI) - COS(ALPHA))
  X(K) = X(K) + DXP + XSHIFT
  Z(K) = Z(K) - DZP + ZSHIFT

  IF (GATHER .EQ. 'S' .OR. GATHER .EQ. 's') X(K) = -X(K)
  IF (KOUNT .EQ. 2) X(K) = -X(K)
340 CONTINUE
  PHI = -PHI
  CALL PTJOIN(X, Z, 1, LOOP - 1, 1)
  WRITE (7,*) (LOOP - 1)
  DO 350 M = 1, LOOP - 1
    WRITE (7,*) T(M), -X(M), Z(M)
350 CONTINUE
360 CONTINUE
370 CONTINUE
  IF (KOUNT .EQ. 1) THEN
    IF (HSTYPE .EQ. 'F' .OR. HSTYPE .EQ. 'f') THEN
      KOUNT = 2
      PHI = -PHI1
      GO TO 40
    END IF
  END IF
END IF
CALL GREND
END

```

Appendix A.3

The VSP-CDP transform program

XHRMAP

The program XHRMAP takes upgoing or downgoing reflection data and transforms it on to reflection point loci calculated by the program REFLOC. Mapped data are subsequently binned on to a rectangular grid and then plotted.

```

C      PROGRAM XHRMAP
C
C      Program to map XHR reflection data to correct CDP position
C      Initially for upgoing wavefield only
C      MUST first run REFLOC1 - to calculate reflection locus for
C      source/receiver
C      Option allows user to stack individual CDP sections in unformatted
C      file
C
C      Subroutines required:      *ghost80 and TimesSeriesAnalysis(REAL*4) 11bs
C                                  SAMNOR (included in this file)
C
C      WRITTEN BY M J FINDLAY 1988
C
C      CHARACTER FNAME*16, A1*20, ANS*1, INSTAK*16, ANSWR*1
C      DIMENSION A1(9)
C      REAL R4DAT(1024,40), RECDER(24), DBGAIN(24), GCMSC1(24),
C      1      REFL(300,40)
C      REAL X(1024), Z(1024), X0(1024), WT(300,40), REF0(300,40)
C      REAL NSTCK(300,40)
C      REAL TT(1500,24), XX(1500,24), ZZ(1500,24), NPOINT(24),
C      1      WTD(300,40)
C      REAL CSGS(300,40,24), CDP(300,24), SECTN(300,40), KNORM(24)
C      INTEGER NFIRST(24), IDPROC(4,24), NCR(2)
C      N1 = 300
C      N2 = 40
C      N3 = 24
C      XOFF = 0.0
C      CALL PAPER(2)
C
C      10 WRITE (6,*)
C      WRITE (6,*) '      XHRMAP MENU'
C      WRITE (6,*) '      ~~~~~ ~~~~~'
C      WRITE (6,*) '      1 Put reflectors on loc1, plot and bin '
C      WRITE (6,*) '      2 Plot binned traces '
C      WRITE (6,*) '      3 Stack binned traces '
C      WRITE (6,*) '      4 Output binned traces '
C      WRITE (6,*) '      5 AS OPTION 1 but for headerless files '
C      WRITE (6,*) '      6 Selective CDP stacking '
C      WRITE (6,*) '      7 Bandpass filter binned data '
C      WRITE (6,*) '      8 Apply AGC to binned record '
C      WRITE (6,*) '      0 STOP '
C      WRITE (6,*)
C      READ (5,*) IANS
C      IF (IANS.EQ.1.OR.IANS.EQ.5) THEN
C      WRITE (6,*) 'True scale plot (Y/N)?'
C      READ (5,100) ANSWR
C      WRITE (6,*) 'Enter data file name : '
C      READ (5,20) FNAME
C      FORMAT (A16)
C      WRITE (6,*) 'Enter shot ID number : '
C      READ (5,*) NSHOT
C
C      WRITE (6,*) 'Enter source offset : '
C      WRITE (6,*) XOFF
C      WRITE (6,*) 'Enter Maximum depth to be plotted : '
C      READ (5,*) DMAX
C      WRITE (6,*) 'ENTER amplitude scaling factor:'
C      READ (5,*) AMPSC1
C      WRITE (6,*) 'Enter polarity (-1=SEG-normal +1=SEG-reverse):'
C      READ (5,*) IPOLAR
C      IF (ABS(IPOLAR).NE.1) GO TO 30
C      WRITE (6,*) 'Enter vertical sampling in metres : '
C      READ (5,*) DELTAZ
C      WRITE (6,*) 'Enter vertical trace spacing in metres : '
C      READ (5,*) DELTAD
C      NTRCEX = INT(XOFF/DELTAD) + 1
C      NDIMZ = INT(DMAX/DELTAZ) + 1
C      WRITE (6,*) 'NTRCEX,NDIMZ = ', NTRCEX, NDIMZ
C      NN = 300 * 40
C      CALL ZERO(NN, REF0(1,1))
C      CALL ZERO(NN, REFL(1,1))
C      CALL ZERO(NN, WT(1,1))
C      LEN = 4104
C      NSAMR = 1024
C      NRECS = 24
C      IF (IANS.EQ.1) THEN
C      IDCODE = (NSHOT - 1) * 25 + 1
C      OPEN (10,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',RECL=LEN)
C      READ (10,REC=IDCODE) A1, SORPOS, NRECS, RECDER, DBGAIN,
C      GCMSC1, NFIRST, NCR, NPROCS, IDPROC, DT
C      IF (DT.EQ.0.0) PRINT *, 'DT = 0'
C      DT = 200.0 * 1E-6
C      DO 40 J = 1, NRECS
C      NREC = IDCODE + J
C      READ (10,REC=NREC) (R4DAT(I,J),I=1,NSAMR)
C      CONTINUE
C      CLOSE (10)
C      ELSE
C      WRITE (6,*) 'Enter (uniform) velocity field in metres/sec:'
C      READ (5,*) VFLAT
C      WRITE (6,*) 'ENTER source depth (metres):'
C      READ (5,*) SORDEP
C      WRITE (6,*) 'Enter shallowest receiver & receiver sepn:'
C      READ (5,*) ZMINR, DZR
C      ZRECS = ZMINR
C      DT = 200E-6
C      FORMAT (8F10.4)
C      OPEN (11,FILE=FNAME,STATUS='OLD',FORM='FORMATTED')
C      DO 60 J = 1, NRECS
C      READ (11,*) JDUM
C      READ (11,50) (R4DAT(I,J),I=1,NSAMR)
C      JD = SORT(XOFF**2 + (SORDEP - ZRECS)**2) / VFLAT
C      NFIRST(J) = 1
C      ZRECS = ZRECS + DZR
C      CONTINUE
C      CLOSE (11)
C      END IF
C
C      Normalise so that Peak RMS amplitude over all traces is 1.0
C      RMSMAX = 0.0
C      DO 80 J = 1, NRECS

```

```

      RMS = 0.0
DO 70 I = 1, NSAMR
      RMS = RMS + RADAT(I,J) * RADAT(I,J)
CONTINUE
      RMS = SORT(RMS/NSAMR)
      IF (RMS.GT. RMSMAX) RMSMAX = RMS
CONTINUE
DO 90 J = 1, NRECS
      DO 90 I = 1, NSAMR
        RADAT(I,J) = RADAT(I,J) / RMSMAX
CONTINUE
      WRITE (6,*) 'Apply T-squared ramp (Y/N) ?'
      READ (5,100) ANS
      FORMAT (A1)
      IF (ANS.EQ. 'Y'.OR. ANS.EQ. 'Y') THEN
        DO 120 J = 1, NRECS
          DO 110 I = 1, NSAMR
            RADAT(I,J) = RADAT(I,J) * (I - 1) ** 2 / 10000
          CONTINUE
        CONTINUE
      END IF
      WRITE (6,*) 'Enter name of file of reflection loci : '
      READ (5,20) FNAME
      OPEN (1,FILE=FNAME,STATUS='OLD')
      READ (1,*) NRECS
      DO 140 J = 1, NRECS
        READ (1,*) NPOINT(JJ)
        DO 130 II = 1, NPOINT(JJ)
          READ (1,*) TT(II,JJ), XX(II,JJ), ZZ(II,JJ)
        CONTINUE
      CONTINUE
    130 CONTINUE
    140 CLOSE (1)

C
C PLOT REFLECTORS ON LOC1
C
      POX = 0.1
      PIX = 0.9
      IF (ANSWR.EQ. 'Y'.OR. ANS.EQ. 'Y') THEN
        DPX = XOFF / DMAX
        PIX = POX + 0.8 * DPX
      END IF
      CALL PSPACE(POX, PIX, 0.1, 0.9)
      CALL MAP(0.0, XOFF, DMAX, 0.0)
      CALL AXES
      CALL BORDER
      TTRMS = 0.0
      DO 200 J = 1, NRECS
        PRINT *, 'CHANNEL = ', J
        WRITE (4,*) 'RECEIVER #', J
        TG = (NFIRST(J) - 1) * DT
        WRITE (4,*) 'First break time = ', TG
        INDT = 2
        NO = 0
        TTER = TG - TT(1,J)
        TTRMS = TTRMS + TTER ** 2
        WRITE (4,*) 'RECEIVER ', J, ' TTER = ', TTER
        IF (ABS(TTER).GT. 0.001) WRITE (6,*) J, ' TTER = ',
          TTER
      CONTINUE
      WRITE (6,*) NFIRST(J), NSAMR
      DO 160 I = NFIRST(J), NSAMR
        INDI = I
        TSAMP = (I - 1) * DT
        IF (INDT.GT. NPOINT(J)) GO TO 170
        IF (TSAMP.LT. TT(1,J)) THEN
          NO = NO + 1
          GO TO 160
        END IF
        NPJ = NPOINT(J)
        IF (TSAMP.GT. TT(NPJ,J)) GO TO 170
        IF (TSAMP.GT. TT(INDT,J)) THEN
          INDT = INDT + 1
          GO TO 150
        END IF
        DELTAT = TT(INDT,J) - TT(INDT - 1,J)
        FAC = (TSAMP - TT(INDT - 1,J)) / DELTAT
        XO(I) = XX(INDT - 1,J) + FAC * (XX(INDT,J) - XX(INDT -
          1,J))
        Z(I) = ZZ(INDT - 1,J) + FAC * (ZZ(INDT,J) - ZZ(INDT -
          1,J))
        X(I) = XO(I) + AMPSCL * RADAT(I,J)
      CONTINUE
    160 CONTINUE
C
C Simple plot of repositioned traces
C
    170 LEN = INDI - 1
        CALL PJOIN(X, Z, NFIRST(J) + NO, LEN, 1)
        DO 180 I = NFIRST(J) + NO, LEN
          IF (RADAT(I,J)*TPOLAR.LT. - 0.01) THEN
            CALL POSITN(XO(I), Z(I))
            CALL JOIN(X(I), Z(I))
          END IF
        CONTINUE
    180 CONTINUE
C
C Now calculate array REFL(300,40) of binned traces for surface reflectn
C
      DO 190 I = NFIRST(J), NSAMR
        NHORI = INT(XO(I)/DELTAD) + 1
        NVERT = INT(Z(I)/DELTAZ) + 1
        IF (NHORI.GT. NTRCEX.OR. NVERT.GT. NDIMZ)
          GO TO 190
        IF (NHORI.LE. 0.OR. NVERT.LE. 0) GO TO 190
        G2 = (XO(I) - (NHORI - 1)*DELTAD) / DELTAD
        G1 = 1.0 - G2
        F2 = (Z(I) - (NVERT - 1)*DELTAZ) / DELTAZ
        F1 = 1.0 - F2
        REFL(NVERT,NHORI) = RADAT(I,J) * F1 * G1 + REFL(NVERT,
          NHORI)
        REFL(NVERT,NHORI + 1) = RADAT(I,J) * F1 * G2 + REFL(
          NVERT,NHORI + 1)
        REFL(NVERT + 1,NHORI) = RADAT(I,J) * F2 * G1 + REFL(
          NVERT + 1,NHORI)
        REFL(NVERT + 1,NHORI + 1) = RADAT(I,J) * F2 * G2 +
          REFL(NVERT + 1,NHORI + 1)
        WT(NVERT,NHORI) = F1 * G1 + WT(NVERT,NHORI)
        WT(NVERT,NHORI + 1) = F1 * G2 + WT(NVERT,NHORI + 1)
        WT(NVERT + 1,NHORI) = F2 * G1 + WT(NVERT + 1,NHORI)
        WT(NVERT + 1,NHORI + 1) = F2 * G2 + WT(NVERT + 1,
          NHORI + 1)
      CONTINUE
    190 CONTINUE
    200 CONTINUE

```

```

C C No longer need record of weighting
C
C REFMAX = 0.0
WRITE (6,*) 'NTRCEX, NDIWZ ', NTRCEX, NDIWZ
DO 220 J = 1, NTRCEX
  DO 210 II = 1, NDIWZ
    IF (WT(II,J)) .EQ. 0.0) WT(II,J) = 1.0
    REFL(II,J) = REFL(II,J) / WT(II,J)
    REFMAX = AMAX1(REFMAX, ABS(REFL(II,J)))
  CONTINUE
  TTRMS = SORT(TTRMS/NRECS)
  WRITE (4,*) 'RMS TERROR = ', TTRMS
  CALL FRAME
CONTINUE
210
220
C End of option 1
C
C ELSE IF (IANS .EQ. 2) THEN
C
C Plot binned traces
C
  IF (XOFF .EQ. 0.0) THEN
    WRITE (6,*) 'True scale plot (Y/N) ?'
    READ (5,100) ANSWR
    WRITE (6,*) 'Enter source offset : '
    READ (5,*) XOFF
    WRITE (6,*) 'Enter Maximum depth to be plotted : '
    READ (5,*) DMAX
    WRITE (6,*) 'ENTER amplitude scaling factor: '
    READ (5,*) AMPSCI
    WRITE (6,*) 'Enter vertical sampling in metres : '
    READ (5,*) DELTAZ
    WRITE (6,*) 'Enter vertical trace spacing in metres : '
    READ (5,*) DELTAD
    NTRCEX = INT(XOFF/DELTAD) + 1
    WRITE (6,*) NTRCEX
  END IF
  WRITE (6,*) 'Enter polarity (-1=SEG-normal +1=SEG-reverse) : '
  READ (5,*) IPOLAR
  IF (ABS(IPOLAR) .NE. 1) GO TO 230
  WRITE (6,*) 'Normalise traces by RMS(1), PEAK(2), None(0) : '
  READ (5,*) IVAL
  PEAKMX = 1.0
  IF (IVAL .EQ. 1) THEN
    WRITE (6,*) 'Enter depth range for RMS calculation : '
    READ (5,*) Z0, Z1
    NZ0 = INT(Z0/DELTAD) + 1
    NZ1 = INT(Z1/DELTAD) + 1
    PEAKMX = 0.0
    DO 250 I = 1, NZ
      RMS = 0.0
      PEAK = 0.0
      DO 240 J = NZ0, NZ1
        RMS = RMS + REFL(J,I) ** 2
        IF (ABS(REFL(J,I)) .GT. PEAK) PEAK = ABS(REFL(J,I))
      )
    CONTINUE
    RMS = RMS / REAL(NI)
    RMS = SORT(RMS)
  240
  1
  230
  250
  260
  270
  280
  290
  300
  310
  IF (RMS .GT. 0.0) PEAK = PEAK / RMS
  IF (PEAK .GT. PEAKMX) PEAKMX = PEAK
  IF (RMS .GT. 0.0) THEN
    RNORM(I) = RMS
  ELSE
    RNORM(I) = 1.0
  END IF
  CONTINUE
  PEAKMX = PEAKMX / 1.4
  ELSE IF (IVAL .EQ. 2) THEN
    DO 260 I = 1, NZ
      CALL MAXSN(N1, REFL(1,I), PEAK, IND)
      CALL MINSN(N1, REFL(1,I), PEAK0, IND)
      IF (ABS(PEAK0) .GT. PEAK) PEAK = ABS(PEAK0)
      RNORM(I) = PEAK
    CONTINUE
  ELSE
    N12 = N1 * NZ
    CALL MAXSN(N12, REFL(1,1), PEAK, IND)
    CALL MINSN(N12, REFL(1,1), PEAK0, IND)
    IF (ABS(PEAK0) .GT. PEAK) PEAK = ABS(PEAK0)
    DO 270 I = 1, NZ
      RNORM(I) = PEAK
    CONTINUE
  END IF
  NREFL = N1 * NZ
  CALL MAXSN(NREFL, REFL(1,1), REFMX, IND)
  CALL MINSN(NREFL, REFL(1,1), REFMIN, IND)
  IF (ABS(REFMIN) .GT. REFMX) REFMX = ABS(REFMIN)
  IF (AMPSCL .EQ. 0.0) AMPSCL = DELTAD / REFMX
  PIX = 0.1
  PIX = 0.9
  IF (ANSWR .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
    DPX = XOFF / DMAX
    PIX = PIX + 0.8 * DPX
  END IF
  CALL PSPACE(PIX, PIX, 0.1, 0.9)
  CALL MAP(0.0, XOFF, DMAX, 0.0)
  CALL BORDER
  CALL AXES
  NDIWZ = INT(DMAX/DELTAD) + 1
  WRITE (6,*) 'NDIWZ = ', NDIWZ
  DO 280 J = 1, NDIWZ
    Z(J) = (J - 1) * DELTAZ
    CONTINUE
  DO 310 I = 1, NTRCEX
    X1 = (I - 1) * DELTAD
    IF (RNORM(I) .EQ. 0.0) RNORM(I) = 1.0
    IF (RNORM(I) .NE. 1.0) AMPSCL = DELTAD / (RNORM(I) * PEAKMX)
    DO 290 J = 1, NDIWZ
      X(J) = X1 + REFL(J,I) * AMPSCL
    CONTINUE
    CALL PJOIN(X, Z, 1, NDIWZ, 0)
    DO 300 J = 1, NDIWZ
      IF (REFL(J,I) * IPOLAR .LT. - 0.01) THEN
        CALL POSITN(X1, Z(J))
        CALL JOIN(X(J), Z(J))
      END IF
    CONTINUE
  300
  310

```

```

CALL FRAME
C
C End of option 2
C
ELSE IF (IANS .EQ. 3) THEN
C
C Stack binned traces
C
NN = 300 * 40
CALL ZERO(NN, REFO)
CALL ZERO(NN, REFL)
CALL ZERO(NN, WT)
WRITE (6,*) 'Enter dimensions of section (NX,NZ) : '
READ (5,*) NTRCEX, NNDIMZ
WRITE (6,*) 'Enter no. of common shot records to be stacked'
READ (5,*) NSTAK
DO 320 J = 1, NTRCEX
DO 320 I = 1, NNDIMZ
NSTACK(I,J) = NSTAK
320 CONTINUE
DO 360 II = 1, NSTAK
WRITE (6,*) 'Enter input filename for stack : '
READ (5,20) INSTAK
WRITE (6,*) 'NTRCEX , NNDIMZ =', NTRCEX, NNDIMZ
OPEN (8,FILE=INSTAK,STATUS='OLD')
READ (8,340) ((REFO(I,J)), I=1,NNDIMZ, J=1,NTRCEX)
CONTINUE
CLOSE (8)
FORMAT (16F10.6)
DO 350 J = 1, NTRCEX
DO 350 I = 1, NNDIMZ
REFL(I,J) = REFL(I,J) + REFO(I,J)
IF (ABS(REFO(I,J)) .LT. 1E-3) NSTACK(I,J) =
NSTACK(I,J) - 1
350 CONTINUE
DO 370 J = 1, NTRCEX
DO 370 I = 1, NNDIMZ
IF (NSTACK(I,J) .EQ. 0.0) NSTACK(I,J) = 1.0
REFL(I,J) = REFL(I,J) / NSTACK(I,J)
370 CONTINUE
C
C End of option 3
C
ELSE IF (IANS .EQ. 4) THEN
C
C output binned traces
C
WRITE (6,*) 'Output the section (I=Y 0=Output bin) : '
READ (5,*) NANS
WRITE (6,*) 'Enter OUTPUT filename : '
READ (5,20) INSTAK
NLEN = NNDIMZ * NTRCEX
OPEN (8,FILE=INSTAK,STATUS='UNKNOWN')
WRITE (6,*) 'NTRCEX , NNDIMZ =', NTRCEX, NNDIMZ
IF (NANS .EQ. 0) THEN
WRITE (8,340) ((REFL(I,J)), I=1,NNDIMZ, J=1,NTRCEX)
ELSE
WRITE (8,340) ((SECTN(I,J), I=1,NNDIMZ, J=1,NTRCEX)
END IF

CLOSE (8)
C
C End option 4
C
ELSE IF (IANS .EQ. 6) THEN
C
C OPTION TO USE CDP SELECTIVE STACKING
C
WRITE (6,*) 'Enter dimensions of binned arrays : NX,NZ '
READ (5,*) NTRCEX, NNDIMZ
WRITE (6,*) 'Enter no. of common shot gathers : NY '
READ (5,*) NGATH
FORMAT (A10)
DO 390 KK = 1, NGATH
WRITE (6,*) 'Enter name of file containing binned gather #'
1
READ (5,380) FNAME
OPEN (12,FILE=FNAME,STATUS='OLD')
READ (12,340) ((CSGS(I,J,KK), I=1,NNDIMZ, J=1,NTRCEX)
CLOSE (12)
390 CONTINUE
WRITE (6,*) 'Enter horizontal trace separation in metres : DX : '
READ (5,*) DELTAD
WRITE (6,*) 'Enter vertical sampling interval in metres : DZ '
READ (5,*) DELTAZ
CALL CDPSTK(N1, N2, NGATH, NNDIMZ, NTRCEX, DELTAD, DELTAZ,
NGATH, CSGS, CDP, SECTN)
1
C
C END OF OPTION 6
C
ELSE IF (IANS .EQ. 7) THEN
WRITE (6,*) 'Enter sample interval in metres : '
READ (5,*) DELTAZ
NSC = 1024 * 40
CALL ZERO(NSC, RADAT)
DO 410 J = 1, NTRCEX
DO 400 I = 1, NNDIMZ
RADAT(I,J) = REFL(I,J)
400 CONTINUE
CONTINUE
WRITE (6,*) 'Plot spectra I=Y 0=N?'
READ (5,*) NPLOT
IF (NPLOT .EQ. 1) THEN
CALL FOURFFT(1024, 40, RADAT, DELTAZ)
END IF
WRITE (6,*) 'Filter traces I=Y 0=N?'
READ (5,*) NFILT
IF (NFILT .EQ. 1) THEN
CALL FILTER(1024, 40, RADAT, DELTAZ)
END IF
DO 420 J = 1, NTRCEX
DO 420 I = 1, NNDIMZ
REFL(I,J) = RADAT(I,J)
420 CONTINUE
ELSE IF (IANS .EQ. 8) THEN
WRITE (6,*) 'Enter sample interval in metres : '
READ (5,*) DELTAZ
WRITE (6,*) 'Do you want to apply muting prior to AGC (I=Y 0=N)
1?
READ (5,*) MUTAGC

```

```

430      IF (MUTAGC.EQ. 1) THEN
         WRITE (6,*) 'Enter trace no., start depth, end depth:'
         READ (5,*) NTRMUT, ZMUTO, ZMUT1
         IF (NTRMUT.EQ. 0) GO TO 450
         NMUTO = NINT(ZMUTO/DELTAZ) + 1
         NMUT1 = NINT(ZMUT1/DELTAZ) + 1
         DO 440 IMUT = NMUTO, NMUT1
            REFL(IMUT, NTRMUT) = 0.0
         CONTINUE
      GO TO 430
440
450      END IF
         WRITE (6,*) 'Enter length of AGC to be applied in metres : '
         READ (5,*) RLEN
         WRITE (6,*) 'Enter low amplitude cutoff mute : '
         READ (5,*) OFF
         DO 460 JMUT = 1, NTRCEX
            DO 460 IMUT = 1, NDIMZ
               IF (ABS(REFL(IMUT, JMUT)) .LT. OFF) THEN
                  REFL(IMUT, JMUT) = 0.0
               END IF
            END IF
         CONTINUE
460      LENAGC = NINT(RLEN/DELTAZ)
         WRITE (6,*) 'NDIMZ, NTRCEX = ', NDIMZ, NTRCEX
         CALL AGC(N1, N2, REFL, LENAGC)
         ELSE IF (IANS.EQ. 0) THEN
            STOP
         END IF
470      END IF
480      GO TO 10
      END

C
C      SUBROUTINE SAMNOR(LX, LY, X, NSAMF, NSAML)
C
C      NORMALIZES 2D ARRAY TO PEAK WITHIN REQUIRED RANGE (NORMALISES ENTIR
C      ARRAY)
C
      DIMENSION X(LX,LY)
      RMAX = 0.0
      DO 20 J = 1, LY
         DO 10 I = NSAMF, NSAML
            IF (I.EQ. 7) GO TO 10
            IF (ABS(X(I,J)) .GE. RMAX) RMAX = ABS(X(I,J))
         CONTINUE
      IF (RMAX.EQ. 0.0) GO TO 50
      DO 40 J = 1, LY
         DO 30 I = 1, LX
            IF (I.EQ. 7) X(I,J) = X(I - 1, J) + X(I + 1, J)
            X(I,J) = X(I,J) / RMAX
         CONTINUE
      CONTINUE
50      RETURN
      END

```

Appendix A.4

The raytracing program

TRACER

The program TRACER carries out two-point raytracing within a velocity model defined at points on a rectangular grid. The algorithm presented by Chang and McMechan (1986) is used. The program traces rays from each source point to each grid point and then from each receiver point to each grid point, thus providing an image time for all the grid points for each combination of source and receiver positions. The image times can then be used by the reverse-time finite difference migration program.

```

C program to raytrace through 2-d velocity model to obtain image times
C for reverse time migration.
C
C PROGRAM TRACE1
C
C Algorithm from Chang & McMechan (source Lee & Stewart)
C
C At any point along ray there exist 4 quantities :
C   x , dx/ds , z , dz/ds
C   x & z are spatial coordinates ds is along ray.
C   These 4 quantities are the initial conditions required to extend
C   the ray by one additional step.
C   Let  $U = 1/V$  be the slowness;  $velocity = V(x,z)$ 
C   Define  $Ux = dU/dx$   $Uz = dU/dz$ 
C   Rewrite ray equation as:  $dW1/ds = W2$ 
C                            $dW2/ds = V * (Ux - G*W2)$ 
C                            $dW3/ds = W4$ 
C                            $dW4/ds = V * (Uz - G*W4)$ 
C
C  $W1 = x$  ,  $W2 = dx/ds$  ,  $W3 = z$  ,  $W4 = dz/ds$  ,  $G = Ux*W2 + Uz*W4$ 
C
C Solve for  $W1, W2, W3, W4$  by 4th order Runge-Kutta algorithm
C Travel time for a ray is calculated by
C
C   SIGMA ( ds )
C   1 ( ---- ) where  $V1$  is the average velocity on the 1th
C   ( V1 ) segment
C
C
C PARAMETER (N1=64, N2=90, N3=128, N4=180, NVAR=4, PI=3.1415926535)
C DOUBLE PRECISION T, X(4), X, H, W(4,7), XO, ZO, XOMIN, XOMAX
C DOUBLE PRECISION ZOMIN, ZOMAX, V(N1,N2), V0, UX0, UZ0
C DOUBLE PRECISION H0, XS(24), ZS(24), SMOTH1, SMOTH2
C DOUBLE PRECISION XVAL(50000), ZVAL(50000), TVAL(50000)
C DOUBLE PRECISION LAMBDA(104), MU(104), FF(N3*N4), W1(50000)
C DOUBLE PRECISION SIGMA, CO(50000), CT(50000), FVAL(50000)
C DOUBLE PRECISION XVAL0(N3*N4), ZVAL0(N3*N4)
C DOUBLE PRECISION XPT(1), ZPT(1), VAL(1)
C DOUBLE PRECISION XAXIS(N3), ZAXIS(N4), WRK(250000), SMOOTH
C REAL XCOORD(10000), ZCOORD(10000), FFL(N3,N4), CLEVEL(15)
C INTEGER IW2, ADRES(N3*N4), PX, PZ
C INTEGER VPOINT(51000), IWRK(50000)
C CHARACTER*1 START
C
C EXTERNAL FCN
C COMMON /DATA/ V0, UX0, UZ0
C IW2 = 6
C WRITE (6,*) 'NUMBER OF SOURCES : '
C READ (5,*) NSOURC
C DO 10 NS = 1, NSOURC
C   WRITE (6,*) 'ENTER SOURCE POSITION (X,Z) in metres : '
C   READ (5,*) XS(NS), ZS(NS)
C10 CONTINUE
C   WRITE (6,*) '***** SOURCE : ', XS(1), ZS(1), ' *****'
C   WRITE (6,*) 'ENTER XMIN,XMAX , ZMIN,ZMAX : '
C   READ (5,*) XMIN, XMAX, ZMIN, ZMAX
C   WRITE (6,*) 'Enter velocity grid spacing in metres: '
C   READ (5,*) H0
C   WRITE (6,*) 'ENTER image time grid spacing in metres: '
C   READ (5,*) H0IM
C
C WRITE (6,*) 'Enter DT in milliseconds for SMOOTH estimation : '
C READ (5,*) DT
C DT = DT * 1.E-3
C WRITE (6,*) 'Enter values of SMOOTH for velocity and time grids: '
C READ (5,*) SMOTH1, SMOTH2
C NX = INT((XMAX - XMIN)/H0)
C NXIM = INT((XMAX - XMIN)/H0IM)
C NZ = INT((ZMAX - ZMIN)/H0)
C NZIM = INT((ZMAX - ZMIN)/H0IM)
C IF (NX.GT. N1 .OR. NZ.GT. N2) THEN
C   WRITE (6,*) ' *** WARNING --- ARRAYS TOO SMALL *** '
C   STOP
C END IF
C WRITE (6,*) 'ENTER RAY STEP LENGTH IN METRES : '
C READ (5,*) H
C WRITE (6,*) 'ENTER NUMBER OF RAYS TO BE TRACED : '
C READ (5,*) NANG
C DANG = PI / REAL(NANG)
C WRITE (6,*) 'Enter fine stepping angle (radians) : '
C READ (5,*) FINANG
C IF (ABS(FINANG) .LT. PI/2.) THEN
C   WRITE (6,*) 'Enter stepping interval increase (times) : '
C   READ (5,*) ANGFAC
C   WRITE (6,*) 'Enter number of rays for fine stepping : '
C   READ (5,*) NRAYFN
C   NANG = NANG + NRAYFN
C END IF
C WRITE (6,*) 'Enter NUMBER of layers : '
C READ (5,*) NLAYER
C IDEPTH = 0
C VMAX = 0.0
C CDEP = 0.0
C CALL ZERO(80, MU)
C CALL ZERO(40, LAMBDA)
C DO 30 NL = 1, NLAYER
C   WRITE (6,*) 'ENTER VELOCITY OF MEDIUM ', NL, ' (metres / sec)'
C   READ (5,*) VL
C   IF (VL.GT. VMAX) VMAX = VL
C   WRITE (6,*) 'VL = ', VL
C   WRITE (6,*) 'ENTER THICKNESS OF LAYER ', NL, ' (metres)'
C   READ (5,*) THICK
C   WRITE (6,*) 'THICK = ', THICK
C   CDEP = CDEP + THICK
C   NTHICK = NINT(THICK/H0)
C   WRITE (6,*) 'NTHICK = ', NTHICK
C   WRITE (6,*) 'DEPTH = ', IDEPTH
C   DO 20 IZ = 1, NTHICK
C     WRITE (6,*) 'IZ = ', IZ
C     DO 20 IX = 1, NX
C       V(IX,IZ + IDEPTH) = VL
C20 CONTINUE
C   IDEPTH = IDEPTH + NTHICK
C30 CONTINUE
C   WRITE (6,*) 'READ VELOCITIES OK'
C   IFAIL = 0
C   WRITE (6,*) 'IFAIL = ', IFAIL
C
C E02DCF-NAG GIVES LEAST SQUARE BICUBIC SPLINE FIT TO SURFACE
C
C ICONT0,PX,PZ,XVAL0,ZVAL0,LAMBDA,MU,VPOINT,NPOINT AS E02ZAF ABOVE

```

```

C      EVAL(ICONT0) IS GRID POINT VALUE
C      C0(NC) COEFFICIENTS OF FIT C0((PY-4) * (I-1) + J) IS C(I,J)
C      SMOOTH Smoothing factor for spline accuracy
C
      SMOOTH = SMOTH1
      NXEST = 50
      NZEST = NZ
      LMRK = 4 * (NX + NZ) + 11 * (NXEST + NZEST) + NXEST * NZ + NZ + 54
      LMRK = LMRK * 2
      LMRK = 3 + NX + NZ + NXEST + NZEST
      LMRK = LMRK * 2
      IFAIL = 0
      START = 'C'
      NKNOTX = 8
      NKNOTZ = 8
C
C      ARRAY EVAL SHOULD CONTAIN VALUES S.T. F(NZ*(Q-1)+R)
C      contains      grid point (X(Q),Z(R))
C
      ICOUNT = 0
      DO 40 IX = 1, NX
      DO 40 IZ = 1, NZ
      ICOUNT = ICOUNT + 1
      EVAL(ICOUNT) = V(IX,IZ)
40 CONTINUE
      DO 50 IZ = 1, NZ
      ZAXIS(IZ) = REAL(IZ) * H0
50 CONTINUE
      DO 60 IX = 1, NX
      XAXIS(IX) = REAL(IX) * H0
60 CONTINUE
      WRITE (6,*) 'NX,NZ,NXEST,NZEST,LMRK,LMRK', NX, NZ, NXEST, NZEST,
1      LMRK, LMRK
      CALL E02DCF(START, NX, XAXIS, NZ, ZAXIS, EVAL, SMOOTH, NXEST,
1      NZEST, NKNOTX, LAMBDA, NKNOTZ, MU, CO, SIGMA, WRK, LMRK,
2      LMRK, LMRK, IFAIL)
      WRITE (6,*) 'IFAIL = ', IFAIL
      WRITE (6,*) 'SIGMA = ', SIGMA
      DO 70 I = 1, 104
      WRITE (12,*) I, LAMBDA(I), MU(I)
70 CONTINUE
      IFAIL = 0
      WRITE (6,*) 'IFAIL = ', IFAIL
C
C      NOW BEGIN RAYTRACING FROM SOURCE
C
      SCA = (ZMAX - ZMIN) / (XMAX - XMIN)
      IF (SCA .LT. 1.0) THEN
        PX0 = 0.1
        PX1 = 0.9
        PZL = 0.8 * SCA
        PZ0 = 0.1
        PZ1 = PZ0 + PZL
      ELSE
        PZ0 = 0.1
        PZ1 = 0.9
        PXL = 0.8 / SCA
        PX0 = 0.1
        PX1 = PX0 + PXL
      END IF
C
C      END IF
C      DO 200 NS = 1, NSOURC
C      X0 = XS(NS)
C      Z0 = ZS(NS)
C      WRITE (7,*) 'SOURCE NUMBER :', NS
C      CALL PAPER(1)
C      CALL ESPACE(PX0, PXL, PZ0, PZ1)
C      CALL MAP(XMIN, XMAX, ZMAX, ZMIN)
C      CALL SCALES
C
C      LOOP OVER TAKEOFF ANGLES MEASURED FROM HORIZONTAL (DOWN=+VE)
C
      ANG = -PI / 2.
      ICOUNT = 0
      WRITE (6,*) 'RAY-TRACING FOR SOURCE #', NS
      IFIN = 0
      DO 100 I = 1, NANG
      WRITE (7,*) 'RAY NUMBER ', I
      IF (ANG .GT. FINANG .AND. IFIN .LT. NRAYFN) THEN
        ANG = ANG + DANG / ANGFAC
        IFIN = IFIN + 1
      ELSE
        ANG = ANG + DANG
      END IF
C
C      SET UP INITIAL RAY CONDITIONS
C
      X = -H
      T = -H
      Y(1) = X0
      Y(2) = COS(ANG)
      Y(3) = Z0
      Y(4) = SIN(ANG)
C
C      STEP ALONG RAY
C
      DO 80 IRAY = 1, 2000
      ICOUNT = ICOUNT + 1
      IF (ICOUNT .GT. 50000) THEN
        WRITE (6,*) 'STOPPED ... ARRAY EVAL TOO SMALL'
        STOP
      END IF
      T = T + H
      X = X + H
      XCOORD(IRAY) = REAL(Y(1))
      ZCOORD(IRAY) = REAL(Y(3))
      XVAL(ICOUNT) = XCOORD(IRAY)
      ZVAL(ICOUNT) = ZCOORD(IRAY)
      XPT(1) = Y(1)
      ZPT(1) = Y(3)
      NEARX = NINT(XPT(1)/H0)
      NEARZ = NINT(ZPT(1)/H0)
      IFAIL = 0
      NC = (NKNOTX - 4) * (NKNOTZ - 4)
      NADRES = (NKNOTX - 7) * (NKNOTZ - 7)
      N = 1
      NPOINT = N + NADRES
      IF (XPT(1) .LT. LAMBDA(4)) XPT(1) = LAMBDA(4)
      IF (XPT(1) .GT. LAMBDA(NKNOTX - 3)) XPT(1) = LAMBDA(NKNOTX -
3)

```

```

1      IF (ZPT(1) .LT. MU(4)) ZPT(1) = MU(4)
      IF (ZPT(1) .GT. MU(NKNOTZ - 3)) ZPT(1) = MU(NKNOTZ - 3)
      CALL E02ZAF(NKNOTX, NKNOTZ, LAMBDA, MU, N, XPT, ZPT, VPOINT,
1      NPOINT, ADRES, NADRES, IFAIL)
      IFAIL = 0
      N = 1
      CALL E02DBF(N, NKNOTX, NKNOTZ, XPT, ZPT, VAL, LAMBDA, MU,
1      VPOINT, NPOINT, CO, NC, IFAIL)
      VO = VAL(1)
      IF (VO .LT. 1.0) THEN
        WRITE (6,*) 'VO = ', VO
        WRITE (6,*) 'XPT, ZPT = ', XPT, ZPT
      END IF
      IFAIL = 0
      XPT(1) = XPT(1) - HO / 5.
      DELTAX = HO *.4
      IF (XPT(1) .LT. LAMBDA(4)) THEN
        XPT(1) = LAMBDA(4)
        DELTAX = HO / 5.
      END IF
      N = 1
      CALL E02ZAF(NKNOTX, NKNOTZ, LAMBDA, MU, N, XPT, ZPT, VPOINT,
1      NPOINT, ADRES, NADRES, IFAIL)
      IFAIL = 0
      N = 1
      CALL E02DBF(N, NKNOTX, NKNOTZ, XPT, ZPT, VAL, LAMBDA, MU,
1      VPOINT, NPOINT, CO, NC, IFAIL)
      UXO = VAL(1)
      IFAIL = 0
      XPT(1) = XPT(1) + DELTAX
      IF (XPT(1) .GT. LAMBDA(NKNOTX - 3)) THEN
        XPT(1) = LAMBDA(NKNOTX - 3)
        DELTAX = HO / 5.
      END IF
      N = 1
      CALL E02ZAF(NKNOTX, NKNOTZ, LAMBDA, MU, N, XPT, ZPT, VPOINT,
1      NPOINT, ADRES, NADRES, IFAIL)
      IFAIL = 0
      N = 1
      CALL E02DBF(N, NKNOTX, NKNOTZ, XPT, ZPT, VAL, LAMBDA, MU,
1      VPOINT, NPOINT, CO, NC, IFAIL)
      VXO = (VAL(1) - UXO) / (DELTAX)
      UXO = -VXO / VO ** 2
      IFAIL = 0
      XPT(1) = Y(1)
      ZPT(1) = Y(3) - HO / 5.
      DELTAZ = HO * 2. / 5.
      IF (ZPT(1) .LT. MU(4)) THEN
        ZPT(1) = MU(4)
        DELTAZ = HO / 5.
      END IF
      N = 1
      CALL E02ZAF(NKNOTX, NKNOTZ, LAMBDA, MU, N, XPT, ZPT, VPOINT,
1      NPOINT, ADRES, NADRES, IFAIL)
      IFAIL = 0
      N = 1
      CALL E02DBF(N, NKNOTX, NKNOTZ, XPT, ZPT, VAL, LAMBDA, MU,
1      VPOINT, NPOINT, CO, NC, IFAIL)
      UZO = VAL(1)
      IFAIL = 0

2PT(1) = ZPT(1) + DELTAZ
      IF (ZPT(1) .GT. MU(NKNOTZ - 3)) THEN
        ZPT(1) = MU(NKNOTZ - 3)
        DELTAZ = HO / 5.
      END IF
      N = 1
      CALL E02ZAF(NKNOTX, NKNOTZ, LAMBDA, MU, N, XPT, ZPT, VPOINT,
1      NPOINT, ADRES, NADRES, IFAIL)
      IFAIL = 0
      N = 1
      CALL E02DBF(N, NKNOTX, NKNOTZ, XPT, ZPT, VAL, LAMBDA, MU,
1      VPOINT, NPOINT, CO, NC, IFAIL)
      VZO = (VAL(1) - UZO) / (DELTAX)
      UZO = -VZO / VO ** 2
      ZPT(1) = Y(3)
      IF (IRAY .EQ. 1) THEN
        TVALP = 0
      ELSE
        TVALP = TVAL(ICOUNT - 1)
      END IF
      TVAL(ICOUNT) = H / VO + TVALP
      C
      C CALCULATED INTERPOLATED VELOCITY FIELD AND GRADIENTS NOW FIND
      C DERIVATIVES OF FCNS AND APPLY R-K STEP
      C
      CALL FCN(T, Y, W(1,1))
      CALL D02YAF(X, H, NVAR, Y, FCN, W, NVAR, IW2)
      IF (Y(1) .GT. XMAX .OR. Y(3) .GT. ZMAX .OR. Y(1) .LT. XMIN +
1      HO .OR. Y(3) .LT. ZMIN + HO .OR. (ANG .LT. 0.0 .AND. Y(3)
2      Y(3) .GT. ZVAL(ICOUNT)) .OR. (ANG .GT. 0.0 .AND. Y(3)
3      .LT. ZVAL(ICOUNT))) THEN
        RAY TERMINATED !
        LENRAY = IRAY
        GO TO 90
      END IF
      CONTINUE
      CALL PTOJIN(KCOORD, ZCOORD, 2, LENRAY, 1)
      CONTINUE
      CALL FRAME
      WRITE (7,*) ICOUNT
      PX = 8
      PZ = 7 + NLAYER * 4
      CALL ZERO(40, LAMBDA)
      NADRES = (PX - 7) * (PZ - 7)
      NPOINT = ICOUNT + NADRES
      IFAIL = 0
      C
      C ROUTINE FOR BICUBIC SPLINE CALCULATION
      C
      DO 110 III = 1, ICOUNT
        W1(III) = 1.0
        CONTINUE
110      ICOUNT = ICOUNT + 1
        W1(ICOUNT) = 100.
        XVAL(ICOUNT) = XO
        ZVAL(ICOUNT) = ZO
        TVAL(ICOUNT) = 0.0D0
        WRITE (6,*) 'ICOUNT = ', ICOUNT
        NXEST = 20
        NZEST = 40
        WRITE (6,*) 'NXEST , NZEST =', NXEST, NZEST

```



```
DOUBLE PRECISION V0, UX0, UZ0
COMMON /DATA/ V0, UX0, UZ0
G = UX0 * Y(2) + UZ0 * Y(4)
F(1) = Y(2)
F(2) = V0 * (UX0 - G*Y(2))
F(3) = Y(4)
F(4) = V0 * (UZ0 - G*Y(4))
RETURN
END
```

Appendix A.5

The raytracing subroutine

RAYTRA

The subroutine RAYTRA is another two-point raytracing program. It uses a simple, laterally invariant velocity model, resulting in very fast raytracing. The subroutine is called from the Kirchhoff migration program in appendix A.8 and returns the travel time between any two points in the velocity model.

```

C
C SUBROUTINE TO RAYTRACE FROM POINT (XSTART,ZSTART) TO (XEND,ZEND)
C IN 2-DIMENSIONAL VELOCITY FIELD DEFINED BY :
C DEPTH(layer top), VV(layer), VH (layer)
C Variables: ACC defines raytracing accuracy in metres
C LAYIMG = layer of starting point
C THETA0 = takeoff angle  $(-\pi/2 \rightarrow 3\pi/2)$  0.0=horizontal to right
C THETA1 = Raypath angle at (XEND,ZEND)
C TAU0 = travel time
C NLAMAX = total no. of layers there are (NLAMAX+1) depths
C DZGRID = Vertical grid size used by migration algorithm
C MAXLOP = Maximum no. of searches for raypath
C VH,VV specify horizontal and vertical velocity components in
C each layer (elliptical approxn to transverse isotropy)
C No lateral velocity variations
C Written by M J Findlay 1990
C
C SUBROUTINE RAYTRA(NLAMAX, NDEPTH, DEPTH, VH, VV, XSTART, ZSTART,
1 XEND, ZEND, DZGRID, ACC, MAXLOP, THETA0, THETA1, TAU0,
2 THETA1, NRAV)
REAL DEPTH(NDEPTH), VH(NLAMAX), VV(NLAMAX), XINTS(31), DIST(31)
REAL TIME(31), THET(31)
DOUBLE PRECISION TEMA, TEMB, TEMC, TEMCC, TEMD, TEME, TEMDD
LOGICAL RECLAY
PI = 3.1415926535
PI2 = PI / 2.
NRAV = 1
C
C CHECK FOR ZERO - LENGTH RAYPATH
C
C IF (XSTART.EQ. XEND .AND. ZSTART.EQ. ZEND) THEN
C TAU0 = 0.0
C GO TO 150
C END IF
C DETERMINE IMAGE POINT LAYER
C
C DO 10 NL = 1, NLAMAX
C IF (DEPTH(NL + 1) .GT. ZSTART - DZGRID/100.) THEN
C LAYIMG = NL
C GO TO 20
C END IF
C 10 CONTINUE
C 20 CONTINUE
C DETERMINE END POINT LAYER
C
C DO 30 NL = 1, NLAMAX
C IF (DEPTH(NL + 1) .GT. ZEND - DZGRID/100.) THEN
C LAYSOR = NL
C GO TO 40
C END IF
C 30 CONTINUE
C 40 CONTINUE
C RAYTRACE FROM IMAGEPOINT TO SOURCE / RECEIVER
C
C IF (ABS(DEPTH(LAYIMG + 1) - ZSTART) .LT. DZGRID/100.)

```

```

1 THEN
C IF (ZEND .LT. ZSTART) THEN
C ZSTART = ZSTART - DZGRID / 100.
C ELSE
C ZSTART = ZSTART + DZGRID / 100.
C LAYIMG = LAYIMG + 1
C END IF
C IF (ABS(DEPTH(LAYSOR + 1) - ZEND) .LT. DZGRID/100.) THEN
C IF (ZEND .LT. ZSTART) THEN
C ZEND = ZEND + DZGRID / 100.
C ELSE
C ZEND = ZEND - DZGRID / 100.
C LAYSOR = LAYSOR + 1
C END IF
C 50 CONTINUE
C
C TRACING FROM (XSTART,ZSTART) in layer LAYIMG TO (XEND,ZEND) in
C LAYSOR(isot)
C
C IF (THETA0 .LT. 2.*PI) THEN
C THETA0 = THETA0
C ELSE IF (XEND.EQ. XSTART) THEN
C THETA0 = PI2
C IF (ZEND .LT. ZSTART) THETA0 = PI + PI2
C ELSE
C THETA0 = (ZEND - ZSTART) / (XEND - XSTART)
C THETA0 = ATAN(THETA0)
C IF (XEND .LT. XSTART) THETA0 = THETA0 + PI
C END IF
C IF (XEND - XSTART .LT. - DZGRID/100.) THEN
C THETAB = PI2 - .01
C THETAT = PI + PI2 + .01
C ELSE IF (XEND - XSTART .GT. DZGRID/100.) THEN
C THETAB = PI2 + .01
C THETAT = -PI2 - .01
C ELSE
C SPECIAL CASE FOR VERTICAL RAYPATHS
C
C IF (LAYSOR.EQ. LAYIMG) THEN
C DO = ABS(ZEND - ZSTART)
C TAU0 = DO / VV(LAYSOR)
C ELSE IF (THETA0.EQ. PI2) THEN
C DOWN
C DO = DEPTH(LAYIMG + 1) - ZSTART
C TAU0 = DO / VV(LAYIMG)
C DO 60 NL = LAYIMG + 1, LAYSOR - 1
C DO = DEPTH(NL + 1) - DEPTH(NL)
C T0 = DO / VV(NL)
C TAU0 = TAU0 + T0
C 60 CONTINUE
C DO = ZEND - DEPTH(LAYSOR)
C TAU0 = TAU0 + DO / VV(LAYSOR)
C ELSE
C UP
C DO = ZSTART - DEPTH(LAYIMG)

```

```

          TAU0 = D0 / VV(LAYING)
DO 70 NL = LAYING - 1, LAYSOR + 1, -1
  DO = DEPTH(NL + 1) - DEPTH(NL)
  T0 = D0 / VV(NL)
  TAU0 = TAU0 + T0
CONTINUE
DO = DEPTH(LAYSOR + 1) - ZEND
TAU0 = TAU0 + D0 / VV(LAYSOR)
END IF
  THETA = THETA0
GO TO 150
END IF
ZINTB = 1E9
ZINTT = -1E9
DTHETA = 0.05
TAU0 = 0.0

C
C NOW START RAYTRACING
DO 140 I = 1, MAXIOP
  NRAY = I
  CALL ZERO(31, TIME)
  RECLAY = .FALSE.
  CALL ZERO(31, DIST)
  DO 80 JJJ = 1, 31
    XINTS(JJJ) = -1000.
80  CONTINUE
    IF (THETA0 .GT. 3.*PI2 .OR. THETA0 .LT. - PI2) THEN
      IF (THETA0 .GT. 3.*PI2) THETA0 = THETA0 - DTHETA
      IF (THETA0 .LT. - PI2) THETA0 = THETA0 + DTHETA
      GO TO 140
    END IF
    XPO = XSTART
    ZPO = ZSTART
    THETA = THETA0
    NLAY = LAYING
    CONTINUE
    IF (ABS(THETA) .LT. 1.E-3 .OR. ABS(THETA - PI) .LT. 1.E-3)
      THEN
        THETA = THETA + 2.E-3
      END IF
      IF (NLAY .EQ. LAYSOR) RECLAY = .TRUE.
      IF (THETA .LT. 0.0 .OR. THETA .GT. PI) THEN
        DZ = DEPTH(NLAY) - ZPO
        ZPO = DEPTH(NLAY)
      ELSE
        DZ = DEPTH(NLAY + 1) - ZPO
        ZPO = DEPTH(NLAY + 1)
      END IF
      IF (ABS(THETA) .EQ. PI2 .OR. THETA .EQ. 3*PI2) THEN
        DX = 0.0
      ELSE
        DX = DZ / TAN(THETA)
      END IF
      DIST(NLAY) = SQRT(DX**2 + DZ**2)
      THET(NLAY) = THETA
      TEMP1 = VV(NLAY) ** 2
      TEMP2 = VH(NLAY) ** 2
      TEMP3 = TEMP1 * COS(THETA) ** 2
      TEMP4 = TEMP2 * SIN(THETA) ** 2

          VTHETA = (TEMP1*TEMP2/(TEMP3 + TEMP4))
          IF (VTHETA .LT. 0.0) WRITE (6,*) '108', TEMP1, TEMP2, TEMP3,
            TEMP4
          VTHETA = SORT(VTHETA)
          TIME(NLAY) = DIST(NLAY) / VTHETA
          XPO = XPO + DX
          IF (THETA .LT. 0.0 .OR. THETA .GT. PI) THEN
            XINTS(NLAY) = XPO
          ELSE
            XINTS(NLAY + 1) = XPO
          END IF
1      CONTINUE
          C FIRST CHECK FOR RAY WITHIN CAPTURE RADIUS
          C
          ERR = SORT((XPO - XEND)**2 + (ZPO - ZEND)**2)
          IF (ERR .LE. ACC) THEN
            C RAY FOUND
            TAU0 = 0.0
            DO 100 NL = 1, NLMAX
              TAU0 = TAU0 + TIME(NL)
            CONTINUE
            GO TO 150
          END IF
          C CHECK FOR RAY PASSING BEYOND ENDPOINT
          C
          IF ((THETA .GT. PI2 .AND. XPO .LT. XEND) .OR. (THETA .LE.
            1      PI2 .AND. XPO .GT. XEND)) THEN
            IF (THETA .GT. PI2) THEN
              WRITE (6,*) 'RAY OUT L.H. SIDE '
            ELSE
              WRITE (6,*) 'RAY OUT R.H. SIDE '
            END IF
          C
          C FIND INTERSECTION POINT
          C
          IF (DX .EQ. 0.0) THEN
            ZINT = ZPO
            DXDASH = 0.0
            DZDASH = ZEND - (ZPO - DZ)
          ELSE
            XOLD = XPO - DX
            ZOLD = ZPO - DZ
            DXDASH = XEND - XOLD
            DZDASH = DZ * DXDASH / DX
            ZINT = ZOLD + DZDASH
          END IF
          C CHECK TO SEE IF CLOSE TO SOURCE POSITION
          C
          IF (ABS(ZINT - ZEND) .LT. ACC) THEN
            C RAY FOUND
            DIST(NLAY) = SQRT(DXDASH**2 + DZDASH**2)
            THET(NLAY) = THETA
            TEMP1 = VV(NLAY) ** 2
            TEMP2 = VH(NLAY) ** 2
            TEMP3 = TEMP1 * COS(THETA) ** 2
            TEMP4 = TEMP2 * SIN(THETA) ** 2

```

```

C      VTHETA = (TEMP1*TEMP2/(TEMP3 + TEMP4))
C      IF (VTHETA .LT. 0.0) WRITE (6,*)'152', TEMP1, TEMP2,
C          TEMP3, TEMP4
C      VTHETA = SORT(VTHETA)
C      TIME(NLAY) = DIST(NLAY) / VTHETA
C      TA0 = 0
C      DO 110 NL = 1, NLMAX
C          TAU0 = TAU0 + TIME(NL)
C      CONTINUE
110
C RAY FOUND SO RETURN TO MAIN PROGRAM
C
C GO TO 150
C
C NEED TO STORE THETA0 FOR INTEGRAL KERNEL
C
C ELSE
C
C NEXT RAY (RAY NOT FOUND)
C     IF (ZINT .GT. ZINTT .AND. ZINT .LT. ZEND)
C         THEN
C             THETAT = THETA0
C             ZINTT = ZINT
C         ELSE IF (ZINT .LT. ZINTTB .AND. ZINT .GT. ZEND)
C             THEN
C                 THETAB = THETA0
C                 ZINTB = ZINT
C             END IF
C             THETA0 = .5 * (THETAT + THETAB)
C             GO TO 140
C         ELSE IF
C             THETA0 = .5 * (THETAT + THETAB)
C             GO TO 140
C         END IF
C
C CHECK FOR RAY OUT TOP OR BOTTOM & CALCULATE NEW INTERCEPTION POINTS
C
C ELSE IF (NLAY .EQ. 1 .AND. (THETA .LT. 0.0 .OR. THETA .GT. PI)
C     ) THEN
C     ) THEN
C         DXNEW = XEND - XPO
C         IF (ABS(THETA) .EQ. PIZ) THEN
C             ZINT = -1.E9
C         ELSE
C             ZINT = ZPO - ABS(DXNEW*TAN(THETA))
C         END IF
C         IF (ZINT .GT. ZINTT .AND. ZINT .LT. ZEND) THEN
C             THETAT = THETA0
C             ZINTT = ZINT
C         ELSE IF (ZINT .LT. ZINTTB .AND. ZINT .GT. ZEND)
C             THEN
C                 THETAB = THETA0
C                 ZINTB = ZINT
C             END IF
C             THETA0 = .5 * (THETAT + THETAB)
C             GO TO 140
C         ELSE IF (NLAY .EQ. NLMAX .AND. (THETA .GT. 0.0 .AND. THETA .
C             LT. PI)) THEN
C             L.T. PI)) THEN
C
C RAY OUT BOTOM
C     DXNEW = XEND - XPO
C     IF (ABS(THETA) .EQ. PIZ) THEN
C         ZINT = 1.E9
C     ELSE
C         ZINT = 1.E9
C     END IF
C
C RAY NOT FOUND AND STILL IN GRID SO APPLY SNEEL'S LAW AND CHECK FOR
C SUPERCRITICAL RAYS
C
C V0 = VH(NLAY)
C V00 = VV(NLAY)
C IF (ABS(THETA) .EQ. PIZ .OR. THETA .EQ. 3*PI/2) THEN
C     NLAY = NLAY + 1
C     IF (THETA .LT. 0.0 .OR. THETA .GT. PI) NLAY = NLAY - 2
C     GO TO 130
C END IF
C IF (THETA .LT. 0.0 .OR. THETA .GT. PI) THEN
C     V1 = VH(NLAY - 1)
C ELSE
C     V1 = VH(NLAY + 1)
C END IF
C IF (V0 .GE. V1) GO TO 120
C IF (V0 .LE. V1) GO TO 120
C TEM = V0 ** 2 * (V1**2 - V0**2) / V0 ** 4
C IF (TEM .LT. 0.0) WRITE (6,*)'235', V00, V1, VO
C CRIT = ATAN(SORT(1./TEM))
C CRIT = PI/2 - CRIT
C IF (ABS(THETA) .LT. CRIT .OR. ABS(THETA - PI) .LT. CRIT)
C     THEN
C         IF (THETA .GT. PI) THEN
C             IF (ZPO .LT. ZEND) THEN
C                 THETAT = THETA0
C                 ZINTT = ZPO
C             ELSE
C                 THETAB = THETA0
C                 ZINTB = ZPO
C             END IF
C             THETA0 = .5 * (THETAT + THETAB)
C             ZINT = ZINTT
C         ELSE IF (THETA .GT. PI) THEN
C             IF (ZPO .LT. ZEND) THEN
C                 THETAT = THETA0
C                 ZINTT = ZPO
C             ELSE
C                 THETAB = THETA0
C                 ZINTB = ZPO
C             END IF
C             THETA0 = .5 * (THETAT + THETAB)
C             ZINT = ZINTT
C         ELSE IF (ZPO .LT. ZEND) THEN
C             THETAT = THETA0
C             ZINTT = ZPO
C         ELSE IF (ZPO .GT. ZEND) THEN
C             THETAB = THETA0
C             ZINTB = ZPO
C         END IF
C         THETA0 = .5 * (THETAT + THETAB)
C         ZINT = ZINTT
C     ELSE IF (THETA .GT. 0.0) THEN
C         IF (ZPO .LT. ZEND) THEN
C             THETAT = THETA0
C             ZINTT = ZPO
C         ELSE IF (ZPO .GT. ZEND) THEN
C             THETAB = THETA0
C             ZINTB = ZPO
C         END IF
C         THETA0 = .5 * (THETAT + THETAB)
C         ZINT = ZINTT
C     ELSE IF (THETA .GT. -PI/2) THEN
C         IF (ZPO .LT. ZEND) THEN
C             THETAT = THETA0
C             ZINTT = ZPO
C         ELSE IF (ZPO .GT. ZEND) THEN
C             THETAB = THETA0
C             ZINTB = ZPO
C         END IF
C         THETA0 = .5 * (THETAT + THETAB)
C         ZINT = ZINTT
C     END IF

```

```

      THETA0 = THETA0 - DTHETA
    END IF
    GO TO 140
  C RAY IS SUPERCRITICAL SO JUMP OUT AND TRY NEXT RAY
  C
    ELSE
  C APPLY SNELL'S LAW
  120 NLAY0 = NLAY
      NLAY = NLAY + 1
      IF (THETA.LT. 0.0 .OR. THETA.GT. PI) NLAY = NLAY - 2
      TEMA = VH(NLAY) ** 4 * VV(NLAY0) ** 2
      TEMB = VV(NLAY) * VH(NLAY0)
      TEMCC = DBLE(PI2 - THETA)
      TEMCC = TAN(TEMCC)
      TEMC = DBLE(VH(NLAY0)) / TEMCC
      TEMC = TEMC ** 2
      TEMD = VH(NLAY0) ** 2 - VH(NLAY) ** 2
      TEMD = VV(NLAY0) / VH(NLAY0)
      TEMD = TEMD ** 2
      ALFAC = TEMC + TEMD * TEMD
      IF (ALFAC.LT. 0.0) WRITE (6,*) '281', VH(NLAY0), VH(NLAY),
        VV(NLAY0), VV(NLAY), THETA, TEMA, TEMB, TEMC, TEMCC,
        TEMD, TEMD, TEME
      TEME = TEMB * SQRT(TEMC + TEMD*TEMD)
      THETA2 = ATAN(SQRT(TEMA)/TEME)
      IF (THETA.GT. PI) THEN
        THETA2 = 3 * PI2 - THETA2
      ELSE IF (THETA.GT. PI/2) THEN
        THETA2 = PI2 + THETA2
      ELSE IF (THETA.GT. 0.0) THEN
        THETA2 = PI2 - THETA2
      ELSE
        THETA2 = THETA2 - PI2
      END IF
      THETA = THETA2
  130 CONTINUE
      GO TO 90
    END IF
  C NEXT RAY TO THIS SOURCE
  140 CONTINUE
  C START RAYTRACING TO RECEIVERS
  150 CONTINUE
      THETA1 = THETA
  C set theta1 to final ray angle at XEND,ZEND
  C GO BACK TO MAIN PROG
      RETURN
    END

```

Appendix A.6

The finite-difference migration program

EXTRAPREV

The program EXTRAPREV carries out finite-difference migration in the (x,z,t) domain. A file of imaging times generated for each grid point by raytracing is attached to unit 11 at run time. The wavefield may be plotted at particular time steps as it regresses from the receivers, and the final migrated section is plotted.

```

C
C PROGRAM TO CARRY OUT ACOUSTIC REVERSE TIME EXCITATION IMAGING
C
C CONDITION FINITE DIFFERENCE MIGRATION.
C
C ALGORITHM PROVIDED BY MCMCHAN (GEOPHYSICS V50 P627-636)
C
C (ABSORBING BOUNDARY CONDITIONS FROM RENAUTE-PETERSEN GEOPHYSICS 1989)
C
C CAN HANDLE LATERAL VELOCITY VARIATIONS
C
C REQUIRES USE OF SUBROUTINE LIBRARY TSAR4.L, *GHOST80
C
C AND PRIOR DETERMINATION OF EXCITATION TIMES BY RAYTRACING PROG.
C
C (TRAVEL TIMES ATTACHED TO UNIT 11 AT RUN TIME)
C
C WRITTEN BY M J FINDLAY 1989
C
C
C PROGRAM EXTRAP
C
C PARAMETER (NHOR=140, NVER=140, NTIM=3, NSAM=1024, NCHN=95)
C
C PARAMETER (PI=3.1415926535898, POL0=21.*PI/64., POL2=-15.*PI/64.)
C
C
C POL0, POL2 DEFINE POLYNOMIAL COEFFICIENTS FOR ABSORBING BOUNDARIES
C
C SEE RENAUTE-PETERSEN, GEOPHYSICS 1989
C
C
C IMPLICIT REAL(A - H, O - Z)
C
C REAL U(NHOR, NVER, NTIM), TERM1, TERM2, TERM3, A
C
C REAL V(NHOR, NVER), TRACES(NSAM, NCHN)
C
C REAL TT(NSAM), AMPMAX, XMAX, ZMAX, AMPSC1, RREC, TMAX
C
C REAL UR4(NVER, NHOR), RPLOT(NSAM), Z(NVER), XCORS(NCHN),
C
C 1 ZCORS(NCHN)
C
C REAL REFRS(NVER, NHOR), IMT(NHOR, NVER), ARR(128), RMT(NHOR)
C
C DOUBLE PRECISION X(256), X1(256), Z1(256), C(256), XEVAL,
C
C 1 XPRIME(256)
C
C DOUBLE PRECISION ZPRIME(256)
C
C CHARACTER*1 ALPDM(NVER*4)
C
C COMPLEX CX(2048), CX1(256)
C
C INTEGER NMOV(20), IXS(NCHN), IZS(NCHN), NCOL(9)
C
C CHARACTER*17 OPFILE, IPFILE, IFNAM1, IFNAM2, IFNAM3
C
C DATA NCOL /1, 7, 4, 6, 5, 3, 8, 2, 1/
C
C WRITE (6,*) PI
C
C ITO = 3
C
C NSAMSP = NHOR * NVER
C
C CALL ZERO(NSAMSP, REFRS(1,1))
C
C NSAMT = NHOR * NVER * NTIM
C
C CALL ZERO(NSAMT, U(1,1,1))
C
C DO 10 I = 1, NVER * 4
C
C ALPDM(I) = '0'
C
C 10 CONTINUE
C
C THIS ZEROS ARRAY U(X,Z,T=0,1,2)
C
C
C WRITE (6,*) 'ENTER GRID POINT SPACING (METRES): '
C
C READ (5,*) H
C
C WRITE (6,*) 'ENTER MAXIMUM GRID VALUES IN X & Z (<128) : '
C
C READ (5,*) IXMAX, IZMAX
C
C WRITE (6,*) 'ENTER TIME SAMPLE INTERVAL (millssecs): '
C
C READ (5,*) DT
C
C DT = DT * 1.E-3
C
C WRITE (6,*) 'LOCATION OF REAL SOURCE : '
C
C READ (5,*) IX0, IZ0
C
C WRITE (6,*) 'ENTER DELAY TIME OF SOURCE IN MILLISECS : '
C
C READ (5,*) DELAYT
C
C DELAYT = DELAYT * 1.E-3
C
C
C WRITE (6,*) 'Enter number of "source" positions : '
C
C READ (5,*) NSOR
C
C DO 20 II = 1, NSOR
C
C WRITE (6,*) 'Enter coords (X,Z) of "source" #', II, ' : '
C
C READ (5,*) XCORS(II), ZCORS(II)
C
C 20 CONTINUE
C
C WRITE (6,*) 'ENTER NAME OF FILE CONTAINING SEISMOGRAMS '
C
C READ (5,190) IFNAM2
C
C LEN = 4104
C
C WRITE (6,*) 'Enter shot number to be restored in file: '
C
C READ (5,*) NSHOT
C
C IDSHOT = (NSHOT - 1) * (NSOR + 1) + 1
C
C OPEN (12, FILE=IFNAM2, STATUS='UNKNOWN', FORM='UNFORMATTED', ACCESS=
C
C 1 'DIRECT', RECL=LEN)
C
C DO 30 J = 1, NSOR
C
C NIDC = IDSHOT + J
C
C READ (12, REC=NIDC) (TRACES(I,J), I=1, NSAM)
C
C 30 CONTINUE
C
C CLOSE (12)
C
C LENTAP = NINT (REAL(NSOR)/5.0) * 2
C
C DO 50 J = 1, LENTAP / 2
C
C FACTOR = .5 - COS(2.*PI*REAL(J)/REAL(LENTAP)) / 2.
C
C DO 40 I = 1, NSAM
C
C TRACES(I,J) = TRACES(I,J) * FACTOR
C
C TRACES(I, NSOR - J + 1) = TRACES(I, NSOR - J + 1) * FACTOR
C
C 40 CONTINUE
C
C 50 CONTINUE
C
C WRITE (6,*) 'REMOVE T**2 RAMP 0=NO 1=YES (AND APPLY T**5) : '
C
C READ (5,*) NOYST2
C
C IF (NOYST2.EQ. 1) THEN
C
C DO 70 I = 1, NSAM
C
C FACTOR = 1. / (REAL(I)) ** 1.5
C
C DO 60 J = 1, NSOR
C
C TRACES(I,J) = TRACES(I,J) * FACTOR
C
C 60 CONTINUE
C
C 70 CONTINUE
C
C END IF
C
C WRITE (6,*) 'INTERPOLATE SPATIALLY BY POWER OF 2 (ENTER POWER) : '
C
C READ (5,*) NPOWER
C
C RNP = REAL(2**NPOWER)
C
C IF (NPOWER.EQ. 0) GO TO 180
C
C NSAMFT = NSOR
C
C CALL POWER2(NSAMFT, LX0)
C
C NSAMIT = NSAMFT * 2.0 ** NPOWER
C
C CALL POWER2(NSAMIT, LX1)
C
C LRAMP = LX0 - NSAMFT
C
C DO 130 JI = 1, NSAM
C
C CALL ZERO(512, CX)
C
C RAMP0 = TRACES(JI, NSAMFT)
C
C DO 80 II = 1, NSAMFT
C
C CX(II) = CMPLX(RAMP0*FAC)
C
C DO 90 II = NSAMFT + 1, LX0
C
C FAC = REAL(LX0 - II) / REAL(LRAMP)
C
C CX(II) = CMPLX(RAMP0*FAC)
C
C CONTINUE
C
C CALL FORK(LX0, CX, -1.0)
C
C DO 100 II = LX0 / 2 + 2, LX1
C
C CX(II) = 0.0
C
C CONTINUE
C
C 100

```

```

110 DO 110 II = LX1, LX1 / 2 + 2, -1
      CX(II) = CONJG(CX(LX1 + 2 - II))
      CONTINUE
      CALL FORK(LX1, CX, 1.0)
      DO 120 II = 1, NSAMT - 1
            TRACES(JJ,II) = REAL(CX(II)) * SORT(RNP)
120    CONTINUE
130  CONTINUE
C
C NOW DO INTERPOLATION OF COORDINATE VALUES
C
      DXINTF = 1.0
      DXINTL = DXINTF / (RNP)
      NFINAL = NSOR * 2 ** NPOWER - (2*NPOWER - 1)
      N1 = NSOR
      N2 = (N1 - 1) * N1 / 2
      NO = N1 - 1
      WRITE (6,*) 'DXINTL = ', DXINTL
      WRITE (6,*) 'NFINAL = ', NFINAL
      DO 160 II = 1, NFINAL
            DO 140 III = 1, NSOR
                  X(III) = DBLE(III)
                  X1(III) = XCORS(III)
                  Z1(III) = ZCORS(III)
140    CONTINUE
            XEVAL = X(1) + DBLE(DXINTL*(II - 1))
            CALL E01AAF(X, X1, C, N1, N2, NO, XEVAL)
            XPRIME(II) = C(N2)
            DO 150 III = 1, NSOR
                  X(III) = DBLE(III)
150    CONTINUE
            CALL E01AAF(X, Z1, C, N1, N2, NO, XEVAL)
            ZPRIME(II) = C(N2)
160  CONTINUE
            DO 170 II = 1, NFINAL
                  XCORS(II) = XPRIME(II)
                  ZCORS(II) = ZPRIME(II)
170  CONTINUE
            NSOR = 2 ** NPOWER * NSOR - (2*NPOWER - 1)
180  CONTINUE
190  FORMAT (A17)
      WRITE (6,*) 'NSOR = ', NSOR
      IXSMAX = 0
      DO 200 II = 1, NSOR
            IXS(II) = NINT(XCORS(II)/H)
            IZS(II) = NINT(ZCORS(II)/H)
            IF (IXS(II).GT. IXSMAX) IXSMAX = IXS(II)
200  CONTINUE
      WRITE (6,210) (II, IXS(II), IZS(II), II=1, NSOR)
210  FORMAT (5(I3,2X,I3,I3,I3,4X))
      WRITE (6,*) 'Enter first time sample to start from : '
      READ (5,*) NSTART
      WRITE (6,*) 'INTERPOLATE TEMPORALLY (ENTER POWER OF 2): '
      READ (5,*) NPOWER
      RNP = REAL(2**NPOWER)
      IF (NPOWER.EQ. 0) GO TO 280
      NSAMFT = NSTART
      CALL POWER2(NSAMFT, LX0)
      NSAMIT = NSAMFT * 2.0 ** NPOWER
      CALL POWER2(NSAMIT, LX1)

LAMP = LX0 - NSAMFT
DO 270 JJ = 1, NSOR
      CALL ZERO(2048, CX)
      RAMP0 = TRACES(NSAMFT, JJ)
      DO 220 II = 1, NSAMFT
            CX(II) = CMPLX(TRACES(II, JJ), 0.0)
220    CONTINUE
      DO 230 II = NSAMFT + 1, LX0
            FAC = REAL(LX0 - II) / REAL(LAMP)
            CX(II) = CMPLX(RAMP0 * FAC)
230    CONTINUE
      CALL FORK(LX0, CX, -1.0)
      DO 240 II = LX0 / 2 + 2, LX1
            CX(II) = 0.0
240    CONTINUE
      DO 250 II = LX1, LX1 / 2 + 2, -1
            CX(II) = CONJG(CX(LX1 + 2 - II))
250    CONTINUE
      CALL FORK(LX1, CX, 1.0)
      DO 260 II = 1, NSAMIT - 1
            TRACES(II, JJ) = REAL(CX(II)) * SORT(RNP)
260    CONTINUE
270  CONTINUE
      DT = DT / RNP
      NSAM0 = NSAM * 2 ** NPOWER
      NSTART = NSTART * 2 ** NPOWER
280  CONTINUE
      NSAMP0 = 2
      LENTIM = NINT(REAL(NSTART)/10.0) * 2
290  FORMAT (8(F10.4))
      DO 310 J = 1, NSOR
            DO 300 I = 1, LENTIM / 2
                  FAC = .5 - COS(2.*PI*REAL(I)/REAL(LENTIM)) / 2.
                  TRACES(NSTART - I + 1, J) = TRACES(NSTART - I + 1, J) * FAC
300    CONTINUE
310  CONTINUE
      WRITE (6,*) 'ENTER NO. OF LAYERS : '
      READ (5,*) NLAYER
      IDEPTH = 0
      VMAX = 0.0
      DO 330 NL = 1, NLAYER
            WRITE (6,*) 'ENTER VELOCITY OF MEDIUM ', NL, ' (metres / sec)'
            READ (5,*) VL
            IF (VL.GT. VMAX) VMAX = VL
            WRITE (6,*) 'ENTER THICKNESS OF LAYER ', NL, ' (metres)'
            READ (5,*) THICK
            NTHICK = NINT(THICK/H)
            DO 320 IZ = 1, NTHICK
                  DO 320 IX = 1, IXMAX
                        V(IX, IZ + IDEPTH) = VL
320    CONTINUE
            IDEPTH = IDEPTH + NTHICK
330  CONTINUE
      IXP = IX0
      IZP = IZ0
      WRITE (6,*) 'INCLUDE POINT DIFFRACTOR 1-Y 0=N:'
      READ (5,*) IYNANS
      IF (IYNANS.EQ. 1) THEN
            WRITE (6,*) 'ENTER IX, IZ, VEL(IX, IZ)'
            READ (5,*) IXP, IZP, VPOINT

```

```

V(IXP,IZP) = VPOINT
END IF
WRITE (6,*)'ENTER NUMBER OF DIFFERENT MOVIE FRAMES REQUIRED'
READ (5,*) NMOVIE
DO 340 NM = 1, NMOVIE
  WRITE (6,*)'ENTER SAMPLE TIME #', NM
  READ (5,*) NMOV(NM)
340 CONTINUE
  WRITE (6,*)'ENTER CUTOFF RATIO FOR S/N : '
  READ (5,*) CUTO
  WRITE (6,*)'2nd or 4th Order finite difference approximation : '
  READ (5,*) NORD
350 FORMAT (A10)
  WRITE (6,*)'ENTER WAVEFIELD PLOT TYPE (1=WIGGLE 2=PIXEL(COLOUR)) : '
  1,
  READ (5,*) IFTYPE
  WRITE (6,*)'ENTER OUTPUT FILE NAME : '
  READ (5,190) IFNAM3
C
C DEFINE COLOURS FOR GHOST80 OUTPUT
C
  CALL PAPER(1)
C
  H=GRID POINT SPACING
C
C NOW DEFINE INITIAL WAVEFIELD INCLUDING SOURCE TERM
C
  360 IT = NSTART
  DO 370 IT = 1, NSOR
    U(IXS(II), IZS(II), IT0 - 1) = TRACES(NSTART, II)
C
C START OF SOURCE AT T = 0
C
    A = (V(IXS(II), IZS(II)) * DT / H)
    IF (DT .GE. H/(SQRT(2.) * V(IXS(II), IZS(II)))) THEN
      WRITE (6,*)'ALGORITHM UNSTABLE'
      WRITE (6,*)'DT, H, V = ', DT, H, V(IXS(II), IZS(II))
      STOP
    END IF
370 CONTINUE
C
C SET UP IMAGE TIME CONDITION
C
  DO 390 JF = 1, NVER
    READ (11,400) (RMT(II), II=1, NHOR)
    DO 380 II = 1, NHOR
      IMT(II, JF) = NINT(RMT(II) / DT)
      IMT(II, JF) = IMT(II, JF) + NINT(IDELAY / DT) + 1
380 CONTINUE
390 CONTINUE
400 FORMAT (8F10.6)
C
C SETUP PLOTTING ARRAY Z(NZ)
C
  DO 410 II = 1, IZMAX
    Z(II) = REAL(II) * H
410 CONTINUE
C
C START LOOP OVER ALL TIME SAMPLES

```

```

420 NPL0T = 1
DO 560 IT = NSTART - 1, NMOV(NMOVIE), -1
  WRITE (6,*)'TIME SAMPLE : ', IT
  TIMEIT = (IT - 1) * DT
  DO 430 IZ = 2, IZMAX - 1
C
C INCLUDE FREE SURFACE BY MAKING U(IX, Z=0) = 0.0 ???
C
    DO 430 IX = 2, IXMAX - 1
      A = (V(IX, IZ) * DT / H)
      IF (NORD .EQ. 2 .OR. IX .EQ. 2 .OR. IZ .EQ. 1 .OR.
        IX .EQ. IXMAX - 1 .OR. IZ .EQ. IZMAX - 1)
        THEN
          TERM1 = 2 * (1 - 2 * A ** 2) * U(IX, IZ, IT0 - 1)
          TERM2 = U(IX, IZ, IT0 - 2)
          TERM3 = U(IX + 1, IZ, IT0 - 1) + U(IX - 1, IZ, IT0 - 1)
          IF (IZ .GT. 1) TERM3 = TERM3 + U(IX, IZ - 1, IT0 - 1)
          IF (IZ .GT. 1) TERM3 = TERM3 + U(IX, IZ - 1, IT0 - 1)
          TERM3 = TERM3 * A ** 2
          U(IX, IZ, IT0) = TERM1 - TERM2 + TERM3
        ELSE
          TERM1 = (2. - 5. * A ** 2) * U(IX, IZ, IT0 - 1)
          TERM1 = (2. - 10. * A ** 2 / 3.) * U(IX, IZ, IT0 - 1)
          TERM2 = U(IX + 1, IZ, IT0 - 1) + U(IX, IZ + 1, IT0 - 1)
          IF (IZ .GT. 1) TERM2 = TERM2 + U(IX, IZ - 1, IT0 - 1)
          IF (IZ .GT. 1) TERM2 = TERM2 + U(IX, IZ - 1, IT0 - 1)
          TERM3 = U(IX + 2, IZ, IT0 - 1) + U(IX, IZ + 2, IT0 - 1)
          IF (IZ .GT. 2) THEN
            TERM3 = TERM3 + U(IX, IZ - 2, IT0 - 1)
            TERM3 = TERM3 + U(IX, IZ - 2, IT0 - 1)
          END IF
          ALPHA = TERM1 + A ** 2 * (16 * TERM2 - TERM3) / 12.
          U(IX, IZ, IT0) = ALPHA - U(IX, IZ, IT0 - 2)
          IF (IT .EQ. IMT(IX, IZ)) THEN
            WRITE (6,*) IX, IZ, IMT(IX, IZ)
            REFRES(IZ, IX) = U(IX, IZ, IT0)
          END IF
        END IF
      END IF
430 CONTINUE
C
C NOW APPLY ABSORBING BOUNDARY CONDITIONS
C
C OPTION TO MAKE SURFACE BOUNDARY ABSORBING
C
    DO 440 IX = 3, IXMAX - 2
      IZ = 1
      A = (V(IX, IZ) * DT / H)
      TERM1 = U(IX, IZ, IT0 - 1) + U(IX, IZ + 1, IT0 - 1)
      TERM1 = TERM1 * 2.0 * POLO
      TERM2 = U(IX, IZ + 1, IT0) + U(IX, IZ, IT0 - 2)
      TERM2 = TERM2 * (A - POLO + 2. * POLO2 * A ** 2)
      TERM3 = U(IX + 1, IZ + 1, IT0) + U(IX - 1, IZ + 1, IT0) + U(
        IX + 1, IZ, IT0 - 2) + U(IX - 1, IZ, IT0 - 2)
      TERM3 = -TERM3 * A ** 2 * POLO2
      UINIT = (TERM1 + TERM2 + TERM3) / (A + POLO)
      U(IX, IZ, IT0) = UINIT - U(IX, IZ + 1, IT0 - 2)
440 CONTINUE
C
C LHS FIRST

```

```

C      DO 450 IZ = 3, IZMAX - 2
C          IX = 1
C              A = V(IX,IZ) * DT / H
C              TERM1 = U(IX,IZ,ITO - 1) + U(IX + 1,IZ,ITO - 1)
C              TERM2 = TERM1 * 2.0 * POLO
C              TERM3 = U(IX + 1,IZ,ITO) + U(IX,IZ,ITO - 2)
C              TERM4 = TERM2 * (A - POLO + 2.*POL2*A**2)
C              TERM5 = U(IX,IZ + 1,ITO - 2) + U(IX,IZ - 1,ITO - 2)
C              TERM6 = -TERM3 * A ** 2 * POL2
C              UNIT = (TERM1 + TERM2 + TERM3) / (POLO + A)
C              U(IX,IZ,ITO) = UNIT - U(IX + 1,IZ,ITO - 2)
C          CONTINUE
C      450
C      NOW BOTTOM
C          DO 460 IX = 3, IXMAX - 2
C              IZ = IZMAX
C              A = V(IX,IZ) * DT / H
C              TERM1 = U(IX,IZ,ITO - 1) + U(IX,IZ - 1,ITO - 1)
C              TERM2 = TERM1 * 2.0 * POLO
C              TERM3 = U(IX,IZ - 1,ITO) + U(IX,IZ,ITO - 2)
C              TERM4 = TERM2 * (A - POLO + 2.*POL2*A**2)
C              TERM5 = U(IX + 1,IZ - 1,ITO) + U(IX - 1,IZ - 1,ITO) + U(
C                  IX + 1,IZ,ITO - 2) + U(IX - 1,IZ,ITO - 2)
C              TERM6 = -TERM3 * A ** 2 * POL2
C              UNIT = (TERM1 + TERM2 + TERM3) / (A + POLO)
C              U(IX,IZ,ITO) = UNIT - U(IX,IZ - 1,ITO - 2)
C          CONTINUE
C      460
C      NOW RHS
C          DO 470 IZ = 3, IZMAX - 2
C              IX = IXMAX
C              A = V(IX,IZ) * DT / H
C              TERM1 = U(IX,IZ,ITO - 1) + U(IX - 1,IZ,ITO - 1)
C              TERM2 = TERM1 * 2.0 * POLO
C              TERM3 = U(IX - 1,IZ,ITO) + U(IX,IZ,ITO - 2)
C              TERM4 = TERM2 * (A - POLO + 2.*POL2*A**2)
C              TERM5 = U(IX,IZ + 1,ITO - 2) + U(IX,IZ - 1,ITO - 2) + U(
C                  IX - 1,IZ + 1,ITO) + U(IX - 1,IZ - 1,ITO)
C              TERM6 = -TERM3 * A ** 2 * POL2
C              UNIT = (TERM1 + TERM2 + TERM3) / (POLO + A)
C              U(IX,IZ,ITO) = UNIT - U(IX - 1,IZ,ITO - 2)
C          IF (ABS(U(IX,IZ,ITO)*TIMEIT .LT. A0*TRF/CUTO))
C              U(IX,IZ,ITO) = 0.0
C      470
C      CONTINUE
C      NOW MAKE BOTTOM CORNERS ABSORBING TOO
C      LOWER RIGHT HAND CORNER
C          IX = IXMAX - 1
C          IZ = IZMAX
C          VDASH = V(IX,IZ)
C          TERM1 = (U(IX,IZ - 1,ITO) + U(IX - 1,IZ,ITO)) / H
C          TERM2 = (U(IX,IZ,ITO - 1)) * SQRT(2.) / (VDASH*DT)
C          U(IX,IZ,ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
C      UPPER RIGHT HAND CORNER
C          IX = IXMAX - 1
C          IZ = 1
C          VDASH = V(IX,IZ)
C          TERM1 = (U(IX,IZ + 1,ITO) + U(IX - 1,IZ,ITO)) / H
C          TERM2 = (U(IX,IZ,ITO - 1)) * SQRT(2.) / (VDASH*DT)
C          U(IX,IZ,ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
C      LOWER LEFT HAND CORNER
C          IX = 1
C          IZ = IZMAX - 1
C          VDASH = V(IX,IZ)
C          TERM1 = (U(IX,IZ - 1,ITO) + U(IX - 1,IZ,ITO)) / H
C          TERM2 = (U(IX,IZ,ITO - 1)) * SQRT(2.) / (VDASH*DT)
C          U(IX,IZ,ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
C          IX = 2
C          IZ = IZMAX
C          VDASH = V(IX,IZ)
C          TERM1 = (U(IX,IZ - 1,ITO) + U(IX + 1,IZ,ITO)) / H
C          TERM2 = (U(IX,IZ,ITO - 1)) * SQRT(2.) / (VDASH*DT)
C          U(IX,IZ,ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
C          IX = 1
C          IZ = 1
C          VDASH = V(IX,IZ)
C          TERM1 = (U(IX,IZ - 1,ITO) + U(IX + 1,IZ,ITO)) / H
C          TERM2 = (U(IX,IZ,ITO - 1)) * SQRT(2.) / (VDASH*DT)
C          U(IX,IZ,ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))

```

```

C UPPER LEFT CORNER
      IX = 1
      IZ = 2
      VDASH = V(IX, IZ)
      TERM1 = (U(IX, IZ + 1, ITO) + U(IX + 1, IZ, ITO)) / H
      TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH*DT)
      U(IX, IZ, ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
      IX = 2
      IZ = 1
      VDASH = V(IX, IZ)
      TERM1 = (U(IX, IZ + 1, ITO) + U(IX + 1, IZ, ITO)) / H
      TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH*DT)
      U(IX, IZ, ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
      U(IX, IZ, ITO) = 0.0
C
      IX = 1
      IZ = 1
      VDASH = V(IX, IZ)
      TERM1 = (U(IX, IZ + 1, ITO) + U(IX + 1, IZ, ITO)) / H
      TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH*DT)
      U(IX, IZ, ITO) = (TERM1 + TERM2) / (2./H + SQRT(2.)/(VDASH*DT))
C
C MAKE THIS LINE THE LAST STEP
C
      DO 480 II = 1, NSOR
      IF (I2S(II) .EQ. IZMAX) GO TO 480
      U(IXS(II), I2S(II), ITO) = TRACES(II, II)
480   CONTINUE
      WRITE (19, *) IXP, IZP, IT, U(IXP, IZP, IT)
      DO 490 J = 1, NVER
      DO 490 I = 1, NHOR
      U(I, J, ITO - 2) = U(I, J, ITO - 1)
      U(I, J, ITO - 1) = U(I, J, ITO)
490   CONTINUE
      IF (IT .LT. NMOV(NMOVIE) + 1) THEN
C WRITE OUT U(IX, IZ, ITO) TO FILE ATTACHED TO UNIT 13
      DO 500 J = 1, NVER
      WRITE (13, 290) (U(I, J, ITO - 1), I=1, NHOR)
500   CONTINUE
      END IF
      IF (IT .EQ. NMOV(NPLOT)) THEN
C GO TO 500
C INCLUDE ABOVE COMMENTED STATEMENT TO SUPPRESS PLOT OUTPUT
510   CONTINUE
C DATA OUT TO UNIT 14
C
      DO 520 J = 1, NVER
      WRITE (14, 290) (U(I, J, ITO - 1), I=1, NHOR)
520   CONTINUE
C
C NOW GET GRAPHICAL OUTPUT OF U(IX, IZ, ITO)
      AMPMAX = 0.0
      DO 530 I = 1, IXMAX
      DO 530 J = 1, IZMAX
      U(4(J, I)) = REAL(U(I, J, ITO - 1))
      IF (ABS(U(4(J, I))) .GT. AMPMAX) AMPMAX = ABS(U(4(J, I)))
      CONTINUE
      IF (AMPMAX .EQ. 0.0) AMPMAX = 1.0
      CALL DEFPEN
      CALL PSPACE(0.91, 0.99, 0.65, 0.90)
      CALL MAP(0.0, 1.0, 0.0, 1.0)
      CALL BORDER
      CALL CTMRAG(15)
      CALL PCSCEN(0.5, 0.8, 'Time')
      CALL PCSCEN(0.5, 0.7, 'Index')
      CALL PLOTINI(0.7, 0.6, IT)
      CALL PCSCEN(0.5, 0.4, 'SCALE')
      NSCALE = NINT(2000/AMPMAX)
      CALL PLOTINI(0.7, 0.3, NSCALE)
      XMAX = (REAL(IXMAX) + 1.) * H
      ZMAX = (REAL(IZMAX) + 1.) * H
      RATIO = XMAX / ZMAX
      IF (RATIO .LT. 1) THEN
      PO = RATIO * .85
      PX1 = 0.475 - PO / 2.
      PX2 = PX1 + PO
      PZ1 = 0.05
      PZ2 = 0.90
      NCHARS = ZMAX / .85
      ELSE
      PO = .85 / RATIO
      PZ1 = 0.475 - PO / 2.
      PZ2 = PZ1 + PO
      PX1 = 0.05
      PX2 = 0.90
      NCHARS = XMAX / .85
      END IF
      CALL PSPACE(PX1, PX2, PZ1, PZ2)
      CALL MAP(0.0, XMAX, ZMAX, 0.0)
      CALL BORDER
      CALL SCALES
      WRITE (6, *) I, ITYPE = ', I, ITYPE
      AMPMAX = SORT(AMPMAX)
      DO 550 IX = 1, IXMAX
      DO 540 IZ = 1, IZMAX
      IF (UR4(IZ, IX) .LT. 0.0) THEN
      TERMSQ = -SQRT(-1.*UR4(IZ, IX))
      ELSE
      TERMSQ = SQRT(UR4(IZ, IX))
      END IF
      RPIOT(IZ) = REAL(IX) + TERMSQ * 2. / AMPMAX
      CONTINUE
      CALL PLOTIN(RPIOT, 2, 1, IZMAX, 1)
      CONTINUE
      CALL FRAME
      NPLOT = NPLOT + 1
      END IF
560 CONTINUE
      AMPMAX = 0.0
      DO 580 IX = 1, IXMAX
      DO 570 IZ = 1, IZMAX
      IF (ABS(REFRES(IZ, IX)) .GT. AMPMAX) THEN

```

```

      AMPMAX = ABS(REFRES(IZ,IX))
      END IF
570 CONTINUE
580 CONTINUE
      LEN = (IZMAX + 2) * 4
      OPEN (10, FILE=IFNAM3, STATUS='UNKNOWN', FORM='UNFORMATTED', ACCESS=
1      'DIRECT', RECL=LEN)
      NXOUT = IXMAX
      IDSHOT = (NSHOT - 1) * (NXOUT + 1) + 1
      NIDC = IDSHOT
      WRITE (10,REC=NIDC) (ALPDUM(I),I=1,LEN)
      DO 590 J = 1, IXMAX
          NIDC = IDSHOT + J - IX0 + 1
          WRITE (10,REC=NIDC) (REFRES(I,J),I=1,IZMAX)
590 CONTINUE
      CLOSE (10)
      XOPL = REAL(IX0) * H
      X1PL = REAL(IXSMAX) * H
      XSIZ = X1PL-XOPL
      ZMAX = (REAL(IZMAX) * H + 1.
      ZMAX = (REAL(IZMAX) + 1.) * H
      RATIO = XMAX / ZMAX
      XOPL = 0.0
      X1PL = XMAX
      XSIZ = XMAX
      IF (RATIO.LT. 1) THEN
          P0 = RATIO * .85
          PX1 = 0.475 - P0 / 2.
          PX2 = PX1 + P0
          PZ1 = 0.05
          PZ2 = 0.90
          NCHARS = ZMAX / .85
      ELSE
          P0 = .85 / RATIO
          PZ1 = 0.475 - P0 / 2.
          PZ2 = PZ1 + P0
          PX1 = 0.05
          PX2 = 0.90
          NCHARS = XSIZ / .85
      END IF
      CALL PSPACE(PX1, PX2, PZ1, PZ2)
      CALL MAP(XOPL,X1PL,ZMAX,0.0)
      CALL MAP(0.0, XMAX, ZMAX, 0.0)
      CALL BORDER
      CALL SCALES
      DO 620 IX = 1, IXMAX
          DO 600 IZ = 1, IZMAX
              IF(IX.LT. IX0.OR. IX.GT. IXSMAX) THEN
                  GO TO 1800
              ELSE
                  RPILOT(IZ) = REAL(IX) * H + REFRES(IZ,IX) * 3. / AMPMAX
                  ENDIF
                  CONTINUE
                  CALL PJOIN(RPILOT, 2, 1, IZMAX, 1)
                  DO 610 IZ = 1, IZMAX
                      IF (RPILOT(IZ).GT. (REAL(IX)*H + 1.E-2)) THEN
                          CALL POSITN(REAL(IX)*H, Z(IZ))
                          CALL JOIN(RPILOT(IZ), Z(IZ))
                      END IF
600 CONTINUE
610 CONTINUE
620 CONTINUE
          RX0 = REAL(IX0) * H
          RZ0 = REAL(IZ0) * H
          CALL PLOTNC(RX0, RZ0, 250)
          DO 630 JJJ = 1, NSOR
              RX1 = REAL(IXS(JJJ)) * H
              RX2 = REAL(IZS(JJJ)) * H
              CALL PLOTNC(RX1, RX2, 245)
630 CONTINUE
          PZ3 = PZ2 * 1.1
          CALL PSPACE(PX1, PX2, PZ2, PZ3)
          CALL MAP(0.0, 1.0, 0.0, 1.0)
          CALL PLOTS(0.05, 0.5, 'DELAY TIME (mllisecs) :')
          DEL = DELAYT * 1.E3
          CALL PLOTNF(0.8, 0.5, DEL, 3)
          CALL GREND
          GO TO 710
      C JUMP TO AVOID FILTERING STAGE
      C 640 CONTINUE
      C Now filter in vertical direction to remove artifacts
      C
          N1 = 128
          N2 = 128
          FNYQ = .5 / H
          FHICUT = FNYQ
          FHIOFF = FNYQ
          FLOWCT = .125
          FLOOFF = 0.05
          DO 670 JJ = 1, IXMAX
              CALL ZERO(N1, ARR)
              DO 650 II = 1, IZMAX
                  ARR(II) = REFRES(II,JJ)
650 CONTINUE
              CALL BPSCOS(ARR, N1, CX, FLOWCT, FHICUT, FLOOFF, FNYQ)
              DO 660 II = 1, IZMAX
                  REFRES(II,JJ) = ARR(II)
660 CONTINUE
670 CONTINUE
          CALL PSPACE(PX1, PX2, PZ1, PZ2)
          CALL MAP(0.0, XMAX, ZMAX, 0.0)
          CALL BORDER
          CALL SCALES
          DO 700 IX = 1, IXMAX
              DO 680 IZ = 1, IZMAX
                  RPILOT(IZ) = REAL(IX) + REFRES(IZ,IX) * 2. / AMPMAX
                  CONTINUE
                  CALL PJOIN(RPILOT, 2, 1, IZMAX, 1)
                  DO 690 IZ = 1, IZMAX
                      IF (RPILOT(IZ).GT. (REAL(IX) + 1.E-2)) THEN
                          CALL POSITN(REAL(IX), Z(IZ))
                          CALL JOIN(RPILOT(IZ), Z(IZ))
                      END IF
690 CONTINUE
700 CONTINUE
          CALL PLOTNC(RX0, RZ0, 250)
          DO 710 JJJ = 1, NSOR
              RX1 = REAL(IXS(JJJ))
              RX2 = REAL(IZS(JJJ))

```

CALL PLOTNC(RXI, RXZ, 245)
710 CONTINUE
END

Appendix A.7

The forward modelling program

EXTRAP

The program EXTRAP is a forward modelling finite-difference program which generates synthetic seismograms for a given velocity model and source and receiver geometry. A tapered zero-phase wavelet is used as the source function and the wavefield may be plotted as it expands in time. The final seismograms recorded at the receiver positions are also plotted.

```

C
C PROGRAM TO CARRY OUT WAVEFIELD EXTRAPOLATION BY FINITE DIFFERENCE APPROX
C TO ACOUSTIC WAVE EQUATION
C SYNTHETIC SEISMOGRAM GENERATION SUPPORTED
C ALGORITHM TAKEN FROM MCMCHAN (GEOPHYSICS V50 P627-636)
C
C WRITTEN BY M.J. FINDLAY 1989
C
C PROGRAM EXTRAP
C IMPLICIT DOUBLE PRECISION (A - H, O - Z)
C PARAMETER (PI=3.14159265358, POLO=1.0, POL2=-.5)
C
C ABSORBING BOUNDARY CONDITIONS
C VALUES OF P0 & P2 (RENAUT&PETERSEN GEOPHYSICS 1989)
C (1,-.5) PADE
C (3G-2G**3,-2G) G=SIN(PI/8) CHEBYCHEV
C (21PI/64,-15PI/64) I2
C (10/3PI,-8/3PI) CHEBYCHEV-PADE
C
C SOURCE FUNCTION IS ZERO-PHASE BUTTERWORTH WAVELET
C
C DOUBLE PRECISION U(128,128,3), TERM1, TERM2, TERM3, A,
1 SORSIG(1024), V(128,128)
REAL SEIS(1024,40), TT(1024), AMPMAX, XMAX, ZMAX, AMPSC1, RREC
REAL UR4(128,128), RPL0T(1024), Z(128), BUTT(1024), BUTT2(1024),
1 RNL, TMAX
REAL BUT1, BUT2, BUT3, BUT4, VINL(5), WIDTH(5)
COMPLEX CBUTT(1024)
INTEGER NMOV(20), IXR(40), IZR(40)
CHARACTER*10 OPFILE, IPFILE, IFNAM1, IFNAM2
ITO = 3
NSAMS = 128 * 128
NSAMT = NSAMS * 6
CALL ZERO(NSAMT, U(1,1,1))
C
C THIS ZEROS ARRAY U(X,Z,T=0,1,2)
C
WRITE (6,*) 'ENTER GRID POINT SPACING (METRES): '
READ (5,*) H
WRITE (6,*) 'ENTER MAXIMUM GRID VALUES IN X & Z (<128) : '
READ (5,*) IXXMAX, IZMAX
WRITE (6,*) 'ENTER TIME SAMPLE INTERVAL (milliseconds): '
READ (5,*) DT
DT = DT * 1.E-3
FNYQ = .5 / DT
NT = 1024
NF = NT / 2
DE = FNYQ / NF
WRITE (6,*) 'ENTER SOURCE POSN (X,Z) GRIDPOINT NUMBERS'
READ (5,*) IX0, IZ0
WRITE (6,*) 'SOURCE IS ZERO PHASE BUTTERWORTH FNCN '
WRITE (6,*) 'Enter low-cut, slope, high-cut, slope '
READ (5,*) BUT1, BUT2, BUT3, BUT4
C
C HIGH CUT
RNL = ALOG10((2.*(10.*(BUT4/10.)) - 1.0)
WRITE (6,*) 'RNL = ', RNL
RNL = RNL / (2.*ALOG10(2.))
WRITE (6,*) 'RNL = ', RNL
DO 10 J = 1, NT / 2 + 1
C
C RFR = DF * REAL(J - 1)
TEM = 1. / (1. + ((RFR/BUT3)**(2.*RNL)))
BUTT(J) = SORT(TEM)
10 CONTINUE
C LOW CUT
WRITE (6,*) 'NOW LOW CUT...'
RNL = ALOG10((2.*(10.*(BUT2/10.)) - 1.0)
WRITE (6,*) 'RNL = ', RNL
RNL = RNL / (2.*ALOG10(2.))
WRITE (6,*) 'RNL = ', RNL
BUTT2(1) = 0.0
DO 20 J = 2, NT / 2 + 1
RFR = DF * REAL(J - 1)
TEM = 1. / (1. + ((BUT1/RFR)**(2.*RNL)))
BUTT2(J) = SORT(TEM)
20 CONTINUE
DO 30 J = 1, NT / 2 + 1
BUTT(J) = BUTT(J) * BUTT2(J)
30 CONTINUE
C TRANSFORM TO TIME DOMAIN
ISAM = 2
DO 40 I = NT, NT / 2 + 2, -1
BUTT(I) = BUTT(ISAM)
ISAM = ISAM + 1
40 CONTINUE
DO 50 I = 1, NT
CBUTT(I) = CMPLX(BUTT(I),0.0)
50 CONTINUE
CALL FORK(NT, CBUTT, 1.)
C
C APPLY TAPER
NTAP0 = NINT(1.E0/(BUT3*DT)) * 1.5
NTAP1 = NINT(1.E0/(BUT3*DT)) * 0.75
LTAP = NTAP0 - NTAP1
WRITE (6,*) 'NTAP1,NTAP0,LTAP', NTAP1, NTAP0, LTAP
DO 60 I = NTAP1, NTAP0
FACTOR = REAL(NTAP0 - I) / REAL(LTAP)
CBUTT(I + 1) = CBUTT(I + 1) * FACTOR
CBUTT(NT - I) = CBUTT(NT - I) * FACTOR
60 CONTINUE
DO 70 I = NTAP0 + 2, NT - NTAP0 - 1
CBUTT(I) = 0.0
70 CONTINUE
C Shift origin to NTAP0+1
CALL ZERO(NT, BUTT2)
DO 80 I = 1, NTAP0 + 1
BUTT2(I + NTAP0 + 1) = REAL(CBUTT(I))
BUTT2(I) = REAL(CBUTT(I + NT - NTAP0 - 1))
80 CONTINUE
C Normalise output wavelet
CALL NORMAN(NT, BUTT2)
BMAX = 0.0
DO 90 I = 1, NT
IF (ABS(BUTT2(I)).GT. BMAX) THEN
BMAX = ABS(BUTT2(I))
INDSHF = I
END IF
SORSIG(I) = DBLE(BUTT2(I))
90 CONTINUE

```

```

WRITE (6,*) 'INPUT SOURCE WAVEFIELD FROM FILES 1=Y 0=N'
100 FORMAT (A10)
NSAMP0 = 2
IF (IPSORY.EQ. 1) THEN
  WRITE (6,*) 'ENTER NAME OF FILE 1:'
  READ (5,100) IFNAM1
  WRITE (6,*) 'ENTER TIME SAMPLE NUMBER FOR FIRST FILE'
  READ (5,*) NSAMP0
  WRITE (6,*) 'ENTER NAME OF FILE 2:'
  READ (5,100) IFNAM2
  WRITE (6,*) 'ENTER SHIFT IN DEPTH TO APPLY +VE ==> UP '
  READ (5,*) IZSHFT
  OPEN (UNIT=11, FILE=IFNAM1, FORM='FORMATTED', STATUS='OLD')
  DO 110 JJ = 1, 128
    READ (11,120) (U(I1,JJ,1), I1=1,128)
  CONTINUE
  FORMAT (8(D10.4))
  CLOSE (11)
  OPEN (UNIT=12, FILE=IFNAM2, FORM='FORMATTED', STATUS='OLD')
  DO 130 JJ = 1, 128
    READ (12,120) (U(I1,JJ,2), I1=1,128)
  CONTINUE
130   CLOSE (12)
  DO 140 JJ = 1, 128 - IZSHFT
    DO 140 II = 1, 128
      U(I1,JJ,1) = U(I1,JJ + IZSHFT,1)
      U(I1,JJ,2) = U(I1,JJ + IZSHFT,2)
    CONTINUE
140   DO 150 JJ = 128 - IZSHFT, 128
      DO 150 II = 1, 128
        U(I1,JJ,1) = 0.0
        U(I1,JJ,2) = 0.0
      CONTINUE
150   CONTINUE
  WRITE (6,*) 'APPLY COSINE WEIGHTING TO SOURCE WAVEFIELD 1=Y:'
  READ (5,*) ICSANS
  IF (ICSANS.EQ. 1) THEN
    DO 160 JJ = 1, 128
      DO 160 II = 1, 128
        RLEN = SORT(REAL((II - IX0)**2 + (JJ - IZ0)**2))
        IF (RLEN.EQ. 0.0) RLEN = 1.0
        FAC = ABS(REAL(JJ - IZ0)) / RLEN
        U(I1,JJ,1) = U(I1,JJ,1) * FAC
        U(I1,JJ,2) = U(I1,JJ,2) * FAC
      CONTINUE
160   END IF
  END IF
  WRITE (6,*) 'ENTER NO. OF LAYERS : '
  READ (5,*) NLAYER
  IDPTH = 0
  VMAX = 0.0
  DO 190 NL = 1, NLAYER
    WRITE (6,*) 'ENTER NO. OF variations in layer : '
    READ (5,*) NVARIA
    WRITE (6,*) 'ENTER VELOCITY(S) OF MEDIUM ', NL, ' (m / sec)'
    READ (5,*) (VINL(I), I=1, NVARIA)
    WRITE (6,*) 'ENTER WIDTHS OF FIELD(S) IN METRES : '
    READ (5,*) (WIDTH(I), I=1, NVARIA)
    WRITE (6,*) 'ENTER THICKNESS OF LAYER ', NL, ' (metres)'
    READ (5,*) THICK
    NTHICK = NINT(THICK/H)
    IXF = 1
    DO 180 I = 1, NVARIA
      VL = VINL(I)
      IF (VL.GT. VMAX) VMAX = VL
      IXL = WIDTH(I) / H + IXF
      DO 170 IZ = 1, NTHICK
        DO 170 IX = IXF, IXL
          V(IX, IZ + IDPTH) = VL
        CONTINUE
170      CONTINUE
      IXF = IXL
180    CONTINUE
    IDPTH = IDPTH + NTHICK
190  CONTINUE
  WRITE (6,*) 'INCLUDE POINT DIFFRACTOR 1=Y 0=N:'
  READ (5,*) IYNANS
  IF (IYNANS.EQ. 1) THEN
    WRITE (6,*) 'ENTER IX, IZ, VEL(IX, IZ)'
    READ (5,*) IX, IZ, VPOINT
    V(IX, IZ) = VPOINT
  END IF
  WRITE (6,*) 'ENTER NUMBER OF DIFFERENT MOVIE FRAMES REQUIRED'
  READ (5,*) NMOVIE
  DO 200 NM = 1, NMOVIE
    WRITE (6,*) 'ENTER SAMPLE TIME #', NM
    READ (5,*) NMOV(NM)
    NMOV(NM) = NMOV(NM) + NTAPO + 1
  CONTINUE
200  CONTINUE
  WRITE (6,*) 'ENTER CUTOFF RATIO FOR S/N : '
  READ (5,*) CUTO
  WRITE (6,*) 'ENTER NUMBER OF RECEIVERS TO BE SEISMOGRAPHED : '
  READ (5,*) NREC
  DO 210 NR = 1, NREC
    WRITE (6,*) 'ENTER GRID LOCATION (IX, IZ) OF RECEIVER #', NR
    READ (5,*) IXR(NR), IZR(NR)
  CONTINUE
210  CONTINUE
  WRITE (6,*) '2nd or 4th Order finite difference approximation : '
  READ (5,*) NORO
  WRITE (6,*) 'APPLY T-RAMP 1=Y 0=N:'
  READ (5,*) IRAMP
  WRITE (6,*) 'Enter name of file for seismogram output : '
  READ (5,220) ODFILE
220  FORMAT (A10)
  C
  C UNIFORM VELOCITY TO START WITH
  C
  C
  C H-GRID POINT SPACING
  C
  KB40 = 1024 * 40
  CALL ZERO(KB40, SETS(1,1))
  IF (IPSORY.EQ. 1) GO TO 250
  C
  C NOW DEFINE INITIAL WAVEFIELD INCLUDING SOURCE TERM
  C
  DO 230 IT = 1
    WRITE (7,*) IT, SORSIG(IT)
    U(IX0, IZ0, ITO - 1) = (SORSIG(1))
  CONTINUE
  C
  C START OF SOURCE AT T = 0

```

```

C
260 A = (V(IX0,I20)*DT/H)
C IF (DT.GE.H/(SQRT(2.)*V(IX0,I20))) THEN
C   WRITE (6,*)'ALGORITHM UNSTABLE'
C   WRITE (6,*)'DT,H,V =', DT, H, V(IX0,I20)
C   STOP
C 240 END IF
C
C START LOOP OVER ALL TIME SAMPLES
C
250 NPL0T = 1
C   DO 410 IT = NSAMP0, NMOV(NMOVIE)
C     WRITE (6,*)'TIME SAMPLE : ', IT
C     TIMEIT = (IT - 1) * DT
C     DO 270 IZ = 2, IZMAX - 1
C
C INCLUDE FREE SURFACE BY MAKING U(IX,Z=0) = 0.0 ???
C
C   DO 270 IX = 2, IXMAX - 1
C     A = (V(IX,I2)*DT/H)
C     IF (NORD.EQ.2.OR.IX.EQ.2.OR.IZ.EQ.1.OR.
C       IX.EQ.IXMAX-1.OR.IZ.EQ.IZMAX-1)
C       THEN
C         TERM1 = 2 * (1 - 2*A**2) * U(IX,I2,ITO - 1)
C         TERM2 = U(IX,I2,ITO - 2)
C         TERM3 = U(IX + 1,I2,ITO - 1) + U(IX - 1,I2,ITO -
C           1) + U(IX,I2 + 1,ITO - 1)
C         IF (I2.GT.1) TERM3 = TERM3 + U(IX,I2 - 1,ITO -
C           1)
C         TERM3 = TERM3 * A ** 2
C         U(IX,I2,ITO) = TERM1 - TERM2 + TERM3
C       ELSE
C         TERM1 = (2. - 5.*A**2) * U(IX,I2,ITO - 1)
C         TERM2 = U(IX + 1,I2,ITO - 1) + U(IX,I2 + 1,ITO -
C           1) + U(IX - 1,I2,ITO - 1)
C         IF (I2.GT.1) TERM2 = TERM2 + U(IX,I2 - 1,ITO -
C           1)
C         TERM3 = U(IX + 2,I2,ITO - 1) + U(IX,I2 + 2,ITO -
C           1) + U(IX - 2,I2,ITO - 1)
C         IF (I2.GT.2) THEN
C           TERM3 = TERM3 + U(IX,I2 - 2,ITO - 1)
C         END IF
C         ALPHA = TERM1 + A ** 2 * (16*TERM2 - TERM3) / 12.
C         U(IX,I2,ITO) = ALPHA - U(IX,I2,ITO - 2)
C       END IF
C     IF (ABS(U(IX,I2,ITO))*TIMEIT.LT.A0*TRRF/CUTO)
C       U(IX,I2,ITO) = 0.0
C     DO 260 NR = 1, NREC
C       IF (IX.EQ.IXR(NR).AND.IZ.EQ.IZR(NR))
C         THEN
C           IF (IT.GT.(NTAP0 + 1)) SEIS(IT - NTAP0 - 1,
C             NR) = REAL(U(IX,I2,ITO))
C           END IF
C           IF (IX.EQ.IX0.AND.IZ.EQ.IZ0.AND.IT.IT.2*
C             NTAP0 + 3) THEN
C             U(IX,I2,ITO) = SORSIG(IT)
C             SEIS(IT,1) = REAL(SORSIG(IT))
C           END IF
C
C
260 CONTINUE
270 GO TO 399
C ABOVE LINE INSERTED TO REMOVE ABSORBING BOUNDARY CONDITIONS
C
C NOW APPLY ABSORBING BOUNDARY CONDITIONS
C OPTION TO MAKE SURFACE ABSORBING
C
C TOP FIRST
C
C   DO 280 IX = 3, IXMAX - 2
C     IZ = 1
C     A = (V(IX,I2) * DT / H
C     TERM1 = U(IX,I2,ITO - 1) + U(IX,I2 + 1,ITO - 1)
C     TERM1 = TERM1 * 2.0 * POL0
C     TERM2 = U(IX,I2 + 1,ITO) + U(IX,I2,ITO - 2)
C     TERM2 = TERM2 * (A - POL0 + 2.*POL2*A**2)
C     TERM3 = U(IX + 1,I2,ITO - 2) + U(IX - 1,I2,ITO - 2) + U(
C       IX + 1,I2 + 1,ITO) + U(IX - 1,I2 + 1,ITO)
C     TERM3 = -TERM3 * A ** 2 * POL2
C     UINIT = (TERM1 + TERM2 + TERM3) / (POL0 + A)
C     U(IX,I2,ITO) = UINIT - U(IX,I2 + 1,ITO - 2)
C     IF (ABS(U(IX,I2,ITO))*TIMEIT.LT.A0*TRRF/CUTO)
C       U(IX,I2,ITO) = 0.0
C
C   DO 300 IZ = 3, IZMAX - 2
C     IX = 1
C     A = (V(IX,I2) * DT / H
C     TERM1 = U(IX,I2,ITO - 1) + U(IX + 1,I2,ITO - 1)
C     TERM1 = TERM1 * 2.0 * POL0
C     TERM2 = U(IX + 1,I2,ITO) + U(IX,I2,ITO - 2)
C     TERM2 = TERM2 * (A - POL0 + 2.*POL2*A**2)
C     TERM3 = U(IX,I2 + 1,ITO - 2) + U(IX,I2 - 1,ITO - 2) + U(
C       IX + 1,I2 + 1,ITO) + U(IX + 1,I2 - 1,ITO)
C     TERM3 = -TERM3 * A ** 2 * POL2
C     UINIT = (TERM1 + TERM2 + TERM3) / (POL0 + A)
C     U(IX,I2,ITO) = UINIT - U(IX + 1,I2,ITO - 2)
C     IF (ABS(U(IX,I2,ITO))*TIMEIT.LT.A0*TRRF/CUTO)
C       U(IX,I2,ITO) = 0.0
C
C   CONTINUE
C 300 CONTINUE
C
C NOW BOTTOM
C
C   DO 310 IX = 3, IXMAX - 2
C     IZ = IZMAX
C     A = (V(IX,I2) * DT / H
C     TERM1 = U(IX,I2,ITO - 1) + U(IX,I2 - 1,ITO - 1)
C     TERM1 = TERM1 * 2.0 * POL0
C     TERM2 = U(IX,I2 - 1,ITO) + U(IX,I2,ITO - 2)
C     TERM2 = TERM2 * (A - POL0 + 2.*POL2*A**2)
C     TERM3 = U(IX + 1,I2 - 1,ITO) + U(IX - 1,I2 - 1,ITO) + U(
C       IX + 1,I2,ITO - 2) + U(IX - 1,I2,ITO - 2)
C     TERM3 = -TERM3 * A ** 2 * POL2
C     UINIT = (TERM1 + TERM2 + TERM3) / (A + POL0)
C     U(IX,I2,ITO) = UINIT - U(IX,I2 - 1,ITO - 2)
C
C   CONTINUE
310 CONTINUE

```

```

C
C NOW RHS
C
DO 320 IZ = 3, IZMAX - 2
  IX = IXMAX
  A = V(IX, IZ) * DT / H
  TERM1 = (U(IX, IZ, ITO) - 1) + U(IX - 1, IZ, ITO) - 1)
  TERM1 = TERM1 * 2.0 * POLO
  TERM2 = (U(IX - 1, IZ, ITO) + U(IX, IZ, ITO) - 2)
  TERM2 = TERM2 * (A - POLO + 2. * POL2 * A * 2)
  TERM3 = U(IX, IZ + 1, ITO - 2) + U(IX, IZ - 1, ITO - 2) + U(
    IX - 1, IZ + 1, ITO) + U(IX - 1, IZ - 1, ITO)
  TERM3 = -TERM3 * A * 2 * POL2
  UINIT = (TERM1 + TERM2 + TERM3) / (POLO + A)
  U(IX, IZ, ITO) = UINIT - U(IX - 1, IZ, ITO - 2)
  IF (ABS(U(IX, IZ, ITO)) * TIMEIT .LT. AO * TREF / CUTO)
    U(IX, IZ, ITO) = 0.0
320 CONTINUE
C
C NOW MAKE BOTTOM CORNERS ABSORBING TOO
C
C LOWER RIGHT HAND CORNER
C
  IX = IXMAX - 1
  IZ = IZMAX
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = IXMAX
  IZ = IZMAX
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = IXMAX
  IZ = IZMAX
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
C UPPER RIGHT HAND CORNER
C
  IX = IXMAX - 1
  IZ = 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = IXMAX
  IZ = 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = IXMAX
  IZ = 2
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
  IF (ABS(U(IX, IZ, ITO)) * TIMEIT .LT. AO * TREF / CUTO) U(IX, IZ, ITO) =

```

```

C
C 1 0.0
C
  IX = IXMAX
  IZ = 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
  IF (ABS(U(IX, IZ, ITO)) * TIMEIT .LT. AO * TREF / CUTO) U(IX, IZ, ITO) =
    0.0
C
C LOWER LEFT HAND CORNER
C
  IX = 1
  IZ = IZMAX - 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX + 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = 2
  IZ = IZMAX
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX + 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = 1
  IZ = IZMAX
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX + 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
C UPPER LEFT CORNER
C
  IX = 1
  IZ = 2
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ - 1, ITO) + U(IX + 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
  IX = 1
  IZ = 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
  IX = 1
  IZ = 1
  VDASH = V(IX, IZ)
  TERM1 = (U(IX, IZ + 1, ITO) + U(IX - 1, IZ, ITO)) / H
  TERM2 = (U(IX, IZ, ITO - 1)) * SQRT(2.) / (VDASH * DT)
  U(IX, IZ, ITO) = (TERM1 + TERM2) / (2. / H + SQRT(2.) / (VDASH * DT))
C
C MAKE THIS LINE THE LAST STEP
C
330 WRITE (7, *) IT, SORSIG(IT)
DO 340 J = 1, 128
DO 340 I = 1, 128

```

```

      U(I,J,IT0 - 2) = U(I,J,IT0 - 1)
      U(I,J,IT0 - 1) = U(I,J,IT0)
340    CONTINUE
      IF (IT .GE. NMOV(NMOVIE) - 1) THEN
C     WRITE OUT U(IX,IZ,ITEND-1) & U(IX,IZ,ITEND) TO FILE ATTACHED TO
C     UNIT 13
C
        DO 350 J = 1, 128
          WRITE (13,120) (U(I,J,IT0 - 1),I=1,128)
350    CONTINUE
      END IF
      IF (IT .EQ. NMOV(NPLOT)) THEN
C
C     NOW GET GRAPHICAL OUTPUT OF U(IX,IZ,ITEND)
C
        AMPMAX = 0.0
        DO 360 I = 1, IZMAX
          DO 360 J = 1, IZMAX
            UR4(U,I) = REAL(U(I,J,IT0 - 1))
            IF (ABS(UR4(J,I)) .GT. AMPMAX) AMPMAX = ABS(UR4(J,
              I))
          CONTINUE
        IF (AMPMAX .EQ. 0.0) AMPMAX = 1.0
        CALL PAPER(1)
        CALL PSPACE(0.91, 0.99, 0.65, 0.90)
        CALL MAP(0.0, 1.0, 0.0, 1.0)
        CALL BORDER
        CALL CTMAG(15)
        CALL PCSCEN(0.5, 0.8, 'Time')
        CALL PCSCEN(0.5, 0.7, 'Index')
        INDEX = IT - NTAP0
        CALL PLOTNI(0.7, 0.6, INDEX)
        CALL PCSCEN(0.5, 0.4, 'SCALE')
        RSCALE = (1.0/AMPMAX)
        CALL PLOTNE(0.7, 0.3, RSCALE, 2)
        CALL PSPACE(0.05, 0.90, 0.05, 0.90)
        XMAX = REAL(IXMAX) + 1.
        ZMAX = REAL(IZMAX) + 1.
        CALL MAP(0.0, XMAX, ZMAX, 0.0)
        CALL BORDER
        CALL SCALES
        DO 370 II = 1, IZMAX
          Z(II) = REAL(II)
370    CONTINUE
        DO 400 IX = 1, IXMAX
          DO 380 IZ = 1, IZMAX
            IF (UR4(IZ,IX) .LT. 0.0) THEN
              AMPPL = -1.0 * SORT(-UR4(IZ,IX))
            ELSE
              AMPPL = SORT(UR4(IZ,IX))
            END IF
            RPILOT(IZ) = REAL(IX) + AMPPL * 2. / SORT(AMPMAX)
          CONTINUE
        CALL PTOJOIN(RPILOT, 2, 1, IZMAX, 1)
        DO 390 IZ = 1, IZMAX
          IF (RPILOT(IZ) .GT. (REAL(IX) + 1.E-2))
            THEN
              CALL POSITN(REAL(IX), Z(IZ))
              CALL JOIN(RPILOT(IZ), Z(IZ))
            1
          380
          390
          400
          410
          420
          430
          440
          450
          460
          470
          480
          490
          500
          510
          520
          530
          540
          550
          560
          570
          580
          590
          600
          610
          620
          630
          640
          650
          660
          670
          680
          690
          700
          710
          720
          730
          740
          750
          760
          770
          780
          790
          800
          810
          820
          830
          840
          850
          860
          870
          880
          890
          900
          910
          920
          930
          940
          950
          960
          970
          980
          990
          1000
          1010
          1020
          1030
          1040
          1050
          1060
          1070
          1080
          1090
          1100
          1110
          1120
          1130
          1140
          1150
          1160
          1170
          1180
          1190
          1200
          1210
          1220
          1230
          1240
          1250
          1260
          1270
          1280
          1290
          1300
          1310
          1320
          1330
          1340
          1350
          1360
          1370
          1380
          1390
          1400
          1410
          1420
          1430
          1440
          1450
          1460
          1470
          1480
          1490
          1500
          1510
          1520
          1530
          1540
          1550
          1560
          1570
          1580
          1590
          1600
          1610
          1620
          1630
          1640
          1650
          1660
          1670
          1680
          1690
          1700
          1710
          1720
          1730
          1740
          1750
          1760
          1770
          1780
          1790
          1800
          1810
          1820
          1830
          1840
          1850
          1860
          1870
          1880
          1890
          1900
          1910
          1920
          1930
          1940
          1950
          1960
          1970
          1980
          1990
          2000
          2010
          2020
          2030
          2040
          2050
          2060
          2070
          2080
          2090
          2100
          2110
          2120
          2130
          2140
          2150
          2160
          2170
          2180
          2190
          2200
          2210
          2220
          2230
          2240
          2250
          2260
          2270
          2280
          2290
          2300
          2310
          2320
          2330
          2340
          2350
          2360
          2370
          2380
          2390
          2400
          2410
          2420
          2430
          2440
          2450
          2460
          2470
          2480
          2490
          2500
          2510
          2520
          2530
          2540
          2550
          2560
          2570
          2580
          2590
          2600
          2610
          2620
          2630
          2640
          2650
          2660
          2670
          2680
          2690
          2700
          2710
          2720
          2730
          2740
          2750
          2760
          2770
          2780
          2790
          2800
          2810
          2820
          2830
          2840
          2850
          2860
          2870
          2880
          2890
          2900
          2910
          2920
          2930
          2940
          2950
          2960
          2970
          2980
          2990
          3000
          3010
          3020
          3030
          3040
          3050
          3060
          3070
          3080
          3090
          3100
          3110
          3120
          3130
          3140
          3150
          3160
          3170
          3180
          3190
          3200
          3210
          3220
          3230
          3240
          3250
          3260
          3270
          3280
          3290
          3300
          3310
          3320
          3330
          3340
          3350
          3360
          3370
          3380
          3390
          3400
          3410
          3420
          3430
          3440
          3450
          3460
          3470
          3480
          3490
          3500
          3510
          3520
          3530
          3540
          3550
          3560
          3570
          3580
          3590
          3600
          3610
          3620
          3630
          3640
          3650
          3660
          3670
          3680
          3690
          3700
          3710
          3720
          3730
          3740
          3750
          3760
          3770
          3780
          3790
          3800
          3810
          3820
          3830
          3840
          3850
          3860
          3870
          3880
          3890
          3900
          3910
          3920
          3930
          3940
          3950
          3960
          3970
          3980
          3990
          4000
          4010
          4020
          4030
          4040
          4050
          4060
          4070
          4080
          4090
          4100
          4110
          4120
          4130
          4140
          4150
          4160
          4170
          4180
          4190
          4200
          4210
          4220
          4230
          4240
          4250
          4260
          4270
          4280
          4290
          4300
          4310
          4320
          4330
          4340
          4350
          4360
          4370
          4380
          4390
          4400
          4410
          4420
          4430
          4440
          4450
          4460
          4470
          4480
          4490
          4500
          4510
          4520
          4530
          4540
          4550
          4560
          4570
          4580
          4590
          4600
          4610
          4620
          4630
          4640
          4650
          4660
          4670
          4680
          4690
          4700
          4710
          4720
          4730
          4740
          4750
          4760
          4770
          4780
          4790
          4800
          4810
          4820
          4830
          4840
          4850
          4860
          4870
          4880
          4890
          4900
          4910
          4920
          4930
          4940
          4950
          4960
          4970
          4980
          4990
          5000
          5010
          5020
          5030
          5040
          5050
          5060
          5070
          5080
          5090
          5100
          5110
          5120
          5130
          5140
          5150
          5160
          5170
          5180
          5190
          5200
          5210
          5220
          5230
          5240
          5250
          5260
          5270
          5280
          5290
          5300
          5310
          5320
          5330
          5340
          5350
          5360
          5370
          5380
          5390
          5400
          5410
          5420
          5430
          5440
          5450
          5460
          5470
          5480
          5490
          5500
          5510
          5520
          5530
          5540
          5550
          5560
          5570
          5580
          5590
          5600
          5610
          5620
          5630
          5640
          5650
          5660
          5670
          5680
          5690
          5700
          5710
          5720
          5730
          5740
          5750
          5760
          5770
          5780
          5790
          5800
          5810
          5820
          5830
          5840
          5850
          5860
          5870
          5880
          5890
          5900
          5910
          5920
          5930
          5940
          5950
          5960
          5970
          5980
          5990
          6000
          6010
          6020
          6030
          6040
          6050
          6060
          6070
          6080
          6090
          6100
          6110
          6120
          6130
          6140
          6150
          6160
          6170
          6180
          6190
          6200
          6210
          6220
          6230
          6240
          6250
          6260
          6270
          6280
          6290
          6300
          6310
          6320
          6330
          6340
          6350
          6360
          6370
          6380
          6390
          6400
          6410
          6420
          6430
          6440
          6450
          6460
          6470
          6480
          6490
          6500
          6510
          6520
          6530
          6540
          6550
          6560
          6570
          6580
          6590
          6600
          6610
          6620
          6630
          6640
          6650
          6660
          6670
          6680
          6690
          6700
          6710
          6720
          6730
          6740
          6750
          6760
          6770
          6780
          6790
          6800
          6810
          6820
          6830
          6840
          6850
          6860
          6870
          6880
          6890
          6900
          6910
          6920
          6930
          6940
          6950
          6960
          6970
          6980
          6990
          7000
          7010
          7020
          7030
          7040
          7050
          7060
          7070
          7080
          7090
          7100
          7110
          7120
          7130
          7140
          7150
          7160
          7170
          7180
          7190
          7200
          7210
          7220
          7230
          7240
          7250
          7260
          7270
          7280
          7290
          7300
          7310
          7320
          7330
          7340
          7350
          7360
          7370
          7380
          7390
          7400
          7410
          7420
          7430
          7440
          7450
          7460
          7470
          7480
          7490
          7500
          7510
          7520
          7530
          7540
          7550
          7560
          7570
          7580
          7590
          7600
          7610
          7620
          7630
          7640
          7650
          7660
          7670
          7680
          7690
          7700
          7710
          7720
          7730
          7740
          7750
          7760
          7770
          7780
          7790
          7800
          7810
          7820
          7830
          7840
          7850
          7860
          7870
          7880
          7890
          7900
          7910
          7920
          7930
          7940
          7950
          7960
          7970
          7980
          7990
          8000
          8010
          8020
          8030
          8040
          8050
          8060
          8070
          8080
          8090
          8100
          8110
          8120
          8130
          8140
          8150
          8160
          8170
          8180
          8190
          8200
          8210
          8220
          8230
          8240
          8250
          8260
          8270
          8280
          8290
          8300
          8310
          8320
          8330
          8340
          8350
          8360
          8370
          8380
          8390
          8400
          8410
          8420
          8430
          8440
          8450
          8460
          8470
          8480
          8490
          8500
          8510
          8520
          8530
          8540
          8550
          8560
          8570
          8580
          8590
          8600
          8610
          8620
          8630
          8640
          8650
          8660
          8670
          8680
          8690
          8700
          8710
          8720
          8730
          8740
          8750
          8760
          8770
          8780
          8790
          8800
          8810
          8820
          8830
          8840
          8850
          8860
          8870
          8880
          8890
          8900
          8910
          8920
          8930
          8940
          8950
          8960
          8970
          8980
          8990
          9000
          9010
          9020
          9030
          9040
          9050
          9060
          9070
          9080
          9090
          9100
          9110
          9120
          9130
          9140
          9150
          9160
          9170
          9180
          9190
          9200
          9210
          9220
          9230
          9240
          9250
          9260
          9270
          9280
          9290
          9300
          9310
          9320
          9330
          9340
          9350
          9360
          9370
          9380
          9390
          9400
          9410
          9420
          9430
          9440
          9450
          9460
          9470
          9480
          9490
          9500
          9510
          9520
          9530
          9540
          9550
          9560
          9570
          9580
          9590
          9600
          9610
          9620
          9630
          9640
          9650
          9660
          9670
          9680
          9690
          9700
          9710
          9720
          9730
          9740
          9750
          9760
          9770
          9780
          9790
          9800
          9810
          9820
          9830
          9840
          9850
          9860
          9870
          9880
          9890
          9900
          9910
          9920
          9930
          9940
          9950
          9960
          9970
          9980
          9990
          10000
          10010
          10020
          10030
          10040
          10050
          10060
          10070
          10080
          10090
          10100
          10110
          10120
          10130
          10140
          10150
          10160
          10170
          10180
          10190
          10200
          10210
          10220
          10230
          10240
          10250
          10260
          10270
          10280
          10290
          10300
          10310
          10320
          10330
          10340
          10350
          10360
          10370
          10380
          10390
          10400
          10410
          10420
          10430
          10440
          10450
          10460
          10470
          10480
          10490
          10500
          10510
          10520
          10530
          10540
          10550
          10560
          10570
          10580
          10590
          10600
          10610
          10620
          10630
          10640
          10650
          10660
          10670
          10680
          10690
          10700
          10710
          10720
          10730
          10740
          10750
          10760
          10770
          10780
          10790
          10800
          10810
          10820
          10830
          10840
          10850
          10860
          10870
          10880
          10890
          10900
          10910
          10920
          10930
          10940
          10950
          10960
          10970
          10980
          10990
          11000
          11010
          11020
          11030
          11040
          11050
          11060
          11070
          11080
          11090
          11100
          11110
          11120
          11130
          11140
          11150
          11160
          11170
          11180
          11190
          11200
          11210
          11220
          11230
          11240
          11250
          11260
          11270
          11280
          11290
          11300
          11310
          11320
          11330
          11340
          11350
          11360
          11370
          11380
          11390
          11400
          11410
          11420
          11430
          11440
          11450
          11460
          11470
          11480
          11490
          11500
          11510
          11520
          11530
          11540
          11550
          11560
          11570
          11580
          11590
          11600
          11610
          11620
          11630
          11640
          11650
          11660
          11670
          11680
          11690
          11700
          11710
          11720
          11730
          11740
          11750
          11760
          11770
          11780
          11790
          11800
          11810
          11820
          11830
          11840
          11850
          11860
          11870
          11880
          11890
          11900
          11910
          11920
          11930
          11940
          11950
          11960
          11970
          11980
          11990
          12000
          12010
          12020
          12030
          12040
          12050
          12060
          12070
          12080
          12090
          12100
          12110
          12120
          12130
          12140
          12150
          12160
          12170
          12180
          12190
          12200
          12210
          12220
          12230
          12240
          12250
          12260
          12270
          12280
          12290
          12300
          12310
          12320
          12330
          12340
          12350
          12360
          12370
          12380
          12390
          12400
          12410
          12420
          12430
          12440
          12450
          12460
          12470
          12480
          12490
          12500
          12510
          12520
          12530
          12540
          12550
          12560
          12570
          12580
          12590
          12600
          12610
          12620
          12630
          12640
          12650
          12660
          12670
          12680
          12690
          12700
          12710
          12720
          12730
          12740
          12750
          12760
          12770
          12780
          12790
          12800
          12810
          12820
          12830
          12840
          12850
          12860
          12870
          12880
          12890
          12900
          12910
          12920
          12930
          12940
          12950
          12960
          12970
          12980
          12990
          13000
          13010
          13020
          13030
          13040
          13050
          13060
          13070
          13080
          13090
          13100
          13110
          13120
          13130
          13140
          13150
          13160
          13170
          13180
          13190
          13200
          13210
          13220
          13230
          13240
          13250
          13260
          13270
          13280
          13290
          13300
          13310
          13320
          13330
          13340
          13350
          13360
          13370
          13380
          13390
          13400
          13410
          13420
          13430
          13440
          13450
          13460
          13470
          13480
          13490
          13500
          13510
          13520
          13530
          13540
          13550
          13560
          13570
          13580
          13590
          13600
          13610
          13620
          13630
          13640
          13650
          13660
          13670
          13680
          13690
          13700
          13710
          13720
          13730
          13740
          13750
          13760
          13770
          13780
          13790
          13800
          13810
          13820
          13830
          13840
          13850
          13860
          13870
          13880
          13890
          13900
          13910
          13920
          13930
          13940
          13950
          13960
          13970
          13980
          13990
          14000
          14010
          14020
          14030
          14040
          14050
          14060
          14070
          14080
          14090
          14100
          14110
          14120
          14130
          14140
          14150
          14160
          14170
          14180
          14190
          14200
          14210
          14220
          14230
          14240
          14250
          14260
          14270
          14280
          14290
          14300
          14310
          14320
          14330
          14340
          14350
          14360
          14370
          14380
          14390
          14400
          14410
          14420
          14430
          14440
          14450
          14460
          14470
          14480
          14490
          14500
          14510
          14520
          14530
          14540
          14550
          14560
          14570
          14580
          14590
          14600
          14610
          14620
          14630
          14640
          14650
          14660
          14670
          14680
          14690
          14700
          14710
          14720
          14730
          14740
          14750
          14760
          14770
          14780
          14790
          14800
          14810
          14820
          14830
          14840
          14850
          14860
          14870
          14880
          14890
          14900
          14910
          14920
          14930
          14940
          14950
          14960
          14970
          14980
          14990
          15000
          15010
          15020
          15030
          15040
          15050
          15060
          15070
          15080
          15090
          15100
          15110
          15120
          15130
          15140
          15150
          15160
          15170
          15180
          15190
          15200
          15210
          15220
          15230
          15240
          15250
          15260
          15270
          15280
          15290
          15300
          15310
          15320
          15330
          15340
          15350
          15360
          15370
          15380
          15390
          15400
          15410
          15420
          15430
          15440
          15450
          15460
          15470
          15480
          15490
          15500
          15510
          15520
          15530
          15540
          15550
          15560
          15570
          15580
          15590
          15600
          15610
          15620
          15630
          15640
          15650
          15660
          15670
          15680
          15690
          15700
          15710
          15720
          15730
          15740
          15750
          15760
          15770
          15780
          15790
          15800
          15810
          15820
          15830
          15840
          15850
          15860
          15870
          15880
          15890
          15900
          15910
          15920
          15930
          15940
          15950
          15960
          15970
          15980
          15990
          16000
          16010
          16020
          16030

```

Appendix A.8

The migration program

KIRCHMIG

The program KIRCHMIG carries out Kirchhoff-type migrations by raytracing from each image point to each source and receiver using the subroutine RAYTRA (appendix A.5). The image time calculated for each source receiver pairing is then used to extract the contribution to the image from the appropriate sample in the seismic data. The migrated image is then output to a file for later viewing. The program include several processing options such as the choice of migration operator and the range of reflector dips allowed.

```

C . Program to do Kirchhoff migration using RAYTRA anisotropic
C raytracing algorithm
C Allow general source/receiver positions
C Options of different diffraction stacks: Diffraction stack,
C Kirchhoff,
C GRT
C Selective raypath muting for shallow-dipping rays, and geological
C dip range
C Needs subroutine RAYTRA for raytracing
C subroutine library TSAR4_L (Time series analysis -- Robinson,
C Claerbout et al.)
C
C PROGRAM KIRCH
C
C DEPTH(1lay) = depth to top of layer 1lay
C VH(1lay) = horizontal velocity in layer 1lay
C VV(1lay) = vertical velocity in layer 1lay
C (XSOR(1sor),ZSOR(1sor)) = coordinates of source 1sor
C (XREC(1rec),ZREC(1rec)) = coordinates of receiver 1rec
C CSG(1time,1rec,1sor) = recorded data
C IMAGE(1depth,1offset) = IMAGED reflectivity at (1depth,1offset)
C
C
C VERSION 4.0 WITH FASTER RAYTRACING CODE
C
PARAMETER (NSW=512,NSOR1=27,NREC1=24,NIMAGZ=200,NIMAGX=128)
REAL DEPTH(31), VH(31), VV(31), XSOR(100), ZSOR(100)
REAL XREC(NREC1), ZREC(NREC1), CSG(NSM,NREC1,NSOR1)
REAL IMAGE(NIMAGZ,NIMAGX)
REAL TAA5(NSOR1), TAAUR(NREC1), ANGS(NSOR1), ANGG(NREC1),
1 ANGO(NSOR1)
REAL ANGL(NREC1)
COMPLEX CX(NSM), C0, C1
CHARACTER*17 FNAME
PI = 3.1415926535
PI2 = PI / 2.
DO 10 I = 1, 256
A(I) = 'e'
10 CONTINUE
NIMAGE = NIMAGZ * NIMAGX
CALL ZERO(NIMAGE, IMAGE)
C
C SET UP VELOCITY MODEL
C
WRITE (6,*) 'ENTER NO. OF LAYERS : '
READ (5,*) NIMAX
NDEPTH = NIMAX + 1
DEPTH(1) = 0.0
DO 20 J = 1, NIMAX
WRITE (6,*) 'Enter VH,VV,Thickness(S.I.) for layer ', J
READ (5,*) VH(J), VV(J), THICK
DEPTH(J + 1) = THICK + DEPTH(J)
20 CONTINUE
C
C SET UP RAYTRACING PARAMETERS
C
WRITE (6,*) 'ENTER RAYTRACING ACCURACY IN METRES : '
READ (5,*) ACC
WRITE (6,*) 'Enter no. of angles to try for each raypath : '
READ (5,*) MAXLOP
WRITE (6,*) 'ENTER GRID POINT SPACINGS (DX,DZ) : '
READ (5,*) DXGRID, DZGRID
WRITE (6,*) 'ENTER MINIMUM AND MAXIMUM X-VALUES : '
READ (5,*) XMIN, XMAX
WRITE (6,*) 'Enter imaging grid area (X0,X1,Z0,Z1) : '
READ (5,*) XIM0, XIM1, ZIM0, ZIM1
NX = INT((XMAX - XMIN)/DXGRID) + 1
NZ = INT((DEPTH(NDEPTH)/DZGRID) - 1
C
C SET UP SOURCE AND RECEIVER COORDINATES
C
WRITE (6,*) 'Enter no. of source positions : '
READ (5,*) NSOR
DO 30 J = 1, NSOR
WRITE (6,*) 'ENTER COORDINATES OF SOURCE ', J, ' : '
READ (5,*) XSOR(J), ZSOR(J)
30 CONTINUE
WRITE (6,*) 'Enter no. of receiver positions : '
READ (5,*) NREC
DO 40 J = 1, NREC
WRITE (6,*) 'ENTER COORDINATES OF RECEIVER ', J, ' : '
READ (5,*) XREC(J), ZREC(J)
40 CONTINUE
DSG = SQRT((ZREC(2) - ZREC(1))**2 + (XREC(2) - XREC(1))**2)
IF (NREC.EQ.1) DSG = 1.0
DSS = SQRT((ZSOR(2) - ZSOR(1))**2 + (XREC(1))**2)
IF (NSOR.EQ.1) DSS = 1.0
WRITE (6,*) 'Apply Newman amplitude-phase correction (1=y) ?'
READ (5,*) NEWMAN
WRITE (6,*) 'Enter aperture,taper,raymute,apemute in degrees : '
READ (5,*) APRANG, APRTAP, THETMU, APRMUT
APRMUT is also used as stacking angle cut-off
APRANG = .5 * APRANG * PI / 180.
APRTAP = APRTAP * PI / 180.
THETMU = THETMU * PI / 180.
APRMUT = .5 * APRMUT * PI / 180.
WRITE (6,*) 'ENTER Migration type 1=K1r stack,2=K1r Int,3=GK,4=GRT,
15=CDP gath'
READ (5,*) IMIGTY
IF (IMIGTY.EQ.5) NX = 1
WRITE (6,*) 'ENTER stacking cutoff : '
READ (5,*) STKOFF
WRITE (6,*) 'ENTER DIPS OF SOURCE AND RECEIVER ARRAYS IN DEGREES : '
READ (5,*) DIPS, DIPR
DIPS = DIPS * PI / 180.
DIPR = DIPR * PI / 180.
C
C READ IN DATA FILE
C
WRITE (6,*) 'Enter name of datafile to migrate : '
READ (5,50) FNAME
50 FORMAT (A17)
WRITE (6,*) 'UP (1) or DOWN (0) -going wavefield : '
READ (5,*) IUP
IF (FNAME.EQ.'DUMMY') THEN

```

```

C INVERSE IMPULSE RESPONSE
C
WRITE (6,*) 'Enter coordinates of diffracting point : '
READ (5,*) XDIF, ZDIF
DO 70 K = 1, NSOR
  DO = SQRT((XSOR(K) - XDIF)**2 + (ZSOR(K) - ZDIF)**2)
  T0 = D0 / VV(1)
  NSAMR = 256
  DT = .5E-3
  K24 = NSM * NREC
  CALL ZERO(K24, CSG(1,1,K))
  DO 60 J = 1, NREC
    D1 = SQRT((XREC(J) - XDIF)**2 + (ZREC(J) - ZDIF)**2)
    T1 = D1 / VV(1)
    TIMAG = T0 + T1
    AMP = 1.0 / SQRT(TIMAG)
    NIMAG = INT(TIMAG/DT) + 1
    CSG(NIMAG,J,K) = AMP
  CONTINUE
60 CONTINUE
70 CONTINUE
ELSE
  WRITE (6,*) 'Enter no. of samples per trace & NO. NEEDED : '
  READ (5,*) NSAMR, NSAMO
  WRITE (6,*) 'Enter sampling interval in milliseconds : '
  READ (5,*) DT
  DT = DT * 1.E-3
  LEN = 4 * (NSAMR + 2)
  WRITE (6,*) 'Enter I.D. of first common shot gather to migrate
1: '
  READ (5,*) IDSHOT
  WRITE (6,*) 'Enter spreading correction (power of T) to apply: '
  READ (5,*) TPOWER
  OPEN (10, FILE=FNAME, STATUS='OLD', ACCESS='DIRECT', FORM=
    'UNFORMATTED', RECL=LEN)
  DO 160 J = 1, NSOR
    WRITE (6,*) 'READING SHOT ... ', J
    JREC = (NREC + 1) * (J - 2 + IDSHOT) + 1
    RMS = 0.0
    DO 90 K = 1, NREC
      KREC = JREC + K
      READ (10, REC=KREC) (CSG(I,K,J), I=1, NSAMO)
    CONTINUE
  CONTINUE
  DO 80 I = 1, NSAMO
    CSG(I,K,J) = CSG(I,K,J) * (REAL(I)) ** TPOWER
    RMS = RMS + CSG(I,K,J) ** 2
  CONTINUE
  RMS = SQRT(RMS/REAL(NSAMO*NREC))
90 CONTINUE
80 CONTINUE
CC
C NOW EQUATE RMS ENERGIES OF CSG
CC
  DO 110 K = 1, NREC
    DO 100 I = 1, NSAMO
      CSG(I,K,J) = CSG(I,K,J) / RMS
    CONTINUE
  CONTINUE
100 CONTINUE
110 CONTINUE
C NOW APPLY 1/2-DIFFERENTIAL OPERATOR IF NEEDED
C
C
IF (NEWMAN.EQ. 1) THEN
  FNYQ = .5 / DT
  DF = FNYQ / REAL(NSAMO/2)
  DO 150 K = 1, NREC
    DO 120 I = 1, NSAMO
      CX(I) = CSG(I,K,J)
    CONTINUE
    CALL FORK(NSAMO, CX, -1.0)
  CONTINUE
120 CONTINUE
C
C FORWARD FFT
C
  CX(1) = 0.0
  C0 = SQRT(REAL(NSAMO/2)) * (1., -1.)
  C0 = 0.0
  CX(NSAMO/2 + 1) = CX(NSAMO/2 + 1) * C0
C
C Subtract PI/4 from PHASE AND MULTIPLY CX BY OMEGA**.5 RAMP
C
  DO 130 I = 2, NSAMO / 2
    FI = DF * (I - 1)
    C1 = SQRT(REAL(I - 1)) * (1., -1.)
    IF (FI.GT. 3*FNYQ/4.) THEN
      C1 = SQRT(REAL(NSAMO)*3./8.) * (1., -1.)
      C1 = C1 * 4. * (FNYQ - FI) / FNYQ
    END IF
    CX(I) = CX(I) * C1
    CX(NSAMO + 2 - I) = CONJG(CX(I))
  CONTINUE
130 CONTINUE
C
C INVERSE FFT
C
  CALL FORK(NSAMO, CX, +1.0)
  DO 140 I = 1, NSAMO
    CSG(I,K,J) = CX(I)
  CONTINUE
140 CONTINUE
150 CONTINUE
160 CONTINUE
  END IF
  CLOSE (10)
END IF
C
C . DATA NOW IN ARRAY CSG(time, receiver, source)
C
  NTRIES = MAXIOP
  NTOT = NX * NZ
  LRAIOR = 0
  LOST = 0
  IFOUND = 0
C
C LOOP OVER EACH IMAGE POINT DEPTH
C
  DO 220 JJ = 1, NZ
    ZSTART = JJ * DZGRID
    IF (ZSTART.LT. ZIMO.OR. ZSTART.GT. ZIMI) GO TO 220
    WRITE (6,*) 'Migrating depth ', ZSTART, '....'
  CONTINUE
C
C LOOP OVER EACH IMAGE POINT HORIZONTALLY
C
  DO 210 II = 1, NX

```

```

      XSTART = (I1 - 1) * DXGRID + XMIN
      NMUTE = 0
      IF (IMAGTY.EQ. 5) XSTART = XIMO
C
C COMMON DEPTH POINT GATHER
C
      IF (XSTART.LT. XIMO .OR. XSTART.GT. XIM1) GO TO 210
      STACK = 0.0
C
C LOOP OVER EACH SOURCE POINT
C
C PUT TIMES INTO ARRAY TAUS(NSOR), ANGLES INTO ANG0, ANG5
C
      CALL ZERO(NSOR, TAUS)
      CALL ZERO(NSOR, ANG0)
      CALL ZERO(NSOR, ANG5)
      DO 170 KK = 1, NSOR
        XEND = XSOR(KK)
        ZEND = ZSOR(KK)
        IF (IUP.EQ. 1) THEN
          IF (ZSTART - 0.0.LT. ZEND) GO TO 170
        ELSE
          IF (ZSTART + 0.0.GT. ZEND) GO TO 170
        CHECK FOR DOWNGOING WAVEFIELD
      END IF
C
C NOW RAY TRACE TO SOURCE FROM IMAGE POINT
C
      THETAP = 9.9
      TAUD = 0.0
      CALL RAYTRA(NIMAX, NDEPTH, DEPTH, VH, VV, XSTART,
        ZSTART, XEND, ZEND, DZGRID, ACC, NTRIES, THETAP,
        THETA0, TAUD, THETAS, NRAY)
      IF (NRAY.EQ. NTRIES) THEN
C RAY NOT FOUND
        WRITE (7,*) 'Source position not found'
        WRITE (7,*) 'XSTART,ZSTART = ', XSTART, ZSTART
        WRITE (7,*) 'XEND,ZEND = ', XEND, ZEND
        LOST = LOST + NREC
        GO TO 170
      END IF
      TAUS(KK) = TAUD
      ANG0(KK) = THETA0
      ANG5(KK) = THETAS
      IFOUND = IFOUND + 1
170 CONTINUE
C
C NOW RAY trace to receivers
C
      THETAP = 9.9
      CALL ZERO(NREC, TAUR)
      CALL ZERO(NREC, ANG1)
      CALL ZERO(NREC, ANG2)
      DO 180 LL = 1, NREC
        XEND = XREC(LL)
        ZEND = ZREC(LL)
        IF (IUP.EQ. 1) THEN
          IF (ZSTART - 0.0.LT. ZEND) GO TO 180
        CHECK FOR UPGOING WAVEFIELD
      ELSE
        IF (ZSTART + 0.0.GT. ZEND) GO TO 180
        CHECK FOR DOWNGOING WAVEFIELD
      END IF
      CALL RAYTRA(NIMAX, NDEPTH, DEPTH, VH, VV, XSTART,
        ZSTART, XEND, ZEND, DZGRID, ACC, NTRIES, THETAP,
        THETA1, TAUI, THETAG, NRAY)
      IF (NRAY.EQ. NTRIES) THEN
C RAY NOT FOUND
        WRITE (6,*) 'Receiver position not found'
        WRITE (7,*) 'XSTART,ZSTART = ', XSTART, ZSTART
        WRITE (7,*) 'XEND,ZEND = ', XEND, ZEND
        LOST = LOST + 1
        GO TO 180
      END IF
      TAUR(LL) = TAUI
      ANG1(LL) = THETA1
      ANG2(LL) = THETAG
      IFOUND = IFOUND + 1
180 CONTINUE
C
C LOOP FOR EACH SOURCE
C
      DO 200 KK = 1, NSOR
        TAUD = TAUS(KK)
        THETA0 = ANG0(KK)
        THETAS = ANG5(KK)
        IF (TAUD.EQ. 0.0) GO TO 200
        IF (THETA0.GT. PI) THEN
          IF (3.*PI2 - THETA0.GT. THETMU) GO TO 200
        ELSE IF (THETA0.GT. PI2) THEN
          IF (THETA0 - PI2.GT. THETMU) GO TO 200
        ELSE IF (THETA0.GT. 0.0) THEN
          IF (PI2 - THETA0.GT. THETMU) GO TO 200
        ELSE
          IF (PI2 + THETA0.GT. THETMU) GO TO 200
        END IF
        XEND = XSOR(KK)
        ZEND = ZSOR(KK)
        IF (IUP.EQ. 1) THEN
          IF (ZSTART.LT. ZEND) GO TO 200
        CHECK FOR UPGOING WAVEFIELD
        ELSE
          IF (ZSTART.GT. ZEND) GO TO 200
        CHECK FOR DOWNGOING WAVEFIELD
      END IF
C
C LOOP FOR EACH RECEIVER
C
      DO 190 LL = 1, NREC
        TAUI = TAUR(LL)
        THETA1 = ANG1(LL)
        THETAG = ANG2(LL)
        THETAL = THETAG
        IF (THETA1.GT. PI) THEN
          IF (3.*PI2 - THETA1.GT. THETMU)
            GO TO 190
        ELSE IF (THETA1.GT. PI2) THEN
          IF (THETA1 - PI2.GT. THETMU) GO TO 190
        ELSE IF (THETA1.GT. 0.0) THEN

```

```

C IF (PI2 - THETA1 .GT. THETMU) GO TO 190
ELSE
  IF (PI2 + THETA1 .GT. THETMU) GO TO 190
END IF
  IF (TAU1 .EQ. 0.0) GO TO 190
  XEND = XREC(LI)
  ZEND = ZREC(LI)
  IF (IUP .EQ. 1) THEN
    IF (ZSTART .LT. ZEND) GO TO 190
  ELSE
    IF (ZSTART .GT. ZEND) GO TO 190
  END IF
C CHECK FOR DOWNGOING WAVEFIELD
END IF
C CHECK to see if ray within imaging aperture
  XIANG = (THETA0 + THETA1) / 2.
  IF (XIANG .GT. PI) THEN
    IF (ABS(XIANG - 3.*PI2) .GT. APRANG)
      THEN
        WRITE(6,*) 'XIANG OUT OF RANGE'
        GO TO 190
      ELSE
        IF (ABS(XIANG - 3.*PI2) .LT. APRMUT)
          NMUTE = NMUTE + 1
        END IF
        XIANG = ABS(XIANG - 3.*PI2)
      ELSE IF (XIANG .GT. 0.0) THEN
        IF (ABS(XIANG - PI2) .GT. APRANG)
          THEN
            WRITE(6,*) 'XIANG OUT OF RANGE'
            GO TO 190
          ELSE
            IF (ABS(XIANG - PI2) .LT. APRMUT)
              NMUTE = NMUTE + 1
            END IF
            XIANG = ABS(XIANG - PI2)
          ELSE
            IF (ABS(XIANG + PI2) .GT. APRANG)
              THEN
                WRITE(6,*) 'XIANG OUT OF RANGE'
                GO TO 190
              ELSE
                IF (ABS(XIANG + PI2) .LT. APRMUT)
                  NMUTE = NMUTE + 1
                END IF
                XIANG = ABS(XIANG + PI2)
              END IF
            END IF
          C Source and receiver found so image appropriate travel time
          C
          C KIRCHHOFF DIFFRACTION STACK
          FAC = 1.0
          ELSE IF (IMIGTY .EQ. 2 .OR. IMIGTY .EQ. 6)
            THEN
C
1
C KIRCHHOFF INTEGRAL - THETA1 IS FINAL (RECEIVER) RAY ANGLE
  IF (TAU1 .EQ. 0.0) GO TO 190
  THETAG = THETAG - DIPR - PI / 2.
  FAC = SQRT(TAU0/TAU1) * ABS(COS(THETAG))
  ELSE IF (IMIGTY .EQ. 3 .OR. IMIGTY .EQ. 4)
    THEN
C
1
C GK Integral or GRT Integral
  IF (TAU1 .EQ. 0.0 .OR. TAU0 .EQ. 0.0)
    GO TO 190
  THETAS = THETAS - DIPR - PI / 2.
  THETAG = THETAG - DIPR - PI / 2.
  IF (IMIGTY .EQ. 5) THEN
    THETAS = 0.0
    THETAG = 0.0
  END IF
  FAC = (DSS*ABS(COS(THETAS)))*SQRT(TAU1/TAU0) +
    DSG*ABS(COS(THETAG))*SQRT(TAU0/TAU1)
  IF (IMIGTY .EQ. 4) THEN
    SCAT = THETA0 - THETA1
    CSCAT2 = COS(SCAT/2.0)
    FAC = FAC * CSCAT2 ** 2
  END IF
  END IF
  STACK = STACK + 1.0
  IF (ABS(XIANG) .LT. (APRANG - APRAP))
    THEN
      RAMP = 1.0
    ELSE
      RAMP = 1.0 - (ABS(XIANG) - (APRANG - APRAP))
        / (APRTAP)
    END IF
  C
1
C TAPER KIRCHHOFF OPERATOR
  DELTAP = CSG(NT,LI,KK) * FAC * RAMP
  IMAGE(JJ,II) = IMAGE(JJ,II) + DELTAP * REAL(IUP*2
    - 1)
  C
1
C REVERSES SIGN OF CONTRIBUTION FROM DGW
C
C NEXT RECEIVER TRACE
  CONTINUE
190
C
C NEXT SOURCE POSITION
  CONTINUE
200
C
C NEXT GRID POSITION
  STACK = REAL(NMUTE)
  IF (STACK .GT. STKOFF) THEN
    IMAGE(JJ,II) = IMAGE(JJ,II) / STACK
  ELSE
    IMAGE(JJ,II) = 0.0
  END IF
  IF (NMUTE .EQ. 0) IMAGE(JJ,II) = 0
C

```

```

C MUTES point if NO contributions from narrow aperture
C
210 CONTINUE
C NEXT GRID DEPTH
C
220 CONTINUE
    WRITE (6,*) 'Enter name of output file for image : '
    READ (5,50) FNAME
    LEN = (NZ + 2) * 4
    WRITE (6,*) 'Enter I.D. of image to save : '
    READ (5,*) ISHOT
    IDCODE = (ISHOT - 1) * (NX + 1) + 1
    OPEN (11, FILE=FNAME, STATUS='UNKNOWN', ACCESS='DIRECT', FORM=
1      'UNFORMATTED', RECL=LEN)
    WRITE (6,*) 'WRITING HEADER TO RECORD ..', IDCODE
    WRITE (11, REC=IDCODE) (A(I), I=1, NZ)
    DO 230 K = 1, NX
        XIM = K * DXGRID + XMIN
        KREC = IDCODE + K
        WRITE (11, REC=KREC) (IMAGE(I, K), I=1, NZ)
230 CONTINUE
    WRITE (6,*) 'LEN, NX, NZ = ', LEN, NX, NZ
    WRITE (6,*) 'LOST RAYS < ', LOST
    WRITE (6,*) 'FOUND RAYS = ', IFOUND
    PERCL = 100. * REAL(LOST) / REAL(LOST + IFOUND)
    WRITE (6,*) 'PERCENT LOST < ', PERCL
    CLOSE (11)
END

```

