Durham E-Theses

Advances in crosshole seismic reflection processing

Rowbotham, Peter S

How to cite:

Rowbotham, Peter S (1993) *Advances in crosshole seismic reflection processing*, Durham theses, Durham University. Available at Durham E-Theses Online: http://etheses.dur.ac.uk/5760/

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.

Academic Support Office, Durham University, University Office, Old Elvet, Durham DH1 3HP
e-mail: e-theses.admin@dur.ac.uk Tel: +44 0191 334 6107
http://etheses.dur.ac.uk

Advances in Crosshole Seismic Reflection Processing

by

Peter S Rowbotham

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.

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

Department of Geological Sciences
The University of Durham

1993

UNIV DUNELM

8 DEC 1993

Abstract

In recent years there have been significant advances in the acquisition and processing of crosshole seismic reflection data, and the method has been shown to be a high resolution imaging technique. However, the fidelity of the final images produced by this technique needs to be considered carefully to avoid incorrect interpretation. This thesis concerns the imaging capability of crosshole surveys, as well as advances made in processing techniques for application to crosshole seismic reflection surveys.

In a migrated seismic section, a meaningful image is only obtained if a range of dips around the local structural dip is sampled at each image point. For crosshole seismic reflection surveys, the distribution of dips sampled at an image point is controlled principally by the survey geometry, including source and receiver array lengths and their element spacings. By considering the dips sampled, the imaging capability of crosshole reflection surveying is discussed, with suggestions as to how to ensure optimal imaging of the target zone.

To overcome problems encountered in applying standard processing procedures, two new processing techniques are presented which enhance the imaging potential of crosshole reflection seismics. Generalised Berryhill migration has been developed as a full generalised Kirchhoff migration to include the near-field term, with the aim of improving image accuracy close to the source and receiver arrays. 3-D *f-k-k* filtering is an improved method of wavefield separation for crosshole seismic data.

Finally, the results of processing three types of dataset are presented. One is from a site in the Groningen gas field, another was acquired through a model interrogated at ultrasonic frequencies in a water tank, and the third type was acquired using coal exploration boreholes in Yorkshire. The results demonstrate the imaging capability of the crosshole reflection method, and the success of the two new processing schemes.

Acknowledgements

I would like to thank my supervisor, Dr. Neil Goulty for his enthusiasm, encouragement and support, and many fruitful discussions during the last three years. I trust he finds this a useful addition to his shelf.

Thanks also to all those in the department who have helped make this work possible, in particular Ed Kragh, Miles Leggett and Salim Al-Rawahy for all their help with computing and ideas, and Mike Findlay whose programs made life much simpler. Thanks go also to Seres a/s of Trondheim for the Groningen dataset, and to the British Coal Opencast Executive for providing boreholes.

I thank Shell UK for funding this postgraduate study, and especially Dr. Mike Bacon for his interest. I thank Dr. Phil Christie and Dr. Roger Long for their constructive comments during the examination of this thesis.

I would like to thank all my parents for their support, and especially my father, Rex Rowbotham who first told me I would do a PhD over 10 years ago.

Thanks to all who have made my stay in Durham fun; Trish, Miles, Danny, Salim, Ed, Dave, Chris, Neville, Richard, Paul, Jane, and the rest in the department, all in Gradsoc Boat Club and all up at Trevs. I am indebted to Joyce and George for their wonderful hospitality and kindness in providing such a warm home to escape to after a late night on the computers.

Final thanks go to Helen for invaluable support, and for keeping me sane over the phone. Without her patience, understanding and companionship, this work would have been much less enjoyable, and much more of a task.

A little inaccuracy sometimes saves tons of explanation.
Saki, The Square Egg

Contents

Table of Figures

Chapter I

Introduction

1.1 Synopsis

This thesis concerns advances made in processing techniques for application to crosshole seismic reflection surveys. The work presented in this thesis is part of the ongoing research into borehole seismic methods at the University of Durham. My own work reported here is partly refinement of methods previously established by other members of the research group, and partly development of new processing techniques to enhance the quality of the final image.

Chapter I gives a review of the research performed at the University of Durham, and elsewhere, into the use of borehole seismics as an imaging tool, and discusses some developments in crosshole acquisition. The acquisition of the datasets used in this study is described in Chapter II; the basic processing scheme, as previously developed by Findlay (1991) and Findlay et al. (1991), and processing tools devised by other researchers are discussed in Chapter III.

In Chapter IV, I present a critique of the imaging capability of crosshole reflection surveying, with suggestions as to how to ensure optimal imaging of the target zone. Chapter V contains two new processing techniques devised to overcome problems encountered in applying basic processing procedures and to enhance the imaging potential of crosshole reflection seismics.

The results of processing three different datasets are presented in Chapters VI, VII and VIII, and a critical discussion is given in each case. Conclusions are drawn, and suggestions for future work are made, in Chapter IX.

1.2 Borehole seismic methods

1.2.1 Check-shot and VSP

The first borehole seismic method to be developed in the oil industry was the check-shot survey for the purpose of calibrating velocity measurements obtained from borehole sonic logs. A receiver is clamped down the hole at several locations, and a shot set off adjacent to the top of the hole. The traveltimes of the direct arrivals provide corrections for the drift of the sonic curves. An extension of this is Vertical Seismic Profiling (VSP) (e.g. Cassell 1984). Again a downhole receiver is clamped and a surface air-gun positioned away from the rig so that no transmission occurs down the casing. In the land case, a vibrator or a source (water-gun or air-gun) in a mud-pit is used. Zero-offset VSP is principally used for identifying the primary and multiple reflections on surface seismic sections passing through the borehole and for designing deconvolution operators for surface seismics. Although zero-offset VSP does provide some imaging of structure around the borehole, the development of Offset VSP (e.g. Dillon and Thomson 1984) greatly increased the imaging potential of the method with coverage extending out from the borehole up to half the lateral separation of the source and borehole. The imaging potential has been further increased with walkaway, or multi-offset VSP surveys. Jackson et al. (1989), Kragh (1990) and Kragh et al. (1991) have developed the Reverse Multi-Offset VSP method (source in the borehole - line of geophones at multiple offsets at the surface) for shallow exploration, and the technique has been extended to 3-D by using an areal spread of geophones (Jackson et al. 1989). Present developments include novel acquisition geometry such as would be required for horizontal wells and the use of the drill bit energy as a seismic source for imaging (Rector and Marion 1991).

VSP surveys provide greater resolution of the subsurface than conventional surface seismic surveys, since the arrivals have to pass only once through the highly attenuating near-surface layers which absorb relatively more of the higher frequencies.

1.2.2 Crosshole tomography and continuity logging

Many of the recent advances in crosshole techniques have been due to the development of non-destructive sources (§1.3.1) and have been driven by the need for enhanced definition of oil producing reservoirs. Crosshole tomography, the best known crosshole technique, has found uses in monitoring enhanced oil recovery processes such as steam and CO_2 flooding (Macrides et al. 1988, Justice et al. 1989, Justice et al. 1993), fissure detection in granite (Wong et al. 1983) and has been used also in an attempt to delineate coal strata (Goulty et al. 1990). Both traveltime and amplitude information (Leggett 1992, Leggett et al. 1993) have been used to tomographically reconstruct the interborehole velocity function and the differential attenuation, respectively. Goulty (1993) gives an excellent review of applications of tomography in mining and engineering.

A further use of crosshole surveys is continuity logging of the interborehole stratigraphy (Zhong and Worthington 1992). In this, tube wave interaction at geological interfaces (Albright and Johnson 1990) causes conversion to horizontally travelling P-waves which, upon arrival at the receiver borehole, convert back to tube waves. Should the geological strata be broken by a fault, the path of communication between the two boreholes will be cut and therefore the analysis of tube waves can be used to provide a simple crosshole continuity log.

1.2.3 Crosshole reflection imaging

In addition to using the information carried by the direct arrivals, the reflected wavefield has been used for imaging (e.g. Findlay et al. 1991, Lazaratos et al. 1992). The obvious advantage of crosshole seismic reflection processing over tomography is that the coverage is not restricted to zones above the base of the source and receiver arrays through which the direct arrivals pass. Also, tomography fails to resolve discontinuities in the velocity field, and provides an estimate of only the low frequency components of the velocity distribution. Crosshole reflections must be used for imaging discontinuities in the velocity field, and for imaging at or below the base of the holes. Crosshole tomography and reflection imaging therefore provide complementary images.

Crosshole reflection surveys provide yet higher resolution than VSP surveys, as the transmitted energy does not have to pass through the attenuating near-surface layers at all. Compared with conventional surface seismics, the higher resolution of the subsurface structure achievable is of the order of 1 metre as opposed to ~10-20 metres (Harris et al. 1992). Resolution on this scale is critical for the mapping of thin beds, for site investigations involving the location and definition of faults or old mine workings, and for assessing the homogeneity of rock type. A second advantage over surface seismics is that crosshole seismic data contain a wealth of multiple angle and multiple direction (from above and below) views on target reflectors compared to the surface seismics normal-incidence interrogation of beds from above only.

1.2.4 Crosshole reflection processing

The processing of crosshole data for reflection imaging can be very similar to the standard VSP processing sequence, consisting of wavefield separation, deconvolution and mapping (Hardage 1985), though some serious differences exist (§3.4.1). Also of note is the difference in spatial imaging coverage for both types of survey (§4.1).

Several workers have discussed the peculiarities of the crosshole processing scheme. In the pre-imaging stage, Pratt and Goulty (1991), Stewart and Marchisio (1991) and Rector et al. (1992a) have advocated the use of resorting of the data into alternative domains to facilitate the separation of the reflected wavefield by multichannel filtering.

For imaging, several schemes have been suggested. One of the first methods to be used was the VSP-CDP mapping algorithm (Wyatt and Wyatt 1984), so called because it transforms the data from VSP coordinates (depth of source or receiver, traveltime) to surface seismic coordinates (lateral position and depth or vertical traveltime). The method involves raytracing to find the reflection point locus of each source and receiver pair (§4.1), and assigning to each point the amplitude on the recorded trace at the corresponding traveltime. The process is therefore not a migration method and will not collapse Fresnel zones.

Limited aperture Kirchhoff depth migration schemes, such as the one used in this work (§3.5, §5.1), have been developed for crosshole reflection imaging (Findlay et al. 1991), allowing a range of dips to be considered at each image point (§4.2) centred around the local expected dip of strata. Finite difference methods are also common (e.g. Balch et al. 1991). A joint migration/inversion scheme has been advocated by Beydoun et al. (1989) to produce the first published crosshole migrated images. The first operation in this scheme is similar to a Kirchhoff depth migration, with inversion consisting of applying a damped correction to the intermediate images to optimally deconvolve source-receiver effects and decouple parameters.

Frequency domain wave-equation imaging methods, whereby the time-reversed wavefields are propagated through the section, combined with traveltime tomography has been implemented for crosshole processing (Pratt and Goulty 1991). By only using a limited number of frequency components, the computing expense of these methods is reduced. This is appropriate since there is theoretical data redundancy in wide-aperture crosshole surveys that allows useful images to be formed from a single frequency component (Pratt and Worthington 1990).

1.3 Developments in crosshole acquisition

1.3.1 Borehole sources

A comparison of various candidate borehole sources for crosshole seismology and reversed vertical seismic profiling (RVSP) has been performed by Chen, Eriksen and Miller (1990). The sources were explosive charges, a perforating gun, an air gun, and a water gun. Although no visible damage was produced by the sources, some deterioration of the cement bond between the formation and the casing was indicated. Winbow (1991) has performed theoretical modelling to compare the performance of downhole seismic sources.

For the Coal Measures surveys performed by the Research Group at the University of Durham, single detonators were used, sometimes boosted by 25g of gelignite (Findlay et al. 1991). These were found to give good shot repeatability and an adequate bandwidth (100-700Hz) for the purposes of the surveys (borehole separation ~ 30-60m). Detonators have also been used by other workers from Imperial College (Zhong and Worthington 1992, Williamson et al. 1993, Sams et al. 1993), giving a major frequency content above 1kHz for borehole separation of 25m. Other larger scale surveys have been performed with explosive sources (e.g. Macrides et al. 1988, Chen, Zimmerman and Tugnait 1990). Crosshole seismology research has proliferated in recent years, but Geyer (1993) has described a survey performed in 1961 using a Schlumberger 48-gun casing-perforator tool as seismic source. Acquisition time is slow using explosive sources since the source assembly has to be raised to the surface between each shot for priming, and there is the possibility of damage to the borehole walls, even though none has been reported in the literature.

Another source type developed is the sparker source (e.g. Baria et al. 1989). Its main advantages are described as reliability, repeatability, and the production of an impulse wavelet of short duration which is free of bubble oscillation. Successive shots can be fired without having to bring the sparker to the surface, a distinct advantage over explosive sources in terms of acquisition time.

A weight-drop source has been used to produce a significant level of shear waves (Beydoun et al. 1989). The source is reported to generate fewer tube waves than those from a sparker source, it has a shot repeatability of two shots per minute, and probably does not harm the borehole.

The source that has found widest acceptance within the geophysical community in recent years is the piezo-electric cylindrical bender system (Wong et al. 1983, Balogh et al. 1988). In a similar vein, a magnetostrictive transducer has also been used (Albright and Johnson 1990). The piezo-electric transducer can be used as either a seismic source or a seismic detector, and is described in more detail in §2.1.1. The source produces signals of bandwidth 100-2000kHz over borehole separation of 330 feet at 1000 feet depth. The system has been developed to record 'on-the-fly', i.e. with source moving continuously at approximately 500 feet per hour, and receiving trigger signals from the control systems at regular depth intervals. The advantage of this mode of operation is the speed with which extremely large datasets can be acquired (e.g. 37 000 traces acquired in 40 hours as described by Harris et al. (1992)). The source has proved to be reliable in the most exacting of field conditions (depths > 10 000 feet), and of high enough frequency to allow high-resolution reflection imaging.

1.3.2 Borehole receivers

Hydrophones have been the standard receivers for shallow boreholes. They are ideally suited for detecting the pressure variations caused in the fluid-filled boreholes by the arriving energy packets. Their reliability has been enhanced by many years of use in the marine seismic environment, and hence their characteristics are well documented.

Three component borehole geophones have also been tested (Beydoun et al. 1989, Beattie 1990, Emeleus 1993). Their potential for separating out P and S-wave arrivals makes them an attractive receiver, although the necessity of aligning them and clamping them against the borehole wall greatly increases the acquisition time. The cylindrical piezo-electric bender transducer used as a source (§1.3.1) is also used as a detector.

Chapter II

Acquisition

Three types of crosshole seismic datasets have been used in this study. The acquisition of each will be discussed separately.

2.1 Groningen

Data were acquired by Seres a/s of Trondheim using two wells at the Scheemderzwaag site in the Groningen gas field in the north of the Netherlands over the period 19-24 November 1990 (Vaage and Ziolkowski 1992). This site normally produces gas, but gas production had been temporarily stopped in order to carry out routine maintenance work on the wells. Measurements made were:

- Logging of the wells by Schlumberger, using the array sonic tool.
- Crosshole measurements using the Stanford University piezo-electric bender source and hydrophone receiver array.
- Crosshole measurements using the READ Well Services mud-gun source and three-component geophone array.

The crosshole data acquired using the piezoelectric cylindrical bender system of Stanford University, deployed by Jerry Harris of Stanford, were made available for this project.

2.1.1 The Stanford crosshole seismic acquisition system

The Stanford piezoelectric system has been fully documented in the literature (Balogh et al. 1988, Harris et al. 1992). A schematic outline of the system is shown in Figure 2.1. The system comprises a three-element piezoelectric downhole source, a nine-level hydrophone and two logging trucks with associated surface equipment and instruments to control the downhole tools.

The source consists of three active elements, symmetrically placed into the downhole tool to form a mass balanced downhole source. Two banks of power

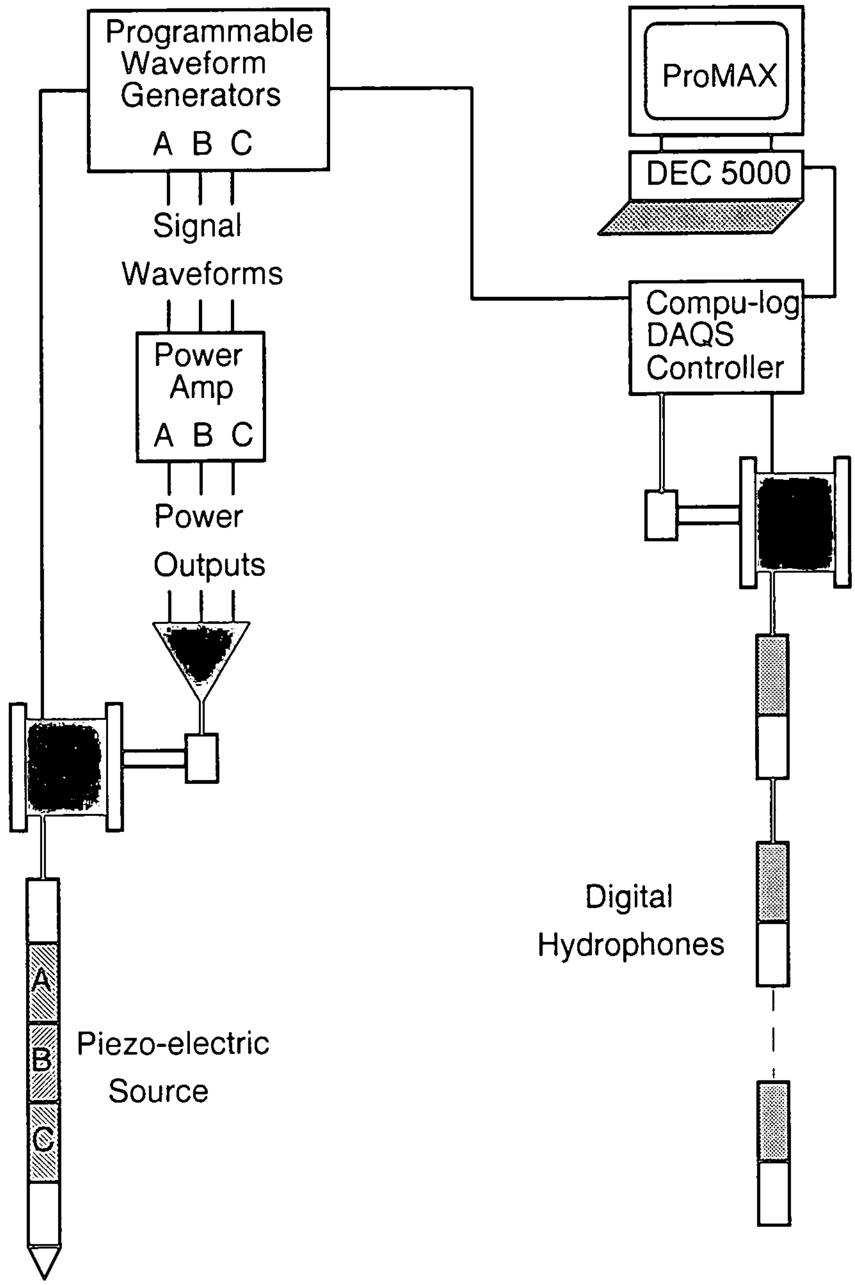

Figure 2.1 Schematic outline of the Stanford University piezo-electric borehole seismic acquisition system (from Harris et al. 1992).

transformers are mounted above and below the active transducers for symmetry. The balance is intended to reduce spurious modes of structural vibration, thus creating more radiation from the desired monopole mode. The elements may be driven as three independent sources or as a single unit for increased coherent output. This design is a slight modification on the original version built for Standard Oil in 1985-86. The far-field wavelet radiated by this source is proportional to the current drawn. Source signatures are generated by three 12-

bit D-to-A phase-coherent waveform generators. Arbitrarily defined waveforms including sweeps, pulses and pulse sequences can be used. For this experiment data were recorded in both sweep and pulse mode. All data gathered with sweeps were recorded uncorrelated, correlation being performed after the completion of the experiment. The source is powered by a three-channel 24kVA linear power amplifier. The power is delivered via 12 000 feet of 7-conductor armoured wireline.

The receiver system consists of a nine-level array of OAS deep ocean hydrophones, arranged at 3m intervals for this experiment. Each level is independently digitised downhole to sixteen bits of resolution. Unfortunately, only four of the nine channels were functional at the site surface. After lowering into the well, two further channels ceased to function. The two remaining channels, spaced 3m apart, were operational for the duration of the experiment. A surface computer, located in the source truck, provides control of recording parameters - sampling rate, downhole analogue and digital gain, vertical stacking depth, and high and low pass filter settings. The hydrophones are interfaced to the surface via a telemetry sonde that controls communications and data transfer. Data are stacked downhole in order to reduce transmission throughput. Though only four conductors are required, the entire system is run on 17 000 feet of standard seven-conductor wireline. Recording parameters, including both downhole source and receiver wireline depths which are electronically monitored, are recorded to the SEG-Y header along with the trace data.

2.1.2 The Scheemderzwaag experiment

The wells selected for the experiment deviated away from each other, giving a separation of 140m at the surface to 298m at 2350m depth, the deepest accessible point in the source well. The lateral position of each receiver relative to the source position at 2350m depth was calculated from x,y,z data provided for both wells at 25m depth intervals over the zone of interest (Appendix A). 2-D geometry was assumed since the deviations in x and y of the receiver well were approximately linear over the depth range of the gather.

In all, over 600 records were obtained. These included pulse tests, an experiment to determine the transmission characteristics as a function of the formation parameters by positioning the source and one of the hydrophones at the same depth, an experiment to study the variation of amplitude with distance, and six common source and common receiver gathers. These gathers had the following parameters:

Source depth	Receiver depth	Spatial interval	No. of Records
2330-2350m	2453m	1m	21
2330-2350m	2456m	1m	21
2331-2350m	2503m	1m	20
2331-2350m	2506m	1m	20
2350m	2507-2538m	1m	32
2350m	2233-2606m	3m	125*

Of these, the surveys with 1m spacing were found to have limited coverage, and to be affected by tube waves. In consequence, the common source gather with 125 records from 2350m depth was used for crosshole seismic reflection imaging (§6.3). For a sketch of the positions of the source and receivers relative to the geological cross section see Figure 6.1.

Following completion of the processing, I was informed that cable stretch had occurred during the course of the experiment. The shot gather used for imaging (§6.3) was the penultimate experiment carried out before the equipment was retrieved, and the adjustments needed were therefore the total stretching in the cables, 3m and 6m for the source and receiver positions, respectively. However, these corrections were considered to be of little value within the context of the aim of demonstrating the crosshole reflection imaging technique on a high frequency dataset from a producing field and have not been applied.

* Some irregular spacing occurred in the middle of this gather i.e. receivers at ...2416m, 2419m, 2423m, 2426m,... depth.

2.2 Physical Model

Physical model datasets designed to represent 'pre-flood' and 'post-flood' stages in an EOR process were acquired in 1989 by N.R. Goulty using Durham University's physical modelling system (Sharp et al. 1985). The models were made of seven layers of epoxy resin, representing strata of different densities and seismic velocities. A channel feature and a fault have been modelled by shaping the interfaces between the layers (Figure 2.2).

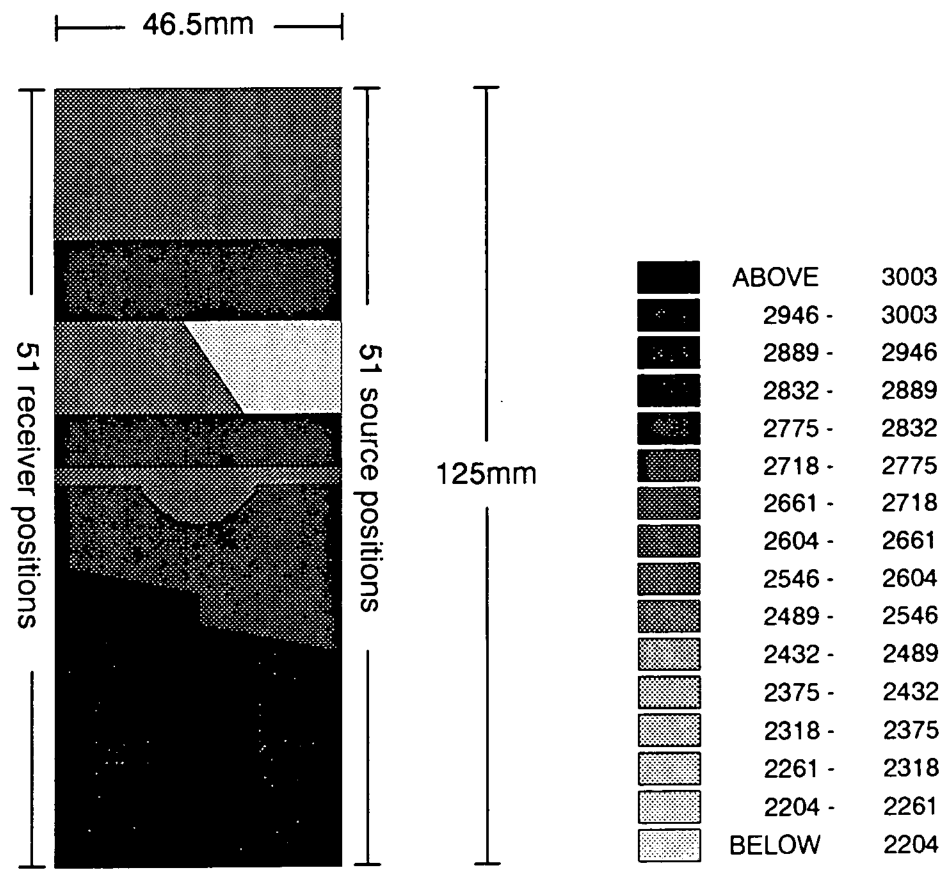

Figure 2.2 Acquisition geometry and velocities of the epoxy resin layers for post-flood. Pre-flood is identical except for absence of low velocity flood zone.

Piezo-electric transducers were used as source and receiver, and were positioned at intervals of 2.5mm over a distance of 125mm down the sides of the model, which was 46.5mm wide. The received signal had a bandwidth of 150-700kHz. Upon scaling all dimensions by a factor of 1000 to represent the case for real data (i.e. mm to m, ms to s), this corresponds to a 150-700Hz bandwidth with a borehole separation of approximately 50m. The reader is referred to Leggett et al. (1993) for a complementary description of the acquisition of these data.

2.2.1 The model

The models were comprised of seven layers made of five different epoxy resin mixtures. In keeping with geological realism, there is an overall increase in velocity with depth (although low velocity zones are present), the deepest interface is faulted, and one layer contains a channel feature. To ensure that the pre-flood and post-flood models were identical apart from the flood zone, they were made together in one solid block in the same mould. After the pouring of each layer, the upper surface was machined off before pouring the next layer. The models can therefore be regarded as identical to within 0.025mm, the tolerance of the milling machine. After the reservoir layer had been poured and set, it was machined to cut out the 'flood zone' over the half of the block which was to form the post-flood model, but left intact for the other half. A different epoxy mixture was then poured in to represent the flood, and two further layers added on top across both halves of the block. The complete block was then cut in two, separating the pre-flood and post-flood models.

2.2.2 The source and receiver

The source and receiver used were piezo-electric transducers. Sketches of their dimensions and nominal positions relative to the model edges are shown in Figure 2.3.

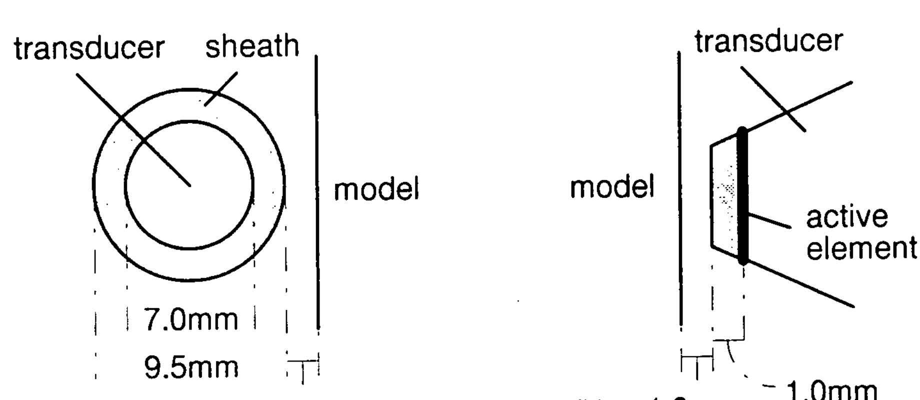

Figure 2.3 Geometry of the (a) source and (b) receiver transducers.

From this figure, it can be seen that the source signal is generated on a cylindrical surface of radius 3.5mm. Assuming that the radiation pattern from the source has cylindrical symmetry, it can be regarded as a 2-D point source in the centre of the transducer. For this assumption it is necessary to make a static shift to the data equivalent to the time taken to travel 3.5mm in water, since at time zero the wavefront is on the surface of the transducer. This shift amounted to 9.54 samples in the pre-flood case where the velocity of water was found to be 1467m/s and 9.52 samples for the post-flood with a water velocity of 1477m/s (Leggett 1992). The transducer was enclosed within a sheath of thickness 1.25mm, and the clearance between the sheath and the model was 1.25mm, the centre of the transducer being 6mm from the model.

The tip of the receiver probe was about 1mm from the model and the active area of the receiver transducer was guessed to be 1mm below the surface of the transducer, making a static correction for the receiver of 2mm. Assuming a point source, the total offset from the model of source and receiver was therefore 8 mm. To shift the source and receiver perpendicularly so as to be touching the model would require a reduction in traveltime by 21.81/21.66 samples (pre/post-flood), giving a total static shift (including the point source shift) of 12.27/12.14 samples. Static shifts are discussed further in §7.5.1.

The positions of the transducers relative to the model could be in error by up to 1mm, as could their positions relative to each other. The receiver transducer was moved manually between runs (accuracy within 0.5mm) and for each receiver position the source was moved automatically with a nominal accuracy of 0.01mm.

2.2.3 The data

The data were recorded in February 1989 onto two SEG-Y format, 1600 bpi half-inch tapes. There were 2048 samples per trace with a sample interval of 0.25 μs, and all the data were acquired with 16-fold vertical stack.

Tape TNK903 contains data from the pre-flood model. There are 55 sets (ids) of 51 traces.

- Ids 1 and 2 are in water only with the receiver in the middle of the array (depth 62.5mm) and the source moving from depths -62.5→62.5mm and 62.5→187.5mm, respectively.
- Ids 3-53 are 51 common receiver gathers with the model in place. Position 1 of both sources and receivers was at depth 125.0mm, with position 51 at 0.0mm.
- Ids 54 and 55 are in water only with the receiver at either end, and the identical source positions as for ids 3-53.

Tape TNK904 contains data from the post-flood model. There are 56 ids of 51 traces.

● Ids 1 and 2 are identical to ids 54 and 55 on the pre-flood model tape (though note that the tapes were recorded on different days and thus the water velocity will have altered).

● Ids 3-53 are 51 common receiver gathers with the model in place.

● Id 54 is with water only and the receiver 25.5 mm from the centre (recorded inadvertently).

● Ids 55 and 56 are identical to ids 1 and 2 on the pre-flood model tape (though again note the change in water velocity).

2.3 Coal Measures

Both Coal Measures datasets used in this study were acquired and originally processed by M.J. Findlay; hence full details of the acquisition of these data can be found in Findlay (1991). Both datasets are from the Lowther South opencast exploration site in Yorkshire, between boreholes 3438-3437 and 3500-3496 (sources in first named borehole). Findlay refers to these as surveys C (between boreholes III and II) and E (between boreholes IV and V). The separation of boreholes 3438-3437 was 37.1m, and that of 3500-3496 was 32.0m.

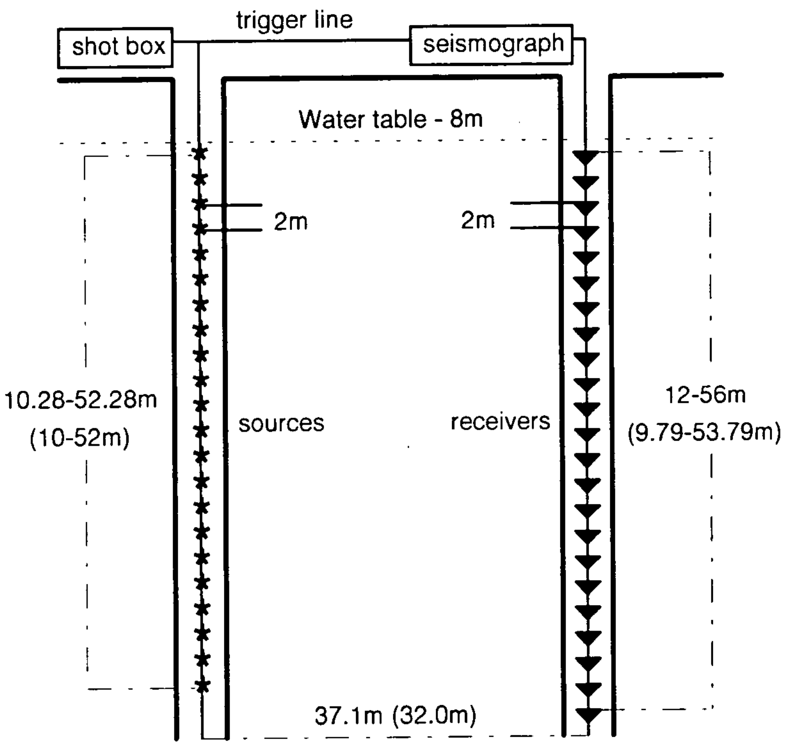

Figure 2.4 Geometry of surveys 3438-3437 and 3500-3496. Dimensions in brackets are for survey 3500-3496.

The depth range occupied by both sources and receivers was restricted by the water table at 8m, and by blockages in the uncased boreholes close to the worked coal seam at 50m depth. Single electrical (no. 8 type) detonators were used as sources, spaced 2m apart at 22 locations in boreholes 3438/3500. Findlay (1991) noted that 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 conditions required for the charge scaling law proposed by Ziolkowski et al. (1980) do not hold for small-sized charges fired in boreholes. A wire was taped around the end of the detonator and attached to the trigger line; the breaking of this wire on detonation triggered the recording system. The charge was positioned at the end of a 4cm diameter hollow steel tube (a section of scaffolding pole) which was 40cm long. The main purpose of this tubing was protection of the firing and triggering leads, and to provide sufficient weight to allow the charge to be raised and lowered easily.

Hydrophone receivers, also 2m apart, were deployed at 23 locations in boreholes 3437/3496. The data were recorded using an EG&G Geometrics ES2401 seismograph. The seismic signals had a bandwidth of 100-700Hz with a peak at about 200Hz.

Boreholes 3438 and 3437 were two of three collinear boreholes. In addition to crosshole surveying, reverse vertical seismic profiles (RVSPs - hole-to-surface) were shot in each borehole to give continuous coverage (Kragh 1990, Kragh et al. 1991). These RVSPs were shot using explosive charges downhole and a line of 24 geophones at the surface in the plane of the section. Hole-to-surface surveys were not performed in boreholes 3500 and 3496 due to borehole collapse following acquisition of the crosshole data.

Chapter III

Basic Processing

The processing tools described in this chapter are established methods for processing crosshole data, devised by Findlay et al. (1991) and other researchers (Kragh et al. 1991, Pratt and Goulty 1991). Their effectiveness is discussed in relation to the datasets used in this study in Chapters VI, VII and VIII.

3.1 Introduction

In processing crosshole data, we can extract several branches of information about the physical properties of the earth's subsurface, such as velocity, attenuation and reflectivity. By considering what information is required, we can design an optimum processing scheme to enhance the result.

For crosshole seismic reflection processing, there are advantages in changing the shape of the effective wavelet in the data. The recorded reflected seismograms can be thought of as the convolution of the source signature with the impulse response of the subsurface and recording system. The first onset of each reflected arrival therefore corresponds to the traveltime along the reflected raypath. For reflection imaging, it would therefore be desirable to shape the resident wavelet to zero phase, with peak amplitude at the onset time of each arrival.

Crosshole reflection imaging will also require the removal of all non-primary reflected arrival modes, whether P or S direct wave, mode conversions, head waves, tube waves or multiples. The high energy P direct arrivals can be muted, whereas methods such as median filtering or velocity (*f-k*) filtering are necessary for other arrivals. Separation between up and downgoing reflections is also necessary.

Since a depth section (i.e. a physical cross section of horizontal distance versus depth) is easier to interpret than the recorded section of receiver position against time, a scheme to migrate the reflected wavefield is then utilised. These will be the main requirements of a crosshole reflection processing scheme - waveshaping deconvolution, direct arrival suppression, wavefield separation and migration (Figure 3.1). These were discussed by Findlay et al. (1991), and as this study follows on from his work, the basic processing steps are similar. These steps are also interchangeable in some respects - some of the criteria governing the order are discussed in §3.6.

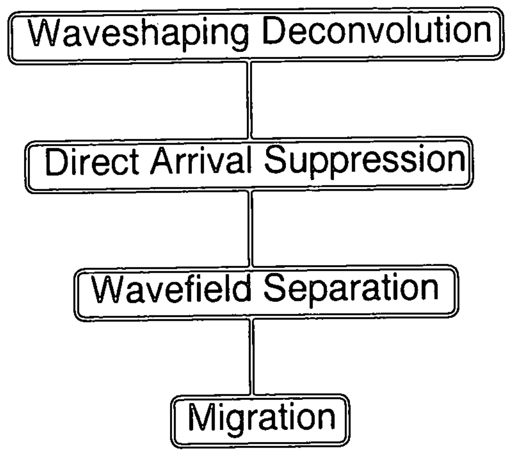

Figure 3.1 The basic crosshole reflection processing sequence.

3.2 Waveshaping Deconvolution

Most of the following methods use the same philosophy of designing a Wiener-shaping deconvolution filter (Robinson and Treitel 1985) with an estimate of the resident wavelet as the input wavelet and a zero-phase Butterworth wavelet (e.g. Sheriff and Geldart 1983) as a desired output. The output wavelet is specified in the frequency domain to correspond to the useful signal bandwidth with tapering at the limits. It can be thought of as a band-limited Dirac delta function. Alternatively, the zero-phase output wavelet can be specified with the same amplitude spectrum as the input wavelet.

3.2.1 Wavelet estimate by assumption of minimum phase wavelet

This method was used by Kragh et al. (1991) and Findlay et al. (1991) for borehole reflection processing. An estimate of the source wavelet is obtained by taking the autocorrelation function of the wavelet to be the sum of the autocorrelation functions of all the traces (or a selection of traces). This assumes that the reflectivity is white and stationary. The minimum phase assumption is then used to obtain a minimum phase input wavelet (Robinson and Treitel 1985).

3.2.2 Wavelet estimate by aligning direct arrivals

The direct arrivals at the receivers can be used to design a deconvolution filter for the reflected arrivals (Kragh et al. 1991). A single direct arrival is estimated by aligning the direct arrival energy of all traces (or several traces, depending on data quality) and stacking the traces. Alternatively a single-trace direct arrival can be used.

3.2.3 Amplitude spectrum of zero-phase output wavelet = input wavelet

Should the previous two methods not produce the desired shaping of the data because of peculiarities in the input wavelet amplitude spectrum, the amplitude spectrum of the desired output wavelet can be equated to that of the input wavelet. However, the output waveform will not be as tidy as for the Butterworth wavelet, as its amplitude spectrum will be further from the ideal of the Dirac delta function. This method was used by Leggett et al. (1993) for deconvolving the direct arrivals through the physical model, prior to picking traveltimes for tomography. One situation where this method might be necessary is where there is a notch in the amplitude spectrum of the input wavelet (see §7.3.1). The standard method to deal with this problem would be to add white noise (Hatton et al. 1988).

3.2.4 Common ray angle gather

An alternative scheme for deconvolution has been proposed by Pratt and Goulty (1991), working on the physical model dataset. This method as applied to the data in this study is discussed fully in §7.3.3. The data are regathered into common ray angle gathers, this reorganisation of the data allowing scattered (i.e. reflected) and direct arrivals to be discriminated on the basis of time moveout. For each trace that was to be operated on, 11 common ray-angle gathers were selected. These were the gathers from the given ray angle and from the five adjacent ray angles on either side. From each of these gathers, five traces were selected: the traces from the given receiver location and two traces on either side. Thus a total of 55 input traces were used for each output trace (except at the edges of the survey). The direct arrival first breaks were aligned and averaged by means of a running mean filter. This then gave an estimate of the direct wavefield for all 2601 traces. This process could also be performed using S wave arrival times to estimate the S-wave direct wavefield.

This direct wavefield estimate provides a robust method of deconvolution (Figure 3.2). A Wiener shaping filter was computed from the estimated direct arrival wavelet on each trace, and applied to the corresponding recorded trace. The desired output was a minimum phase wavelet with the same amplitude spectrum as the input wavelet.

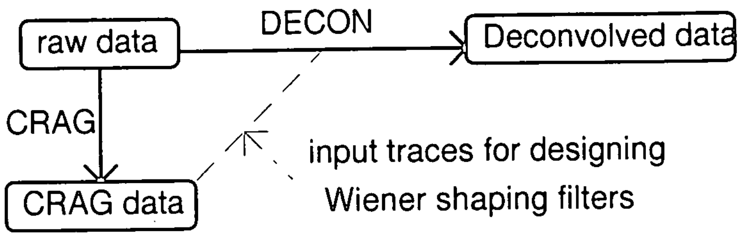

Figure 3.2 The common ray angle gather processing scheme.

Findlay (1991) observed, and it is my experience, that this method is unsuitable for the Coal Measures datasets, which have fewer sources and receivers than the physical model dataset, because of the end effects on the averaging process for the shallowest and deepest source and receiver positions. Also, trace-to-trace variations in the direct arrival waveforms are greater, possibly due to peg-leg multiples.

3.3 Direct Arrival Suppression

3.3.1 Muting

For reflection imaging, it would be desirable to remove direct arrivals, which are of higher amplitude than the reflected arrivals. Hu and McMechan (1987) and Findlay et al. (1991) advocate the use of first break (transmitted) energy muting. Note that this will not remove slower S-wave direct arrivals, nor direct P-waves where the first arrivals are head waves. Also note that it is preferable to perform direct arrival suppression after waveshaping deconvolution. The direct arrival has then been compressed into a narrower wavelet, so the muting may be constrained to a narrower window, therefore corrupting less of the reflected arrivals.

The inherent problem with muting of the direct arrival is that, because the traveltime of the up/downgoing reflected arrival just above/below a reflecting interface is only slightly greater than the traveltime of the direct arrival, reflected energy will also be removed. This means that reflector coverage will be lost near the source and receiver boreholes.

3.3.2 First break estimation

To be able to mute, it is first necessary to estimate the direct arrival times at each receiver position. Findlay's automatic first break picking program within the **xhr** package (Appendix B) is an efficient way of doing this. First breaks are picked on a statistical basis, by locating the time sample where the recorded signal amplitude rises sufficiently above the rms amplitude of the background noise. Leggett et al. (1993) used a method of deconvolving the direct arrival to a zero-phase peak, and then picking the maximum amplitude in a window around the first arrival.

3.3.3 Median filtering

Any coherent arrival can be eliminated from seismic data by use of multichannel filters (Özdemir and Saatçilar 1990). Median filters work by aligning the arrival to be enhanced or removed, and performing a median filter across the traces whereby the median value of a time sample over an odd number of traces is placed at the time sample of the middle trace. Aligned events will be enhanced at the expense of dipping events. The estimate of the aligned arrival can then be subtracted from the original data. To prevent too much destruction of useful signal, the filter should be applied only within a time window specified about the arrival to be suppressed. High cut filtering follows median filtering to reduce the high frequency noise ('jitter') introduced by the filtering.

Although median filtering has been used in crosshole reflection processing (Rector et al. 1992a, Cai and Schuster 1993), the smaller number of source and receiver positions used (i.e. less data redundancy), and trace-to-trace variations (§3.2.4) in the datasets used in this study mean that the level of noise introduced by median filtering is unacceptable.

3.3.4 Common ray angle gather

An alternative scheme for direct arrival removal uses the CRAG method (§3.2.4) of Pratt and Goulty (1991). The CRAG gives an estimate of the direct wavefield for all 2601 traces. Following trace-by-trace deconvolution of the wavefield, the deconvolved total wavefield was used in exactly the same process

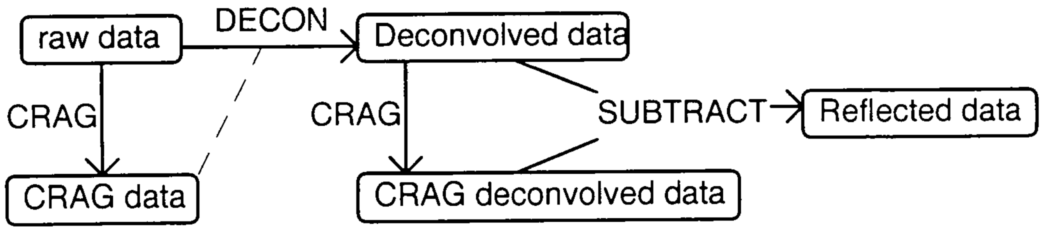

Figure 3.3 Scheme for direct wavefield removal using common ray angle gathers.

of estimating the direct wavefield described earlier. Once this new direct deconvolved wavefield was computed, it was subtracted from the deconvolved total wavefield to produce an estimate of the scattered wavefield (Figure 3.3).

It was realised that this scheme involving a second CRAG would tend to smooth out traces by involving more than 55 traces in producing a single trace, since each trace input to the second CRAG would already be influenced by 55 traces input to the initial CRAG. It was also unnecessarily cumbersome, in that an estimate of the deconvolved direct wavefield could be obtained by applying the same filters to the direct wavefield estimate as had been applied to the total raw wavefield (Figure 3.4). This dramatically reduced the computation time for this method since deconvolution is less CPU intensive than the CRAG method.

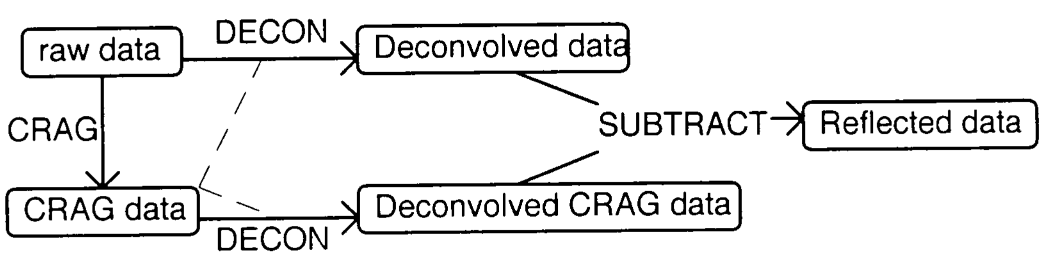

Figure 3.4 Improved scheme for direct wavefield removal using common ray angle gathers.

The reason for performing deconvolution before subtraction was so that any high frequency noise introduced due to slight mis-match would be within a compressed window. The method was significantly more successful at removing the direct wave arrival than the methods described above, and in doing so avoided all use of muting which had proven a problem in removing the desired reflected arrivals in previous methods. However, the caveats mentioned in §3.2.4 of end effects and trace-to-trace variations are pertinent again. A further advantage of this method is that it can be used to mute direct S-waves also, by repeating the CRAG with the traveltime picks of the S-waves instead. The success of this second CRAG will be dependent on the ease of S-wave arrival traveltime picking, since the arrival will be masked by P-wave reflections, and will vary in amplitude with angle of trajectory.

3.4 Wavefield Separation in the *f-k* domain

Both *f-k* velocity filtering and median filtering have been used in crosshole seismic wavefield separation. The first method is the one used by Kragh et al. (1991) and Findlay et al. (1991). The median filter and its associated problems have already been discussed with regard to direct arrival suppression (§3.2.4).

The two-dimensional Fourier transform (2-D FT) and its applications to seismic reflection data processing are well known (Embree et al. 1963, March and Bailey 1983). The computational use of *f-k* techniques has been greatly facilitated by the Fast Fourier Transform (FFT) (Cooley and Tukey 1965), which reduces dramatically the number of calculations necessary for transforming. For our purposes, the most interesting application of the 2-D FT is that of velocity filtering (Christie et al. 1983). Often referred to as pie-slice or *f-k* filtering, the technique has found wide application in VSP processing (Hardage 1985).

3.4.1 Differences between VSP and crosshole

Both crosshole gathers and VSP gathers contain several arrival modes which often intersect in the *z-t* domain, and are therefore difficult to separate using time domain filters. In the *f-k* domain, these arriving energy packets transform to events radiating from the origin. The slope of an event is equal to the apparent velocity with which the arrival passes the receiving array, and it can be deduced that intersecting arrivals in *z-t* have different apparent velocities and will plot along different lines in *f-k*. Attributing positive velocities to upgoing events it can be seen that upgoing and downgoing events will be separated into the right hand and left hand sides of the *f-k* plot in Figure 3.5, respectively. Filtering in the *f-k* domain, by putting to zero a wedge or pie slice of the spectrum, can therefore be used to discriminate between intersecting arrivals, and between up- and downgoing events.

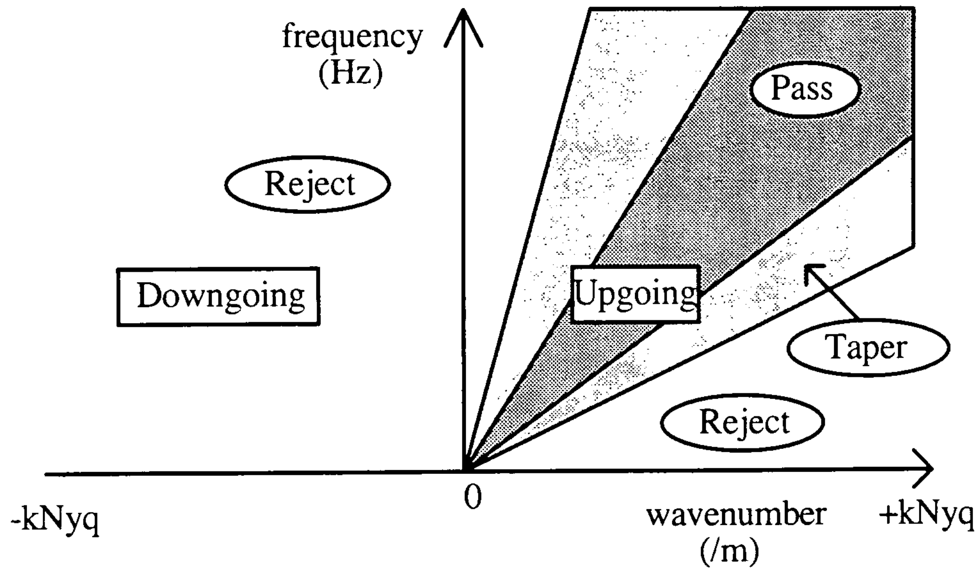

Figure 3.5 *F-k* plot showing the separation of upgoing and downgoing events, and a filter for passing only upgoing events.

For a particular interface, a wave arriving at it from above would 'see' it as the opposite polarity to that 'seen' by one from below, i.e. the impedance contrast would be positive for a downgoing wave and negative for an upgoing wave or vice versa. Therefore the reflected signals from above an interface (upgoing reflections) would be of opposite polarity to those below (downgoing). Separation of wavefields is a necessary pre-migration processing step since unless the up and downgoing events are migrated separately, there would be a net cancellation effect from combining their respective reflectivity amplitudes.

One important difference between VSP and crosshole wavefields is that, in the crosswell situation, many reflection events overlay the direct arrivals in *f-k* space (Figure 3.6), and wavefield separation by *f-k* filtering is difficult (Hardage 1992).

For a VSP wavefield, the direct and reflected events fall into opposite half spaces of the *f-k* domain, and effective *f-k* filters can be designed to suppress either the direct or the reflected wavefields. Note how it is not possible to perform a simple pass filter to separate direct and reflected arrivals for crosshole data in the *f-k* domain. As discussed previously (§3.3.1), this could be avoided by muting the direct arrival either pre- or post- wavefield separation.

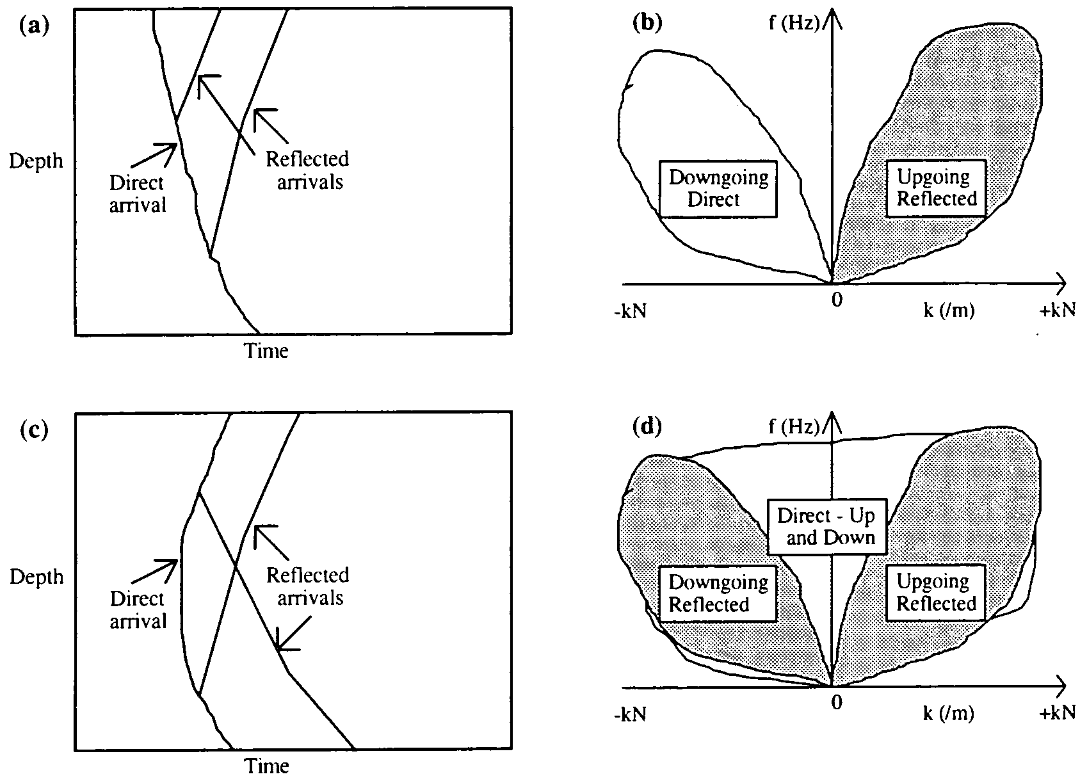

Figure 3.6 (a) VSP data in the z-t domain and (b) in the f-k domain. (c) Crosshole data in the z-t domain and (d) in the f-k domain.

3.4.2 Data preparation for f-k filtering

Data processing prior to transformation to the f-k domain and the design of the f-k filter are crucial for minimising the artefacts of the transform and filtering processes. March and Bailey (1983) have covered most of these factors; here attention will be brought to those which are of particular interest in crosshole processing.

One problem with digitally sampled data is that of undersampling and thus non-unique definition, especially in discrete Fourier space. If the temporal and spatial (i.e. source or receiver spacing) sampling intervals for the data are Δt and Δz, the temporal and spatial Nyquist frequencies are defined as

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

Before transforming the data to the f-k domain, the gather must be normalised so that each trace has equal energy. Large differences between adjacent trace amplitudes would cause ringing in the *f-k* spectra, hence degrading the wavefield separation process. The importance of this step is of course highly data dependent. Coal Measures data are affected by the different source coupling factors (Kragh et al. 1991, Findlay et al. 1991), and may also be affected by variable receiver coupling. The physical model data (§2.2.2) on the other hand have near constant source/receiver coupling, though some slight variation should still be corrected by normalisation.

The FFT algorithm requires that the data consist of 2^n samples, where n is an integer, in both space and time dimensions. Padding out the number of samples with zeroes up to the next integer power of 2 is a simple matter in the time direction. Where the number of traces are not an integer power of 2, padding out with zero traces up to the next power of 2 is performed. In fact these padding traces are highly advantageous for preventing wrap-around and so it is often desirable to pad out with zeroes by a further power of 2.

Prior to the *f-k* transformation, the data must be spatially and temporally tapered. The 2-D FT assumes that the dataset repeats to infinity along both *z* and *t* axes. This assumed periodicity is not a great problem in time, since both the start and tail of the traces can be tapered without loss of desired data. However, a greater problem exists in the spatial direction, where a large discontinuity exists between a data trace and a zero (padding) trace. As mentioned above, this will cause ringing in *f-k* unless a spatial taper is applied to reduce the amplitudes of the edge traces down to zero over three or four traces. It should also be noted that the same tapering requirement exists when transforming back from *f-k* space to *z-t*. With the data used in this project, the amplitude tapers off to zero before the temporal Nyquist frequency. However, sharp cut-offs exist at the positive and negative Nyquist wavenumber, and it is necessary to apply a taper to zero over the edge wavenumber values.

3.4.3 Filter design

The arrivals to be passed and those to be rejected must be identified in the data in both the *z-t* and *f-k* domains. The apparent velocities of the arrivals can then be calculated. Should these arrivals be low amplitude, some intuition in apparent velocity estimation is required on the part of the processor, based on knowledge of the formation velocities. A pie-slice filter is then designed with pass slope velocities around that of the useful arrival (Figure 3.5). For example, P-wave arrivals will have apparent velocities of between 1800m/s (a wave travelling vertically through a formation of velocity 1800m/s) and infinity (a wave travelling horizontally and therefore hitting all sources/receivers simultaneously). Depending on the geometry of the borehole seismic acquisition, the range of useful angles of reflected arrivals can be calculated.

The pie slice filter must not have sharp edges, since this would introduce an event at the velocity of the cut-off upon transforming back into *z-t*. A cosine taper is used by the **xhr** package (Appendix B) to smooth the edges of the pie-slice from the value of unity at the slope velocity down to zero at a velocity input by the user (Figure 3.5). For the same reason, it is prudent to select the edges of the velocity filter to lie along low amplitude channels in the data.

3.5 Migration

The migration process relocates reflection events to their true subsurface positions, and collapses diffractions to a point. Surface seismic surveys may be migrated post-stack on the assumption that the stack may be treated as a zero-offset section. However, this assumption is not valid for strong lateral velocity variations or for strong multiples and conflicting dips, for which the hyperbolic moveout assumption for stacking no longer holds (Yilmaz 1987). Full pre-stack migration, although preferable for overcoming the above problems, is less frequently employed because of the huge volumes of data and hence computing cost and time involved. Instead, pre-stack partial migration (synonymous with dip moveout, DMO) has been developed (Deregowski and Rocca 1981). In contrast, crosshole migration must be performed pre-stack as the source/receiver acquisition geometry precludes the equivalent of forming CMP gathers. Pre-stack migration is feasible for the small crosshole datasets used in this work, and stacking is performed post-migration when required.

Three mainstream migration methods have been developed, based on solutions to the scalar wave equation:- Kirchhoff diffraction stack, finite difference and *f-k* migration (Hood 1981, Yilmaz 1987). The first pre-computer technique used was semicircle superposition. Diffraction summation (diffraction stack) (Hagedoorn 1954) was the first migration scheme commercially available on a computer. A progression from this was Kirchhoff summation, whereby Kirchhoff integral theory, rather than the ray approximation used previously, was employed to make the summation consistent with the wave equation. This then included the effects of spherical spreading, obliquity and amplitude and phase corrections inherent in the depiction of the reflector as a series of closely spaced Huygens' secondary sources. Finite difference solutions to the scalar wave equation based on wavefield extrapolation and an imaging condition have been pioneered by Claerbout (e.g. Claerbout 1970, Claerbout 1971, Claerbout and Johnson 1971). A further set of methods are the frequency-wavenumber (*f*-*k*) methods such as those of Stolt (1978) and the phase shift method of Gazdag (1978). These methods were based originally on a constant velocity medium, and their use in heterogeneous media is not free from problems.

3.5.1 Why Kirchhoff?

Findlay (1991) has made a comparison between several migration algorithms for application to crosshole processing. One conclusion from this was that Kirchhoff methods were preferable to finite difference methods because they were faster and because of the extra control of image quality available during the migration process through the use of an imaging aperture. He also concluded that the diffraction stack technique did not give theoretically correct amplitudes or phases in the migrated section, the finite difference technique (as coded up from Sun and McMechan 1986) produced reverberations due to grid dispersion effects and the Kirchhoff algorithm was unsuitable as it implied that the reflectivity response depended on the image point position between the source and receiver boreholes (§3.5.3). He therefore recommended and employed the Generalised Kirchhoff migration (Dillon 1990), although he noted the similarity to the Generalised Radon Transform (Miller et al. 1987).

3.5.2 Development of Kirchhoff migration

Claerbout's (1971) imaging principle states that reflectors exist at points in the subsurface where the first arrival of the downgoing wave is time-coincident with the upgoing wave. This principle forms the basis of all migration schemes. In wave-equation migration, the scattered wavefield at the geophone array is extrapolated (in reverse time) back along its path of propagation. An image is formed where the extrapolated wavefield meets the direct arrival.

As previously mentioned, diffraction stack migration was originally based on ray tracing concepts and on the scalar diffraction theory of Huygens and Fresnel. French (1975) and Schneider (1978) introduced to the diffraction stack process the Kirchhoff integral solution to the wave equation. When used in conjunction with an imaging condition this becomes Kirchhoff wave-equation migration.

Schneider's form of the 3D Kirchhoff integral formula (Schneider's equation 5) was derived in terms of the free surface Green's function for the wave equation:

$$P(\mathbf{r},t) = \frac{1}{2\pi}\iint dA \frac{\cos\alpha}{|\mathbf{R}|c}\left[\frac{\partial}{\partial t}P(\mathbf{r}_0,t_0) + \frac{c}{|\mathbf{R}|}P(\mathbf{r}_0,t_0)\right] \tag{3.2}$$

In surface seismic applications, this relates the wavefield $P(\mathbf{r},t)$ observed on the plane $z=0$ to its value at a point $P(\mathbf{r}_0,t_0)$ in the earth's subsurface at an earlier time. $|\mathbf{R}|$ is the distance from $\mathbf{r}_0$ to $\mathbf{r}$, c is the velocity of the medium through which the wave has travelled and $\cos\alpha$ is the obliquity term. When thinking of crosshole applications, however, these definitions are inappropriate and instead we consider the observed wavefield at a position vector $\mathbf{r}$ related to its value at a position vector $\mathbf{r}_0$ at an earlier time.

As they stand, the first and second terms in the square brackets can be considered as the far-field and near-field terms, respectively. The near-field term is frequently dropped in seismic applications, giving the Rayleigh-Sommerfield diffraction formula of optics (Goodman 1968). The far-field term is also known as the high frequency approximation, since a higher frequency will mean a greater number of wavelengths in a set distance. Dropping the near-field term gives:

$$P(\mathbf{r},t) = \frac{1}{2\pi}\iint dA \frac{\cos\alpha}{|\mathbf{R}|c}\frac{\partial}{\partial t}P(\mathbf{r}_0,t_0) \tag{3.3}$$

The operations implied by equation (3.3) are simply weighting, scaling and phase shifting of data on a hyperboloid. Note that for surface seismics this hyperboloid will be symmetrical about a vertical line whereas for crosshole configurations the symmetry will be about the normal to the plane through the source and receiver positions. The term $\cos\alpha$ represents a directivity term which falls off from unit value at the apex of the hyperboloid down its flanks. By considering a reflecting surface as a series of closely spaced Huygens' secondary sources, and by analogy with the geometrical optical case of point apertures, the necessity of this term to deal with angular dependent amplitudes becomes apparent (Sheriff and Geldart 1983). The factor $\frac{1}{|\mathbf{R}|c}$ represents a true amplitude scaling factor and can be thought of as compensating for spherical spreading. Differentiation of the pressure function with respect to time, when examined in the frequency domain, represents a $\frac{\pi}{2}$ phase shift together with a linear high-frequency boost. This is commonly known as the Newman (1990) filter.

For 2-D this equation can be adapted by reducing the area integration to a line integral (Devey 1978), and thus cylindrical symmetry. This results in:

$$P(\mathbf{r},t)=\int dx \frac{\cos\alpha}{\sqrt{|\mathbf{R}|c}} \frac{\partial^{1/2}}{\partial t^{1/2}} P(\mathbf{r}_0,t_0) \tag{3.4}$$

The square root differentiation is not defined, except in the frequency domain where it represents a $\frac{\pi}{4}$ phase shift and a non-linear high frequency boost ($f^{1/2}$). It can be rewritten as:

$$\frac{\partial^{1/2}}{\partial t^{1/2}} P(\mathbf{r}_0,t_0)=\int_0^{\infty} dT\, F(T) P(\mathbf{r}_0,t_0) \tag{3.5a}$$

where $F(t)$ is the half differential operator such that

$$F(t)=\frac{d}{dt}\frac{1}{\sqrt{t}} \text{ for } t>0. \tag{3.5b}$$

Dillon (1988) has applied this integral to the VSP configuration with non-collinear source and receiver positions.

Using the quantities defined in Figure 3.7, the integral becomes:

$$P(\mathbf{r},t)=\int_{L_r} dL_r \frac{\cos\theta_r}{\sqrt{cR_r}} \int_0^{\infty} dT\, F(T) P\left(\mathbf{r}_0, T+\frac{(R_s+R_r)}{c}\right) \tag{3.6}$$

To convert the integral from 2-D (line-source and receiver, structure invariant perpendicular to the plane containing the boreholes) to 2.5-D (point-source and receiver, structure invariant perpendicular to the plane containing the boreholes), the integral is multiplied by the amplitude correction factor ($\sqrt{R_r+R_s}$). A further correction factor of ($\sqrt{R_s}$) is needed since we are using an imaging condition, and so must allow for the fact that the image points in crosshole are scatterers and their illumination (i.e. intensity of incident wavefield from source) depends on their distance from the source. The integral becomes:

$$P(\mathbf{r},t)=\int_{L_r} dL_r \cos\theta_r \sqrt{\frac{R_s(R_s+R_r)}{cR_r}}\, M\left(\mathbf{r}_0,\frac{(R_s+R_r)}{c}\right) \tag{3.7a}$$

$$\text{where } M\left(\mathbf{r}_0,\frac{(R_s+R_r)}{c}\right)=\int_0^{\infty} dT\, F(T) P\left(\mathbf{r}_0, T+\frac{(R_s+R_r)}{c}\right) \tag{3.7b}$$

Apart from scalar terms, this is equivalent to Dillon's (1990) equation (4), which forms the basis of his further comparison between Kirchhoff migration and acoustic generalized Radon transform (GRT) migration (Miller et al. 1987).

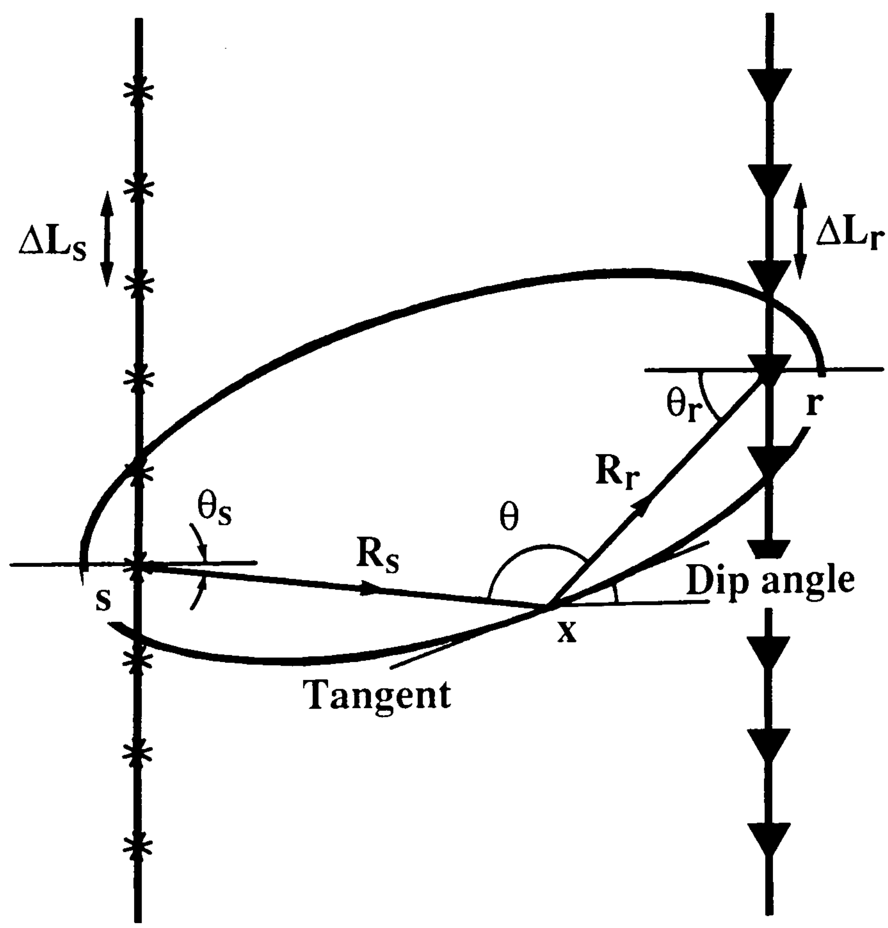

Figure 3.7 Elliptical isochron for the traveltime from a source at s to a scattering point x and on to a receiver at **r**.

Equation (3.7) may be approximated by a summation over the receiver array L_r. Resetting the equation in terms of reflectivity and dropping scalar terms (Dillon 1990 equation 5):

$$\hat{C}(\mathbf{x}) = \sum_{L_r} \Delta L_r \sqrt{\frac{R_s}{R_r}} \cos\theta_r \sqrt{R_s + R_r}\, M\left(\mathbf{r}, \frac{(R_s + R_r)}{c}\right) \tag{3.8}$$

In replacing the integration by a discrete summation, we have introduced the possibility of error if ΔL_r is not small compared with R_r. However, since the far-field approximation has already been made, this should not be a limitation. Note again that the summation in equation (3.8) is similar to a diffraction stack with an obliquity factor, a wavefront spreading term and the amplitude and phase corrections of the Newman filter being incorporated.

3.5.3 Generalised Kirchhoff migration

Findlay (1991) argued that equation (3.8) was clearly inadequate for the crosshole geometry, since the reflectivity would be larger where $R_s > R_r$ i.e. nearer the receiver borehole. An expression which was symmetrical with respect to an exchange of source and receiver arrays (i.e. the reciprocity principle) would be preferable.

Dillon's inspiration for his 1990 paper came from a desire to correctly image deviated well VSPs, where the source is maintained vertically above the down-hole geophone at each well station. However, correct wavefield extrapolation requires that the boundary conditions at the array of geophones satisfy the wave equation. Although a sufficient condition is to perform the survey with a single stationary source, a moving source deviated-well VSP fails to provide the boundary conditions for wave-equation migration. Dillon thus derives a generalized Kirchhoff (GK) migration algorithm from the exploding reflector model and compares this to the GRT migration scheme which was developed to handle any configuration of sources and geophones.

The 2D form of Dillon's (1990) GK migration summation operator is:

$$\hat{C}(\mathbf{x}) = \sum \left(\Delta L_r \sqrt{\frac{R_s}{R_r}} \cos\theta_r + \Delta L_s \sqrt{\frac{R_r}{R_s}} \cos\theta_s \right) \sqrt{R_s + R_r}\, M(\mathbf{s}, \mathbf{r}, t_s + t_r) \tag{3.9}$$

The theory behind this operator does not satisfy the wave equation, and it is necessary to remember this when considering the images produced by the operator. There is also an error term which is proportional to $\cos\phi_s - \cos\phi_r$ (the angles made by the incident and reflected rays to the normal to the surface at **x**), though this is very small if the zone of illumination extends beyond a few wavelengths from **x**, and the geophone array and source array are extensive enough to capture most of the scattered energy. The 2-D form of GRT migation is given as equation (3.10), and the only difference with GK migration is the half-angle weighting term. This would tend to discriminate against high angles of incidence.

$$\hat{C}(\mathbf{x}) = \sum \left(\Delta L_r \sqrt{\frac{R_s}{R_r}} \cos\theta_r + \Delta L_s \sqrt{\frac{R_r}{R_s}} \cos\theta_s \right) \cos^2 \theta/_2 \sqrt{R_s + R_r}\, M(\mathbf{s},\mathbf{r},t_s + t_r) \tag{3.10}$$

3.5.4 Implementation of migration

The program **kirchmig** was written by Findlay (1991) to carry out diffraction stack, Kirchhoff summation (equation (3.8)), GK summation (equation (3.9)) and GRT migration (equation (3.10)). This program has formed the basis of subsequent developments (§5.1). The program's raytracing is performed through a horizontal layered velocity model; different horizontal and vertical velocities can be defined to model anisotropic media. The migration plane is covered by a user-defined grid of image points. Each image point is considered in turn, and raytracing from each source position to the point and from each receiver position provides a total travel time and distance travelled for each source-receiver combination, together with the angles of take-off from source and receiver and incidence at the image point.

For the processing of the tank data, a second raytracing method was adapted from the method used by Leggett et al. (1993) for tomography. The program to perform this was modified from Cassel's (1982) curved-raytacing algorithm. A grid of cells ('boxels') is defined, each with an associated velocity. Refraction calculations using Snell's law therefore are performed at each cell boundary. As above, a grid of image points is considered in turn, and raytracing provides traveltime, distance travelled and angles. This method is discussed further in §7.5.4.

3.5.5 Estimation of the velocity model

To estimate the velocity field for the raytracing, one can use tomographic methods, velocities estimated from horizontal raypaths, and sonic logs and uphole shots. The method used for each of the three datasets is different (§6.3, §7.5.5, §8.1.2).

3.6 Factors influencing the order of the processing scheme

3.6.1 Source or receiver directivity

Should either the source or receiver display directivity, it would be preferable to wavefield separate the data into up or downgoing wavefields prior to the extraction of a minimum phase wavelet from the data (§3.2.1). In this way a separate filter can be designed for the up and for the downgoing wavefields.

3.6.2 Source and receiver depths

Where the source and receiver are at very different depths, the direct wavefield will be downgoing in the CSG or CRG whereas the reflected arrival will be upgoing or vice versa. In this case it would be better to suppress the direct arrival after wavefield separation, so as to preserve as much of the reflected wavefield as possible. In fact, *f-k* filtering will be an effective tool for direct arrival suppression, and further suppression may be unnecessary. When the source and receiver are at similar depths, however, the inclusion of the direct arrival can lead to ringing problems on application of *f-k* filters. A further salient point is that direct arrivals will contain higher frequencies than reflected arrivals which have travelled through more of the subsurface which preferentially attenuates higher frequencies.

3.6.3 Area to be imaged

Following on from the point above (§3.6.2) about *f-k* filtering being an effective tool for direct arrival suppression, this is especially true where reflections from the area of interest are upgoing on the receivers when the direct arrival is downgoing and vice versa. It is interesting to consider which parts of the section this is true for. For the area of section close to the upper half of the receiver array, the primary reflections from deeper sources are downgoing at the receivers whereas the direct waves are upgoing. Thus these wavefields could be separated in *f-k* space for each CSG. For the area close to the lower half of the receiver array, the reflections are upgoing whereas the direct waves are downgoing, and *f-k* filtering in CSGs is effective here too (see §7.4.2). By reciprocity, for imaging the section close to the source array, *f-k* filtering needs to be carried out in common receiver gathers (CRGs). Thus *f-k* filtering is good for removing the direct wave on seismograms where source and receiver are at very different depths.

3.6.4 Direct arrival suppression before or after deconvolution

It has been mentioned already that it is better to perform direct arrival suppression after waveshaping deconvolution (§3.3.1) since the direct energy has been compressed into a narrower window and therefore less reflected energy will be removed. This is indeed true for any method of direct wavefield suppression. However, should the direct waveform be substantially different from the reflected arrival, possibly due to the attenuation of higher frequencies along the longer reflected raypaths, it may be prudent to remove the direct wavefield prior to the design of the deconvolution filter, or to ensure that the design of the filter is not influenced by the direct wavefield.

Chapter IV

Imaging Capability of Crosshole Seismic Reflection Surveying

In a crosshole survey between vertical boreholes, we would hope to image at least that part of the section between source and receiver arrays, as well as tapering zones above and below this area. However, we will see that the range of dips sampled over some of the section is inadequate for satisfactory imaging.

4.1 Introduction

4.1.1 Coverage of crosshole surveys

The coverage of crosshole surveys can be explained by first considering the reflection point locus for horizontal layers with one shot and receiver combination. Here we are only considering upgoing reflections. This is analogous to the CMP in traditional surface seismics (Figure 4.1).

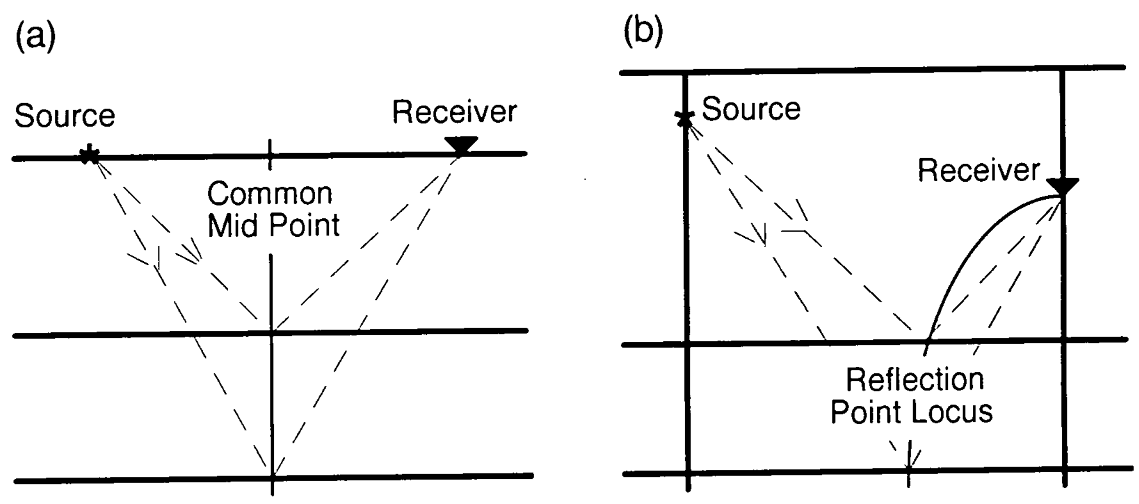

Figure 4.1 The reflection point locus for a single shot and receiver in an isotropic medium. (a) Surface Seismic i.e. CMP and (b) Crosshole.

By considering the reflection point loci for all source and receiver positions (Figure 4.2), and by considering loci for the downgoing wave as well, which would be Figure 4.2 inverted, the expected crosshole coverage is as shown in Figure 4.3. Also shown is the coverage for hole-to-surface surveys which extends out laterally from the borehole to half the distance to the furthest offset geophone. It is possible therefore to build up continuous coverage of the subsurface along lines of boreholes with either crosshole or hole-to-surface surveying.

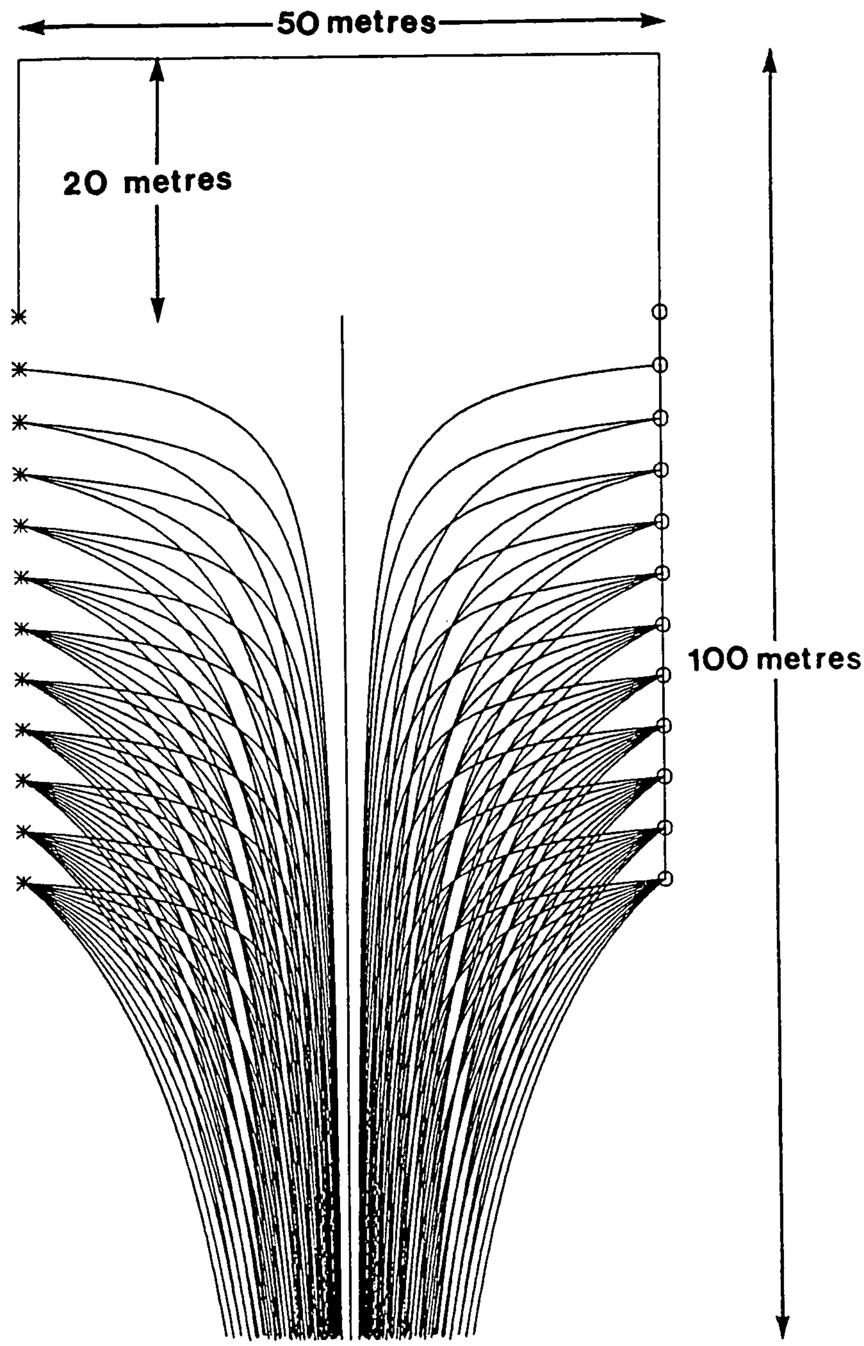

Figure 4.2 Upgoing reflection point loci for a crosshole survey in an isotropic medium (from Findlay et al. 1991).

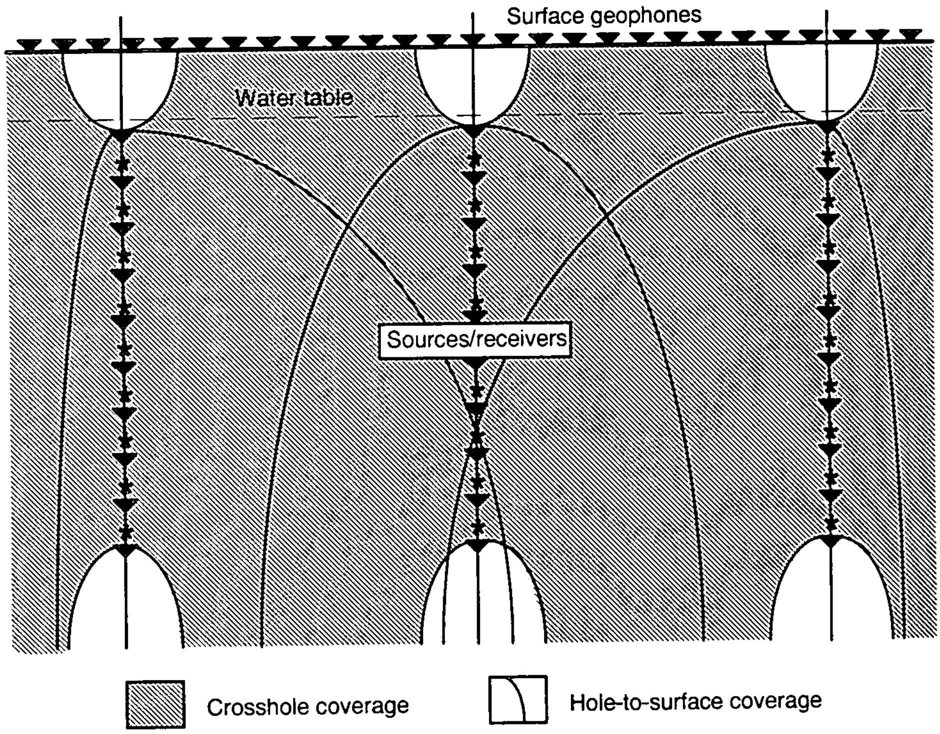

Figure 4.3 Zones of coverage of crosshole and hole-to-surface surveys.

4.1.2 Image quality and dips sampled

When migrating seismic reflection data, image quality depends on the distribution of dips sampled at each image point. To obtain an optimally focused migrated image, our experience is that the migration aperture should include dips of ±15° relative to the local structural dip. If the local structural dip does not lie within the sampled range, then the image is smeared into the familiar, characteristic migration 'smiles' (Carrion et al. 1991). For crosshole seismic reflection surveys, the distribution of dips sampled at each image point is controlled principally by the survey geometry, including source and receiver array lengths and their element spacings. Here it is shown how the survey geometry can limit imaging capability close to the boreholes and even in the middle of the section between the boreholes. It is important to be aware of these limitations in planning surveys and in interpreting crosshole seismic reflection sections. An example of how the image quality of a crosshole reflection survey from coal exploration boreholes relates to the survey geometry is included in §8.1.

We consider here the geometric factors which govern image quality, and hence the spatial extent of useful coverage in crosshole seismic reflection surveys.

We shall principally discuss the range of dips sampled at image points and the effect of the discrete element spacing in source and receiver arrays on the distribution of dips within the sampled range. Related issues such as the elimination of direct waves (§3.3) and the choice of migration scheme (§3.5, §5.1) are discussed elsewhere. These factors are all critical for imaging near to the boreholes. If the lengths of the source and receiver arrays are comparable with (or less than) the borehole separation, then the range of dips sampled will also be too narrow for satisfactory imaging in the middle of the section between the boreholes.

4.2 Dip sampling at image points

For a single source and receiver in a constant velocity medium, an isolated primary arrival might have been scattered from any point on an ellipsoid with source and receiver positions at the foci. The cross-section in the vertical plane through source and receiver is an ellipse (Figure 4.4), which may be called an isochron because the sum of the traveltimes along raypaths from s to **x** and from **x** to **r** is a constant for any position **x** on the ellipse. The tangent to the ellipse at **x** defines the dip angle of a planar reflector through **x** which would give rise to a specular reflection.

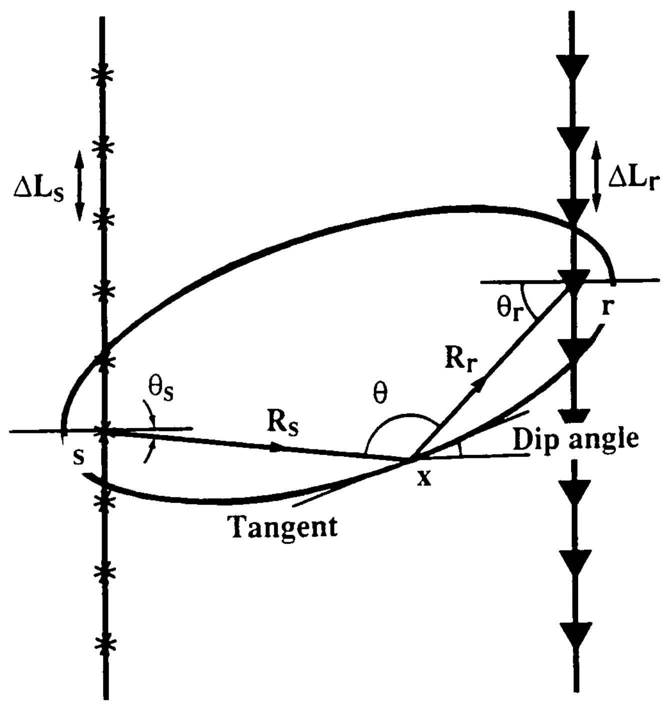

Figure 4.4 Elliptical isochron for the traveltime from a source at **s** to a scattering point **x** and on to a receiver at **r**. (This Fig. is identical to Fig. 3.7)

To perform Kirchhoff migration, rays are traced through a layered velocity model from each image point to each source and receiver location to calculate the traveltimes (see §3.5.4). From the angles with the vertical made by the raypaths at each image point **x**, the dip angle sampled by that source-receiver combination can readily be found.

Figure 4.5 shows the distribution of dips sampled by the upgoing wavefield for part of the section in the example survey from the Lowther South site (boreholes 3438-3437; see §8.1).

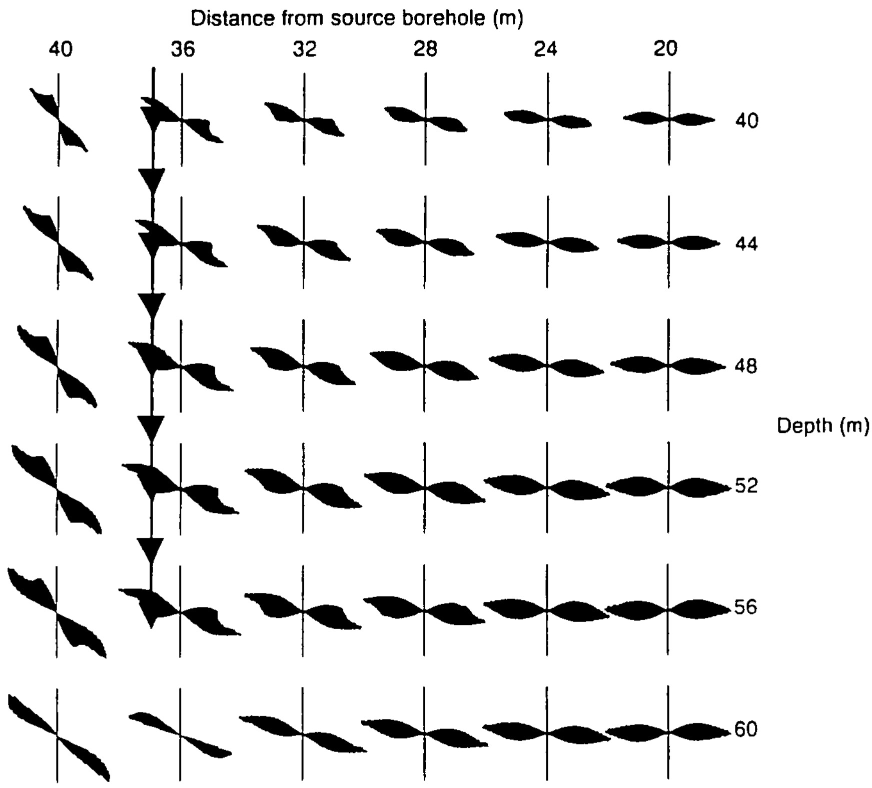

Figure 4.5 Rose diagram showing the distribution of dips sampled at particular image points for the upgoing reflected wavefield. The diagram is plotted for half of the interborehole space over the depth range 40-60 m. Shot depths 10-52m, receiver depths 12-56m, shot and receiver spacing 2m.

For this figure, both source and receiver arrays are assumed to be continuous; the effect of discrete element spacing in each array is considered in the next section. The GB migration scheme (§5.1) of equation (5.3) is equivalent to a stack of migrated CSGs and CRGs, as are the GRT and GK migration schemes, and in every migrated gather the same uniform weighting applies across the range of dips sampled. In forming the rose diagram, therefore, the lengths of the diametrical lines through the centre of each rose are directly proportional to the number of gathers which sample the corresponding dip angle. It can be seen that the weighting tapers off naturally towards each end of the total range of sampled dips.

Another consideration which affects the range of dips sampled at image points is whether any restriction should be placed on the maximum allowed value of the angle θ between the incident and scattered raypaths (see Figure 4.4). As θ increases, the location of the isochron surface on the depth section becomes extremely sensitive to errors in the velocity field. Given that sedimentary rocks are anisotropic, it is virtually impossible to find the velocity field exactly. To prevent the quality of the image being degraded by velocity errors, one could specify a maximum allowed value of θ; we have found it convenient instead to specify a maximum angle which the raypaths can make with the vertical at the image point. This limits the amount of raytracing which has to be done and is consistent with the use of *f-k* filters for direct wave removal. A value of 60° has been found to be suitable to avoid degradation of the image.

The resulting reduction in the ranges of dips sampled by the upgoing wavefield is shown in Figure 4.6. The dip sampling, and therefore the imaging capability, is optimum in the middle of the section around the level of the deepest source and receiver elements. This bodes well for imaging a hydrocarbon reservoir by means of a crosshole reflection survey between two wells which terminate at the base of the reservoir. In such circumstances, traveltime tomography would not be able to image the reservoir because the angular coverage of the raypaths would be much too limited.

At the fault location (see §8.1), near the bottom of the receiver array, the range of dips sampled is from +28° to -7°. Generally, for image points close to the receiver array, the range is limited by the spatial extent of the source array, and vice versa. Towards the centroid of source and receiver arrays (top right-hand corner of Figure 4.6), the range of dips sampled becomes extremely narrow, and the image will become similar to that which would be obtained by reflection point mapping. This is due to the limited vertical extent of source and receiver arrays, which are only slightly longer than the borehole separation. We can thus see it is desirable that the ratio of source/receiver array length to borehole separation should be 2:1 or greater for adequate imaging in this part of the section.

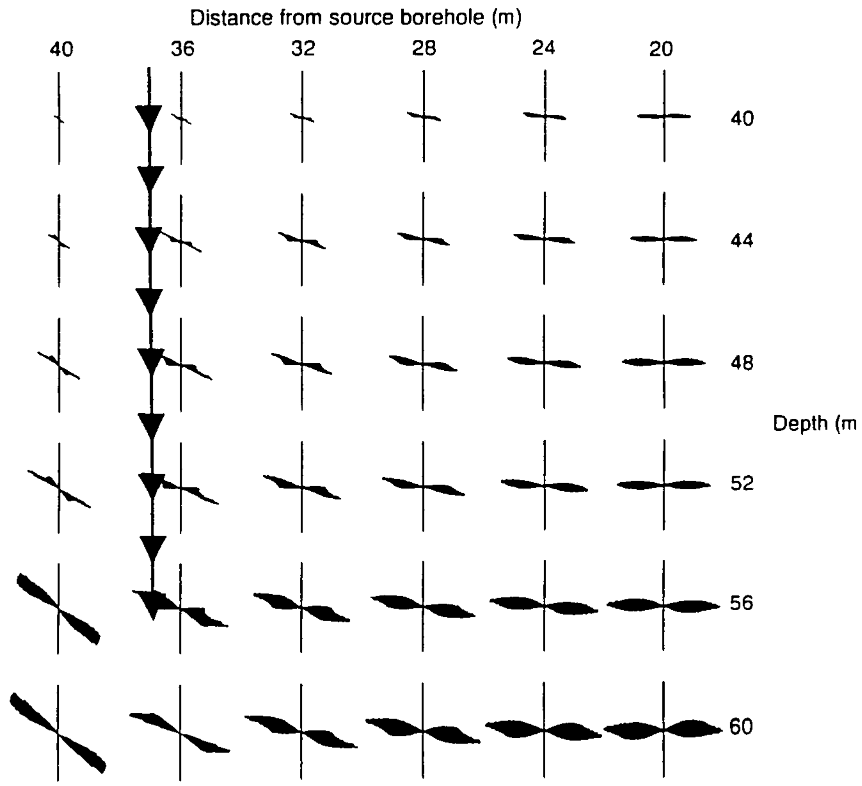

Figure 4.6 Rose diagram showing the distribution of dips sampled at the same image points as Figure 4.5 with the additional restriction that raypaths must be within 60° of the vertical at each image point.

For image points beyond the boreholes, no specular reflections can be received from horizontal interfaces. Even so, there is some imaging capability there if the local structural dip lies within the sampled range, although that would not commonly be the case.

In Figures 4.5 and 4.6 we have only considered dips sampled at image points by the upgoing reflected wavefield. However, it is obvious that similar rose diagrams may be produced for that part of the section which is sampled by the downgoing reflected wavefield. Because of the limited vertical extent of source and receiver arrays, there is little overlap between the areas which can be imaged by the upgoing and downgoing reflected wavefields in this example survey.

4.3 Effect of source and receiver spacing

In the preceding section, we examined the distribution of dips at each image point assuming continuous source and receiver arrays, and we ignored the effect of the spatial sampling interval. In our example survey, the source and receiver positions were spaced at intervals of 2m within their respective arrays. The rose diagram of Figure 4.6 is modified to take account of this in Figures 4.7 and 4.8. As before, the maximum allowed angle between the raypaths and the vertical at each image point is 60°.

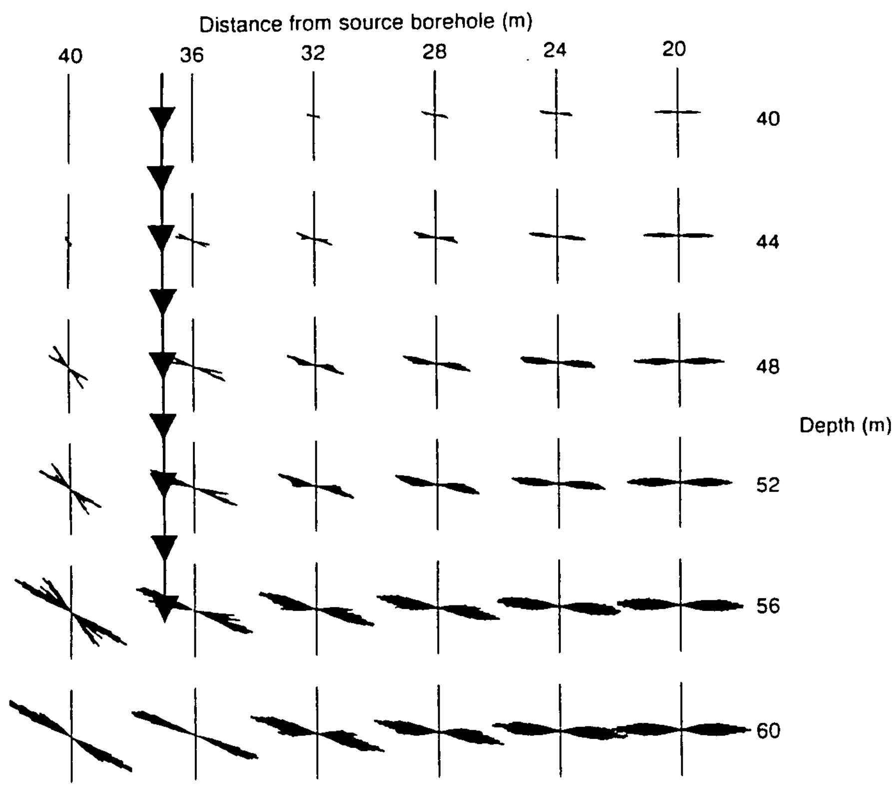

Figure 4.7 Rose diagram showing the effect of discrete spatial sampling on the distribution of dips sampled at the same image points as in Figure 4.6 for the upgoing reflected wavefield.

Towards the middle of the section there are no gaps in the distribution of dips. However, close to the boreholes there are gaps within the range of dips sampled. This is shown in Figure 4.8, plotted on a larger scale, around the suspected fault location adjacent to the receiver borehole. Roses are spaced 0.2 m apart vertically at distances of 1, 2 and 3m from the receiver borehole. The gaps in the range of dips sampled at an image point are dependent on its position relative to the nearest receivers above it. They could be infilled by reducing the receiver spacing in the nearby borehole. Note that reducing the source spacing would only increase the sampling density where dips are already adequately sampled; the gaps in the dip distribution would remain. However, a smaller source spacing would be required in order to improve the image close to the source borehole.

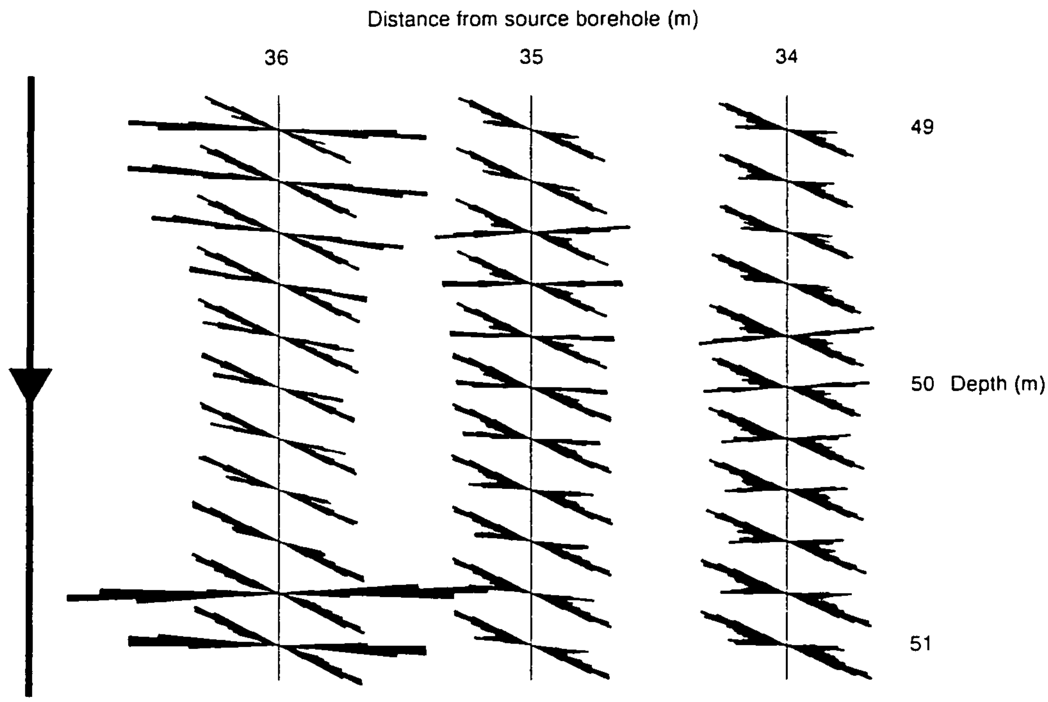

Figure 4.8 Similar rose diagram for image points spaced 0.2 m apart vertically over a 2 m interval at distances of 1-3 m from the receiver borehole.

For image points close to either the source or receiver array, it is the combination of spatial sampling in the near borehole and the extent of the array in the far borehole which govern the ability to form an accurate image. Rose diagrams such as those in Figures 4.7 and 4.8 clearly show where spatial sampling and array lengths are adequate over the section. Where they are inadequate, the image will inevitably be smeared, whatever is done in processing.

Chapter V

Advanced Processing

Two processing techniques are described here which have been developed to enhance the image quality of migrated sections. Generalised Berryhill migration has been developed as a full generalised Kirchhoff migration to include the near-field term, with the aim of improving image accuracy close to the source and receiver arrays. 3-D *f-k-k* filtering is an improved method of wavefield separation for crosshole seismic data.

5.1 Generalised Berryhill Migration

Both GK and GRT migration (equations 3.9 and 3.10) methods are far-field approximations, resulting from dropping the near-field term in the Kirchhoff integral, which might be important for imaging close to the boreholes. Accordingly, Rowbotham and Goulty (1993a) have adapted Berryhill's (1979) method for wave-equation datuming to the crosshole survey geometry to yield a generalised Berryhill (GB) migration scheme, which takes into account both the near-field and far-field terms. This should improve the ability to image close to source and receiver arrays, provided that the element spacing in the nearby array is small enough.

5.1.1 A generalised Berryhill migration scheme

Starting from Kirchhoff's integral theorem, Berryhill (1979) derived the following expression for forward extrapolation of a wavefield in two dimensions from one datum to another through a medium of constant velocity:

$$U_{out}(t) = \frac{1}{\pi} \sum_i \Delta L_i \cos \theta_i \, \frac{t_i}{R_i} \, Q(t - t_i) \tag{5.1}$$

where the summation is over traces on the input datum,

ΔL_i is the trace spacing on the input datum,

θ_i is the angle between the normal to the input datum and the raypath from the input datum to the output datum,

t_i and R_i are the traveltime and distance, respectively, along the raypath, and $Q(t-t_i)$ is obtained by convolving the trace on the input datum with the function $\frac{d^2}{dt^2}\left[\frac{\left(t^2-t_i^2\right)^{1/2}}{t_i}\right]$ for $t > t_i$.

The expression (5.1) retains the contributions from both the near-field term and the far-field term in the Kirchhoff integral. For migrating crosshole data in a CSG, we extrapolate backwards in time and use the imaging principle that the forward extrapolated wavefield from the source must be time-coincident with the backward extrapolated wavefield from the receivers at each image point **x**. To convert from 2-D to 2.5-D (§3.5.2), the summation is multiplied by the amplitude correction factor ($\sqrt{R_r + R_s}$). A further correction factor of ($\sqrt{R_s}$) is needed to allow for the fact that illumination of image points depends on their distance from the source (§3.5.2). Thus we rewrite (5.1) for a CSG in the real survey to give the following expression for the reflectivity estimate:

$$\hat{C}_s(\mathbf{x}) = U_{out}(t_s) = \frac{1}{\pi}\sum_r \Delta L_r \cos\theta_r \frac{t_r}{R_r}\sqrt{R_s(R_s+R_r)}\, Q(\mathbf{s},\mathbf{r},t_s+t_r) \qquad (5.2)$$

where the subscript r refers to the receiver array, t_s is the traveltime from the source at **s** to image point **x**, and t_r is the traveltime from **x** to a receiver at **r**. $Q(\mathbf{s},\mathbf{r},t_s+t_r)$ is obtained by convolving the trace recorded at receiver **r** from source **s** with the function $\frac{d^2}{dt^2}\left[\frac{\left(t^2-t_r^2\right)^{1/2}}{t_r}\right]$ for $t > t_r$, and taking the value of the output trace at $t = t_s + t_r$.

As discussed by Dillon (1990), reciprocity demands that both source and receiver arrays are taken into account symmetrically in forming the reflectivity estimate (§3.5.3). This is achieved by summing the reflectivity estimates for all CSGs and CRGs together. Ignoring the scalar multiplier, this gives for generalised Berryhill migration (where the summation is over all traces):

$$\hat{C}(\mathbf{x}) = \sum \Bigg(\Delta L_r \cos\theta_r \frac{t_r}{R_r} \sqrt{R_s(R_s + R_r)}\, Q(\mathbf{s},\mathbf{r},t_s + t_r)$$

$$+ \Delta L_s \cos\theta_s \frac{t_s}{R_s} \sqrt{R_r(R_r + R_s)}\, Q(\mathbf{r},\mathbf{s},t_r + t_s) \Bigg) \qquad (5.3)$$

Note that for each input trace there are two different functions Q which have to be evaluated independently. In the case of $Q(\mathbf{r},\mathbf{s},t_r + t_s)$, the recorded trace has to be convolved with the function $\frac{d^2}{dt^2}\left[\frac{(t^2 - t_s^2)^{1/2}}{t_s}\right]$ for $t > t_s$. The calculation of these functions represents the additional computational load over GK migration, but this is not heavy because the operator to be convolved with the input trace in each case may be truncated after relatively few terms without introducing significant error (Berryhill 1979).

As for GK migration, GB migration is derived for constant velocity, with reflectivity due to density contrasts in an acoustic medium. The reflection coefficients in these circumstances are independent of incident angle. If desired, a factor $\cos^2(\theta/2)$ could be introduced into the summation as for GRT migration, where reflectivity is due to velocity contrasts in an acoustic medium of constant density. Dropping the terms $\frac{t_r}{R_r}$ and $\frac{t_s}{R_s}$, corresponding to the reciprocal of velocity along the raypath, introduces neglible error as these only fractionally change the weighting across the range of dips sampled. As for GRT and GK migration, GB migration is symmetrical with respect to source and receiver arrays, and actually amounts to a combined stack of individual migrated CSGs plus individual migrated CRGs.

Comparisons between the results obtained using GK and GB migration are made in §7.5.6 (tank dataset - Figure 7.33) and §8.1.3 (coal exploration dataset - Figure 8.3). The migration program **berrymig**, which evolved from Findlay's **kirchmig** program (§3.5.4), is included as Appendix E.

5.2 3-D *f-k-k* wavefield separation

In processing crosshole seismic reflection data, it is necessary to separate the upgoing and downgoing primary reflected wavefields from each other and from the direct wavefield. The problems of achieving this satisfactorily are addressed earlier (§3.4.1). Here the use of 3-D *f-k-k* filters is proposed for separating the wavefields (Rowbotham and Goulty 1993b). The complete dataset is treated as a data volume, with each sample defined by the three coordinates of source depth, receiver depth and time.

5.2.1 Why is it necessary?

I address the following two problems. The first is that direct arrival energy remains in the wavefield separated gather and is imaged as coherent noise in the migrated section (§3.4.1). One remedy for this would be to mute the direct arrival (§3.3.1), either before or after wavefield separation; however, this step will also remove some of the reflected energy required for imaging. Pratt and Goulty (1991), Stewart and Marchisio (1991) and Rector et al. (1992b) have used what they described as "common ray angle" (§3.3.4), "common interval" and "common offset" gathers, respectively (all equating to a constant difference between source and receiver depths), to accentuate the differential time moveout between the direct and reflected waves. The direct arrival is then removed by some type of multichannel filter designed to attenuate arrivals with zero moveout. However, these methods are not ideal for the smaller datasets used in this study (§3.2.4).

The second problem is that, although wavefield separated CSGs appear to contain coherent reflected events, the coherency is much lower if the same traces are viewed in CRGs. Similarly, if wavefield separation is carried out in CRGs, then the CSGs are much less coherent. This may be due to the presence of multiples, direct shear arrivals and mode-converted events. Rector et al. (1992a, 1992b) overcame this second problem by filtering in multiple domains. Following the removal of the direct wavefield in common offset space, two copies of the data were made; one was retained in CSGs and the other was sorted into CRGs. The two sets of gathers were processed to attenuate reflections from horizontal locations near the common variable well, prior to 2-D *f-k* filtering for wavefield separation. The final stacked section (Lazaratos et al. 1992) was then a combination of the two datasets, with CSGs imaging the half of the interborehole space near the receiver borehole and CRGs imaging the half near the source borehole.

5.2.2 The method

I take the next logical step by applying a one-pass *f-k-k* wavefield separation filter to the whole dataset from a crosshole seismic survey. In effect, the method is almost equivalent to two-pass 2-D *f-k* filtering, in both CSGs and CRGs. However, visualization of the dataset as a 3-D volume provides a valuable insight and, as a practical matter, transformation into *f-k-k* space allows the filter tapers to be selected optimally.

Previously, *f-k-k* methods have only been used for conventional geometries, i.e. two orthogonal spatial axes and a time axis (e.g. Peardon and Bacon 1992). However, there is no reason why the *f-k-k* technique cannot be extended to the crosshole geometry with source depth and receiver depth as the independent spatial variables (indeed, stacking diagrams utilize this concept in plotting source and receiver positions along different axes, even though they are collinear). In the crosshole case, where the arrays are parallel, we may envisage the data as a volume with the source depth (s) and receiver depth (r) axes as being orthogonal to each other and to the time (t) axis. Upon 3-D Fourier transformation, the data are transformed from s-r-t space to f-k_s-k_r space, where f is temporal frequency, k_s is wavenumber corresponding to the s-axis, and k_r wavenumber corresponding to the r-axis. We can then select any volume of the data in *f-k-k* space in a one-pass filtering operation.

5.2.3 Viewing the wavefields in 3-D

Let us first consider the direct wavefield and the upgoing reflected wavefield from a single horizontal interface between two vertical boreholes, within a medium of constant velocity V (Figure 5.1). In s-r-t space (Figure 5.2), a slice at constant source depth will give a CSG, and one at constant receiver depth will give a CRG. The direct and reflected waves can be graphically represented as the surfaces in Figure 5.2. The surfaces meet where the source or receiver is positioned at the reflector, i.e., where source or receiver depth is 30m, since direct and reflected traveltime will be equal.

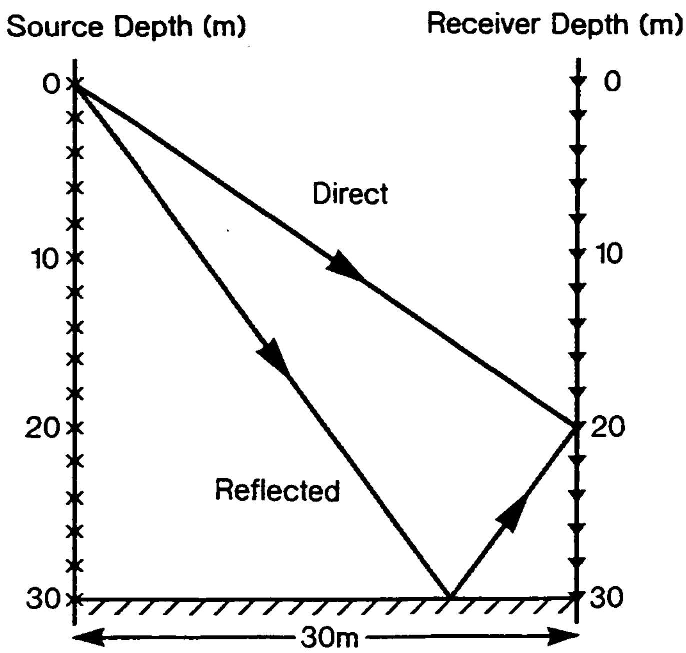

Figure 5.1 Direct and upgoing reflected raypaths from a horizontal interface between two vertical boreholes, within a medium of constant velocity V.

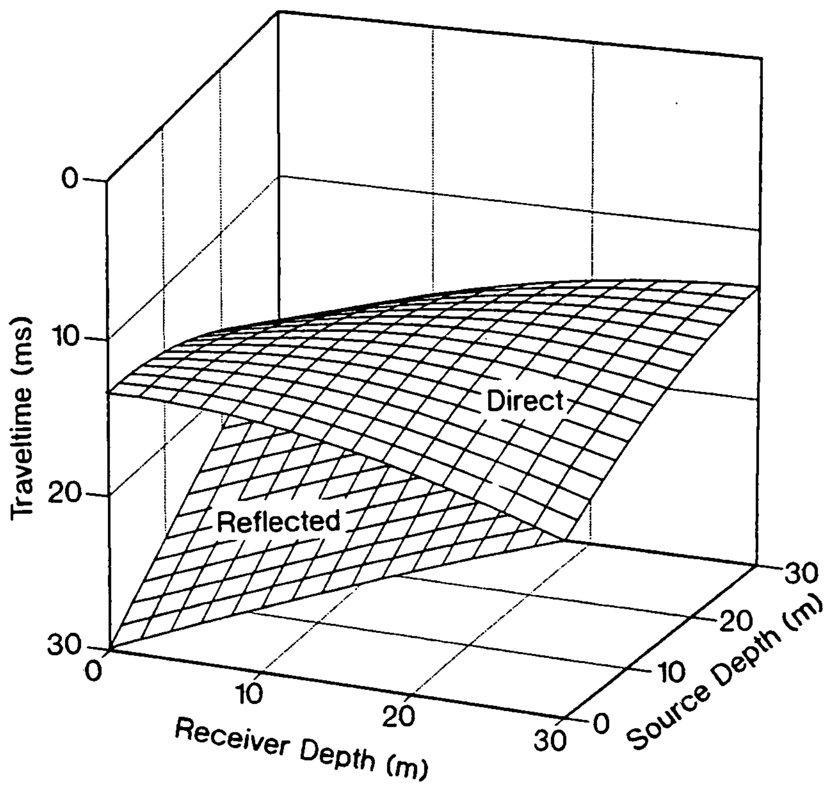

Figure 5.2 The direct and upgoing reflected wavefields from the simple case of Figure 5.1 can be graphically represented as surfaces in s-r-t space.

A plane wave in *s-r-t* space will be represented in *f-k-k* space by a vector through the origin with slope equal to the apparent velocity of the arrival. For the simple circumstances under consideration, the direct wavefield transforms into the plane $k_s = -k_r$ in *f-k-k* space (Figure 5.3). The upgoing reflected wavefield lies in the plane $k_s = k_r$, in one quadrant of the *f-k-k* domain. Likewise a downgoing reflection will lie in the same plane but in the opposite quadrant. For a dipping reflector, the reflected wavefield would lie to one side of the $k_s = k_r$ plane, but still in the same quadrant. Head waves will also lie in these quadrants, having similar moveout characteristics to reflected arrivals.

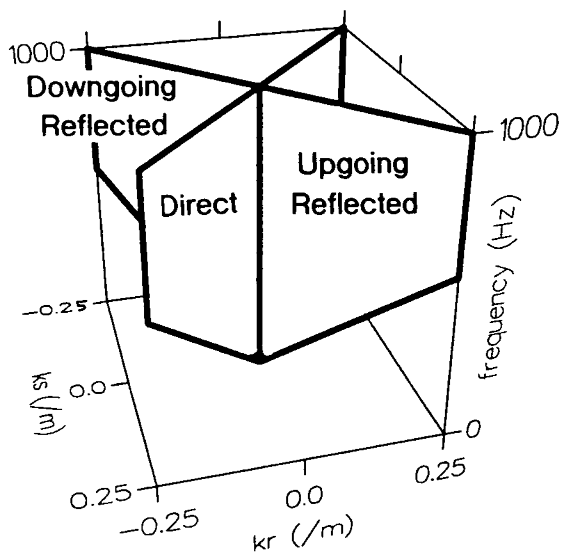

Figure 5.3 The direct wavefield transforms into the plane $k_s = -k_r$ in *f-k-k* space. The upgoing reflected wavefield lies in the plane $k_s = k_r$, in one quadrant of the *f-k-k* domain. Likewise a downgoing reflection will lie in the same plane but in the opposite quadrant.

Raypaths for the strongest multiples in crosshole data undergo two reflections. They will be either up or downgoing at both source and receiver, and so will lie in the same quadrants as the direct waves, around the $k_s = -k_r$ plane. Clearly, direct shear waves will also lie in the same two quadrants.

5.2.4 Filter design

The traveltime minimum of the reflected wavefield surface occurs when both the source and receiver are at the interface depth. Here both the direct and reflected raypaths are horizontal, corresponding to $k_s = k_r = 0$ (infinite apparent velocity). However, for depth migration we have found it convenient to specify a maximum angle which raypaths can make to the vertical at the image point. This is because the calculated location of the reflecting point is extremely sensitive to velocity errors for large values of incident angle; hence, if such raypaths are included, the quality of the migrated image is liable to be degraded (Rowbotham and Goulty 1993a, Qin and Schuster 1993). A maximum angle to the vertical of 60 degrees has been found to be suitable for this purpose. The minimum possible value of k_s and k_r for any frequency f is then $f/2V$, with the maximum value being f/V, corresponding to vertical raypaths across the source and receiver arrays, respectively.

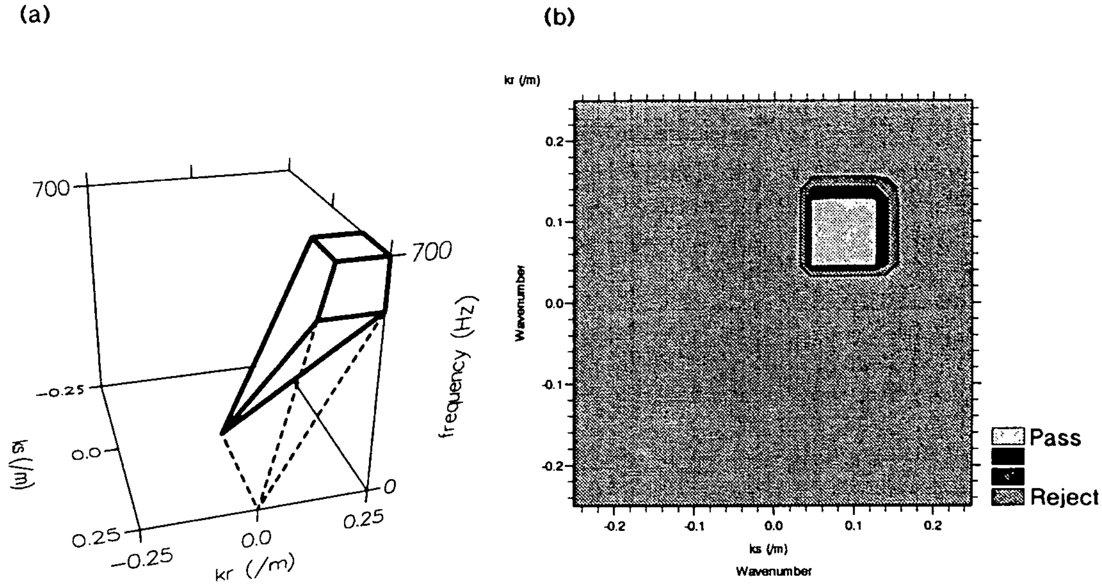

Figure 5.4 (a) Full-pass region of the filter for the upgoing primary reflected events (tapers not shown for clarity). (b) Frequency slice of the filter pass region at 250Hz plotted with a linear amplitude scale (i.e. constant contour interval).

Thus in 3-D f-k-k space, the part of the upgoing wavefield which we wish to use for imaging is defined by $k_s = k_r$ values lying between f/V and $f/2V$. Following the advice of Stewart (1989) and Peardon and Bacon (1992), one-pass filtering in f-k-k space is preferred over two-pass filtering with successive fan-shaped filters in f-k_s and f-k_r space. Although in practical terms the difference is minimal, a smoother pass volume is defined, especially in the tapered zones at the corner of the volume. As in 2-D filtering, tapers are necessary to prevent ringing upon transformation back into s-r-t space. The full-pass region of the filter for the upgoing primary of the reflected wavefield is shown in Figure 5.4a (tapers not shown for clarity) and a frequency slice at 250Hz is shown in Figure 5.4b (tapers shown). The reject region contains the primary downgoing wavefield, both compressional and shear direct waves, and the strongest multiples.

The program **xhr3** (Appendix D) was written to perform *f-k-k* filtering. Figure 5.5 shows an example data volume pre- and post-filtering. Notice that it is dominated by the direct energy along the $k_s = -k_r$ axis. This has been muted in the filtering. The *f-k-k* technique has been used for filtering the physical model and one of the Coal Measures datasets used in this study, and the results are presented in §7.4.3 and §8.2. They show that the use of 3D *f-k-k* filtering has made the muting of direct arrivals in the time domain superfluous, since the separation of direct and reflected wavefields and of up and downgoing reflections is achieved in one operation.

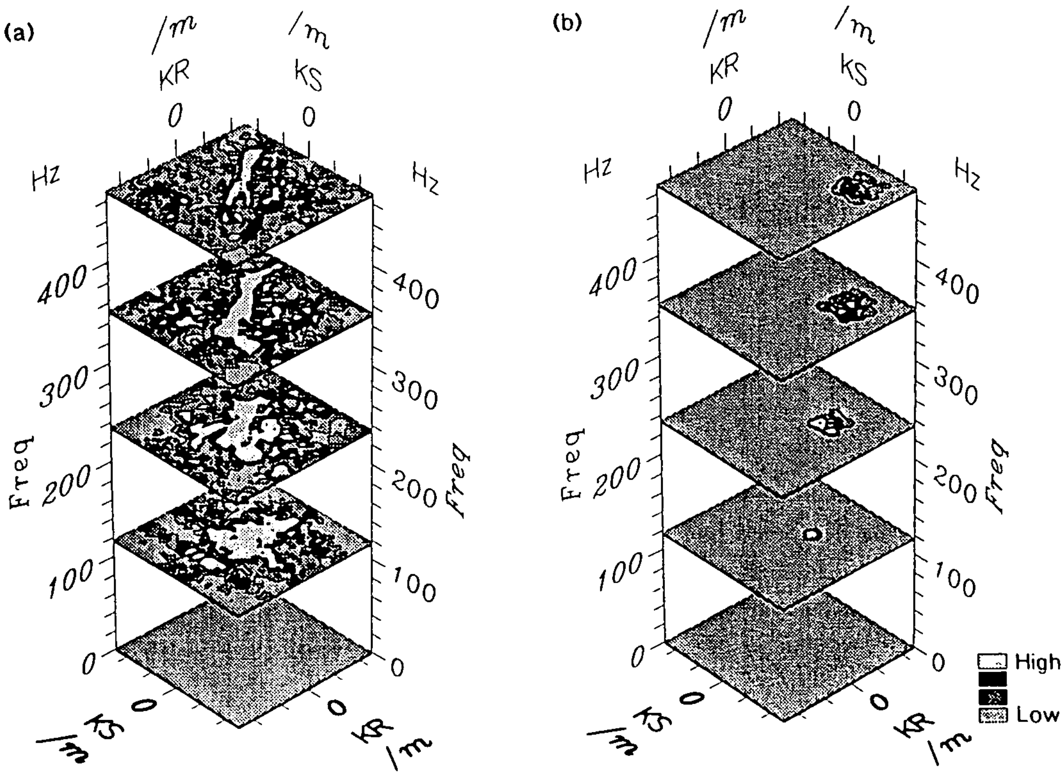

Figure 5.5 The data volume transformed into the *f-k-k* domain, and depicted as *f*-slice plots with a linear amplitude scale: (a) before filtering, and (b) after 3-D *f-k-k* filtering.

Chapter VI

Groningen

This chapter covers the imaging of a common source gather from a crosswell survey in the Groningen gas field.

6.1 Introduction

A crosswell survey in the Groningen gas field was acquired by Seres a/s of Trondheim (Vaage and Ziolkowski 1992). A schematic cross section between the deviated wells used in the experiment is included as Figure 6.1.

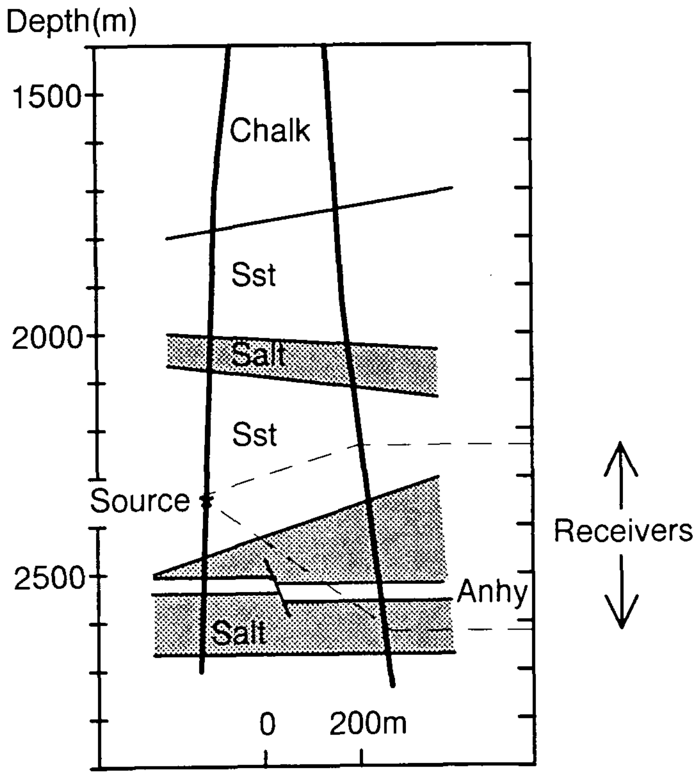

Figure 6.1 Source and receiver positions with the interpretation of the inter-well geology based on Schlumberger logs and knowledge of the general geological setting of the area.

The source was at 2350m depth with receivers in Triassic and Zechstein strata from 2233-2606m depth. The wells deviated away from each other over this gather from approximately 300m well separation at the top receiver, to 330m at the bottom receiver position.

6.2 Data examples

As explained in §2.1.2, six common source and receiver gathers were acquired during this experiment, of which only one common source gather of 125 records provided decent coverage. For completeness, the four CRGs are included as Figures 6.2(a)-(d).

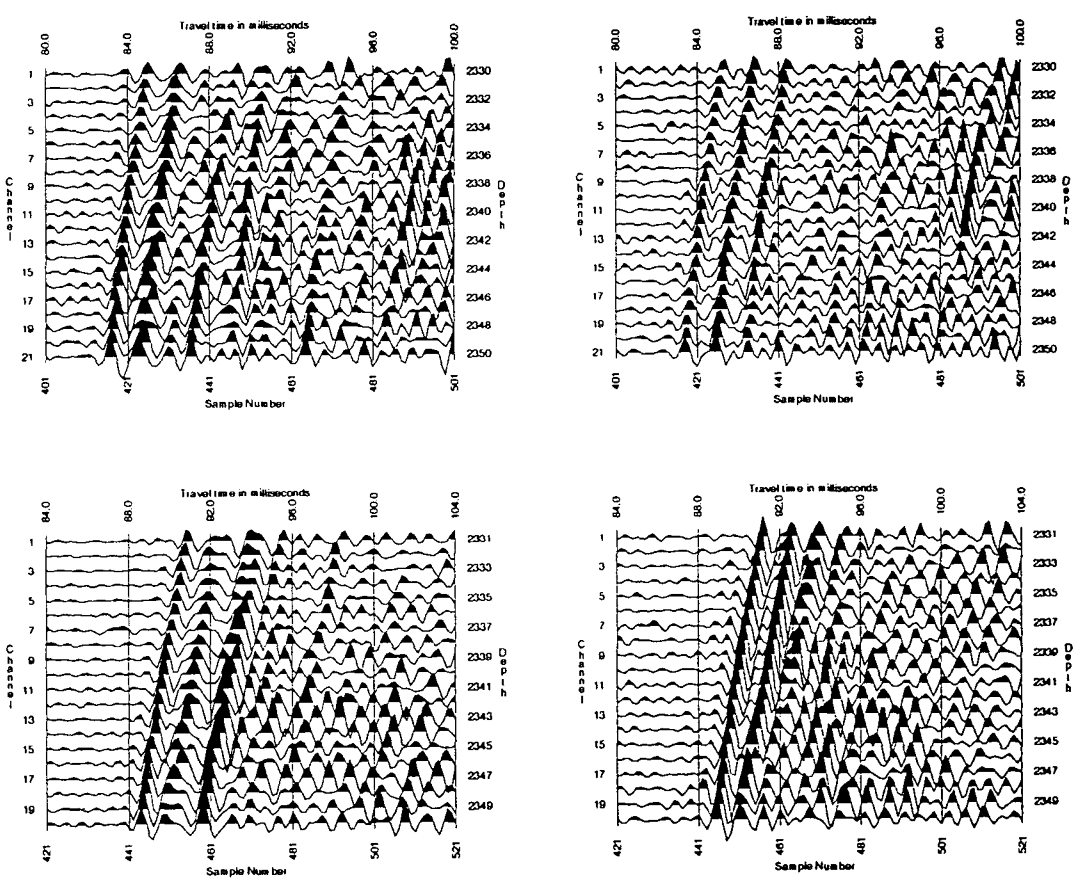

Figure 6.2 CRGs at depths (a) 2453m. (b) 2456m. (c) 2503m. (d) 2506m.

6.2.1 Arrivals in the smaller gathers

The common source gather from 2350m depth with 1m receiver spacing (Figure 6.3) provides detail of the wavefield as the receiver passes through the anhydrite at 2519m depth (Figure 6.1), although this is of limited use for imaging. However, it does make apparent the receiver cable stretch of 6m (§2.1.2), such that the interface appears instead at 2513m receiver depth. Head waves can be seen as the first arrivals above this top anhydrite interface, and there is some indication of reflected energy from the interface.

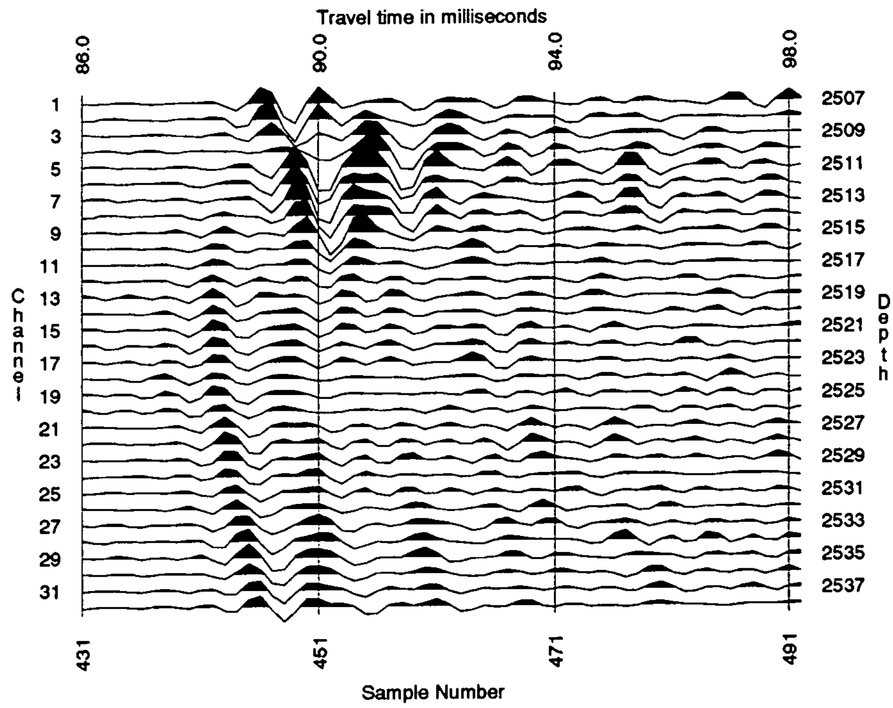

Figure 6.3 CSG from 2350m depth.

Tube wave energy is of high amplitude in these gathers, although it arrives at a later time to that displayed in Figures 6.2 and 6.3, suggesting that a temporal mute can be applied without risk of attenuating the earlier reflected energy. The tube wave can be seen by plotting the traces from the CRG at 2453m depth (Figure 6.2a) on a longer time-scale (Figure 6.4a). On the *f*-*k* spectrum of the gather (Figure 6.4b), the tube wave energy is seen to be aliased with an apparent velocity of about 1600m/s.

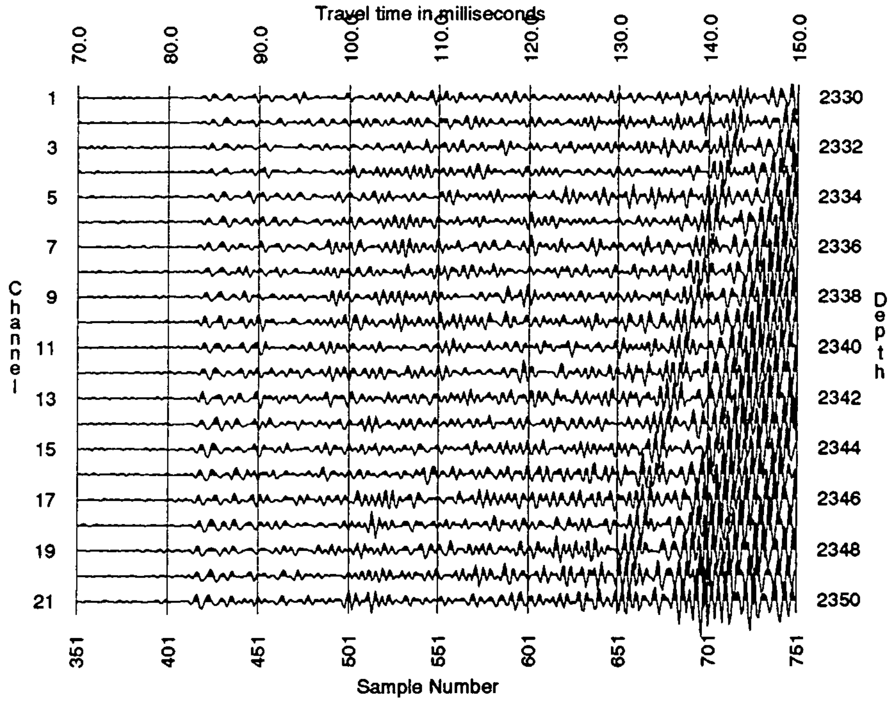

Figure 6.4 (a) CRG from 2453m depth.

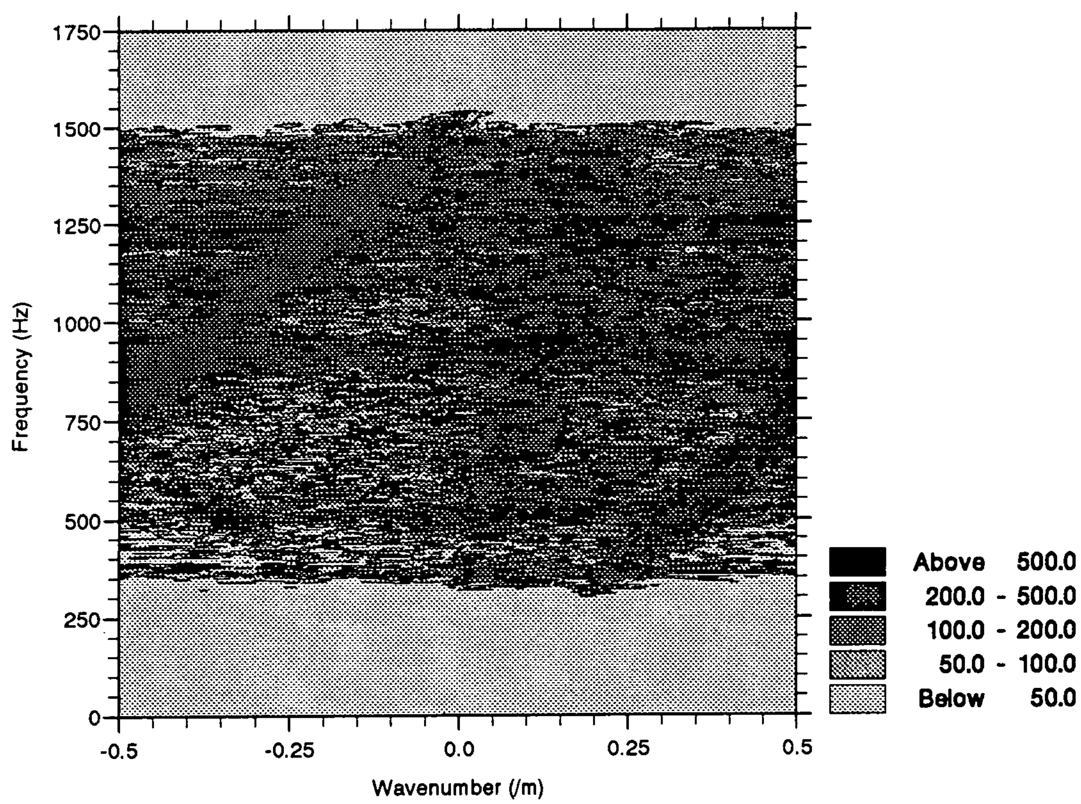

Figure 6.4 (b) *F-k* spectrum of this CRG, showing tube wave energy, with an apparent velocity of 1600m/s, aliased above 800Hz.

6.2.2 Arrivals in the gather used for imaging

The 125-trace CSG is displayed in Figure 6.5. The received signal had a broad bandwidth of 300-1500Hz, with a pronounced peak around 960Hz (Figure 6.6).

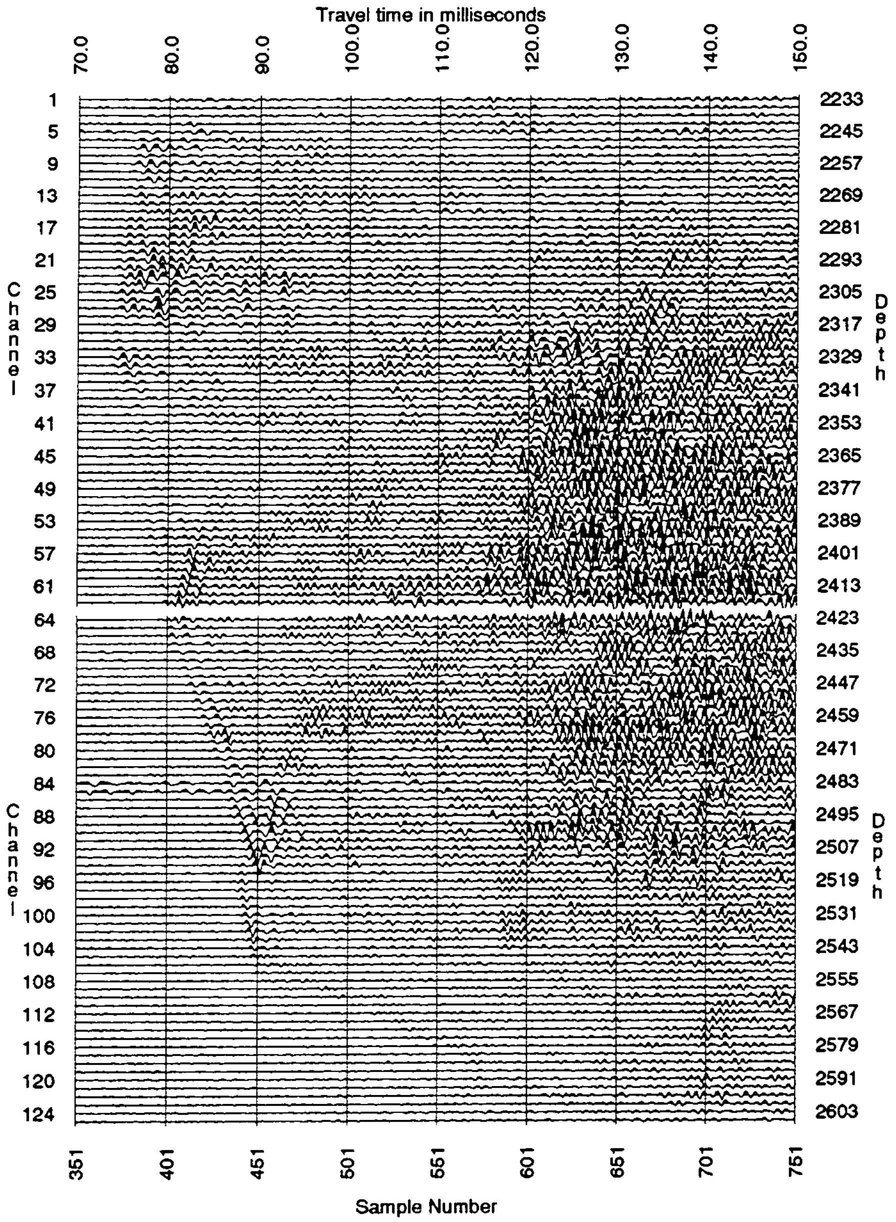

Figure 6.5 CSG from 2350m depth.

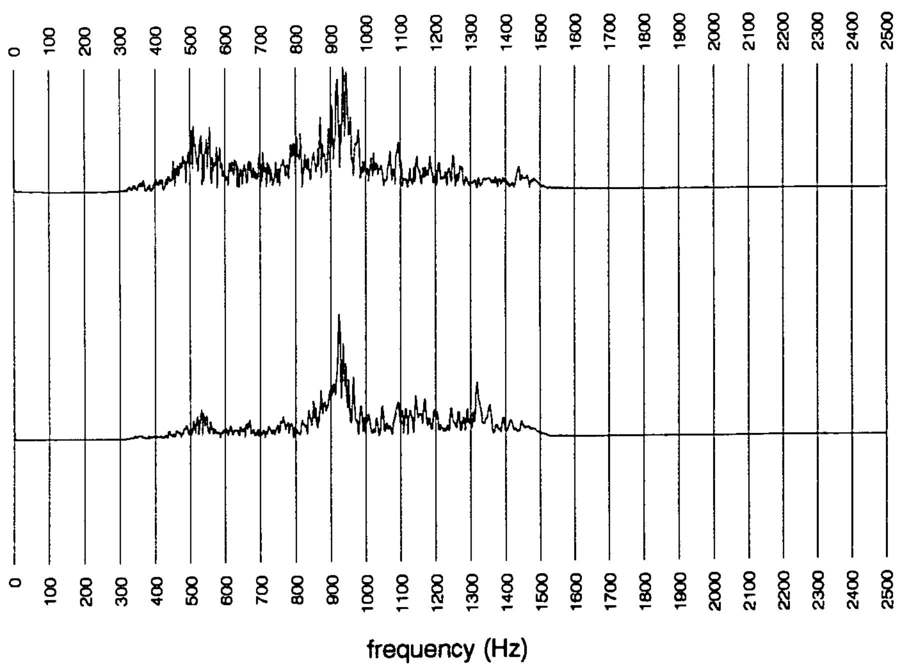

Figure 6.6 Amplitude spectrum of traces 1 (upper trace) and 63 (lower trace) from the CSG at 2350m depth.

To show more detail of the earlier arrivals, this has also been plotted on shorter time-scales as two separate CSGs of 63 and 62 records (2233-2419m, 2423-2606m depth) (Figures 6.7 and 6.8), the separation being where irregular spacing occurred within the gather.

Several arrivals are observed in these data, leading to some interesting observations about the physical properties of the media being travelled through. Firstly, two reflections are prominent with a frequency content around 1kHz, coming from the anhydrite layer at 2519m depth and from a thin layer near the Top Zechstein (2429m), which corresponds to an increase on the density log. There are also weaker reflections from another interface within the Zechstein at around 2471m depth, and possibly from Top Zechstein at 2390m depth. The reason why a prominent reflection is not expected from the Top Zechstein interface (Triassic Bunter Sandstone passing to Permian Halite) is that the density of the halite is less than that of the sandstone even though the velocity may be higher. The anhydrite layers within the Zechstein have a strong impedance contrast with the surrounding halite as both their density and acoustic velocity are greater.

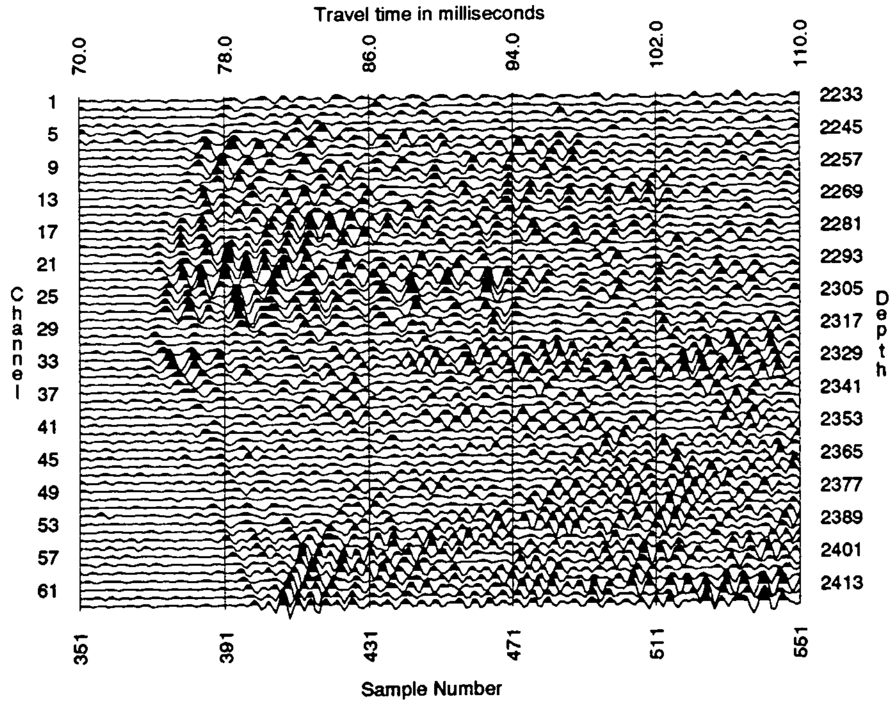

Figure 6.7 CSG from 2350m depth with receivers at depths 2233-2419m.

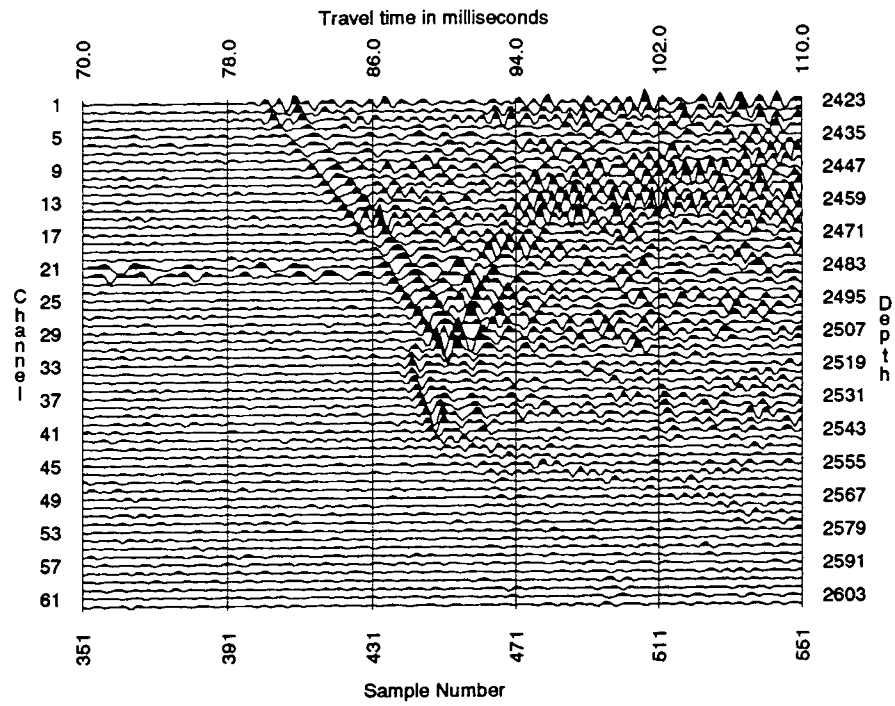

Figure 6.8 CSG from 2350m depth with receivers at depths 2423-2606m.

No strong reflected trough is seen from the base of the larger anhydrite layer to match the large amplitude peak from the top, even though the impedance contrasts should be similar but of opposite sign. However, this is consistent with the large refractive effects expected at the top of a high velocity layer and with the post-criticality at the upper surface of the anhydrite (Figure 6.13).

Raytracing modelling of this CSG has been performed by Vaage (personal communication) and from these a reflection from the top of the Rotliegend reservoir (depth ~2700m) is interpreted as arriving on traces from depths 2554-2606m between sample 700 on the lowest trace to sample 740 at the base of the anhydrite (Figure 6.5). Processing to migrate this energy has been attempted, though this proved unsuccessful because of the lack of constraints available, such as knowledge of the velocities lower down in the section and lack of effective coverage provided by just one CSG.

The direct and reflected arrivals are low amplitude in a zone in the depth range 2341-2395m (Figures 6.5 and 6.7), and normalisation of the traces does not amplify them because of the later tube waves. For processing this data it is a simple matter to mute the tube waves before reapplying normalisation, as the tube waves do not interfere with the reflected arrivals in z-t space. As the pre-first-break noise levels of these traces are comparable to others in the gather, it is not thought that the amplitudes have been altered significantly prior to the data arriving in Durham.

It is suggested that the shadow zone occurs at depths within the Bunter Sandstone, whereas the amplitudes recover lower down once the receivers pass into the salt. This agrees with the interpreted reflection from 2390m at the Top Zechstein boundary. It is not known what causes this shadow zone, though it could be due to poor coupling between the borehole casing and the formation. However, the cement bond log on the Schlumberger Array Sonic tool showed the receiver well to be well cemented, and it is interesting to note that the tube wave amplitudes have not been affected at this depth.

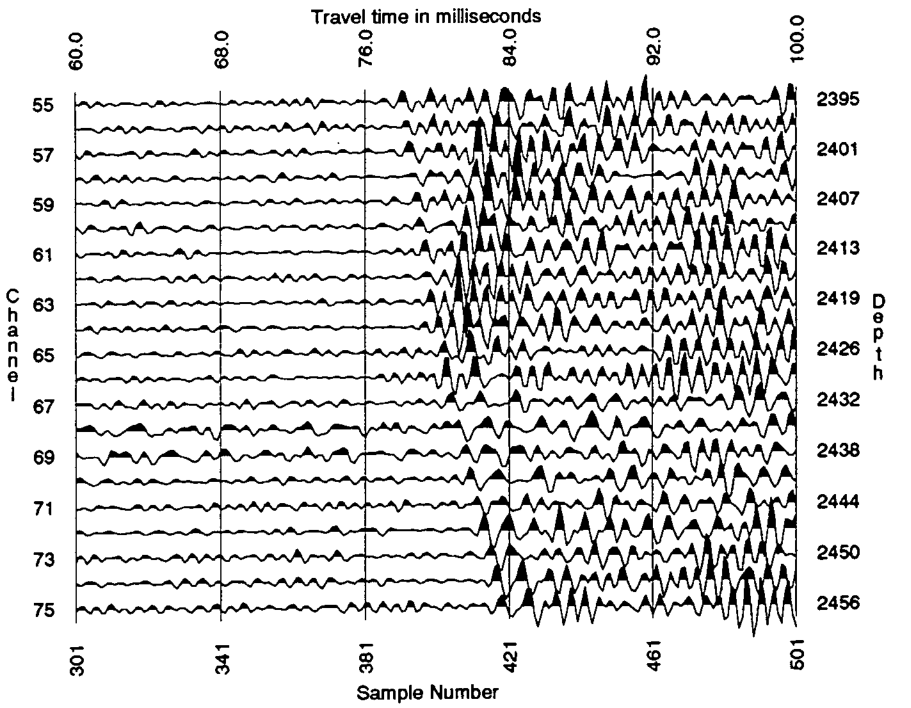

Figure 6.9 CSG from 2350m depth with receivers at depths 2395-2456m. Anhydrite layer interpreted to be at 2429m depth.

The anhydrite layer at 2429m depth has been interpreted as being 5m thick from gamma ray and density logs run in the receiver borehole. Since refraction in the lower anhydrite layer generates head waves (§6.2.1; Figures 6.3 and 6.8), the possibility of channelling by this shallower thin layer exists. Traces around the layer have been plotted in Figure 6.9. Although not at first apparent, the high-frequency low-amplitude arrival ahead of the direct arrival at traveltime 78ms (391 samples) on traces at depths 2426m and 2429m is thought to be channelled energy. This energy is not spiked successfully by the waveshaping method used in §6.3. However, thin layers will preferentially channel energy of higher frequency than the direct arrival, since lower frequencies will 'see' the thin layer as thinner than the high frequencies. The layer will 'leak' head waves, hence the low amplitude.

It is interesting to note that the moveout of the direct arrivals is slightly less steep below this thin anhydrite layer than above it (Figure 6.5), implying a velocity inversion, with the halite above the layer having a higher velocity than that below. This behaviour implies that the anhydrite layer is continuous and locally extensive enough to form a barrier to communication between the two halite layers.

6.3 Processing and Results

A standard crosshole reflection processing scheme was applied to these data - waveshaping deconvolution, wavefield separation and migration (see Chapter 2). A single direct arrival was extracted from those traces where the direct arrival was discernible by aligning the direct arrival energy and stacking the traces. Using this wavelet as input (§3.2.2), an optimum Wiener shaping filter was designed, with a zero-phase Butterworth wavelet as the desired output, with a signal bandwidth of 300-1500 Hz, and applied to the data.

As transformation into the *f-k* domain requires regular trace spacing, the data were transformed in the two separate gathers before being migrated together. The later arriving tube waves were removed by tapering the traces. To prevent ringing in the *f-k* spectra upon transformation, a spatial taper was applied to the deepest and shallowest traces of both halves of the gather to smooth the steps in amplitude at the edge traces. Of primary interest was the upgoing wavefield, since the target interfaces were located below the shot depth. The downgoing migrated image (not shown) displayed little coherent reflectance.

Several processing sequences were performed on the data prior to migration to compare their merits in the final depth migrated sections. It was found that applying an AGC to the data after wavefield separation and before migration improved the migrated section, especially above the Top Zechstein reflector where the first arrivals were weak. The deconvolved upgoing wavefield was then migrated using the GK migration algorithm (§3.5.3) to produce Figure 6.10. It should be noted that GK migration reduces to Kirchhoff wave-equation migration for a single CSG - an experiment that obeys the wave equation. A simple two-layer velocity model was constructed by integrating information provided on the interpreted geological cross section (Figure 6.1), the direct arrival traveltimes and receiver positions, and by knowledge of the expected depth of the reflectors from the CSG.

The migrated depth section (Figures 6.10 and 6.11) shows good imaging of the strong reflections from the sloping thin anhydrite layer and from the anhydrite layer at 2519m depth[1], and of the weaker Top Zechstein (2390m) and sloping reflection (trough) at 2471m. The two anhydrite reflected arrivals in the CSG both had sharp cut-offs, producing good coherent reflectors over about half the theoretical ray-traced coverage.

[1]No correction has been applied for cable stretch - see §2.1.2.

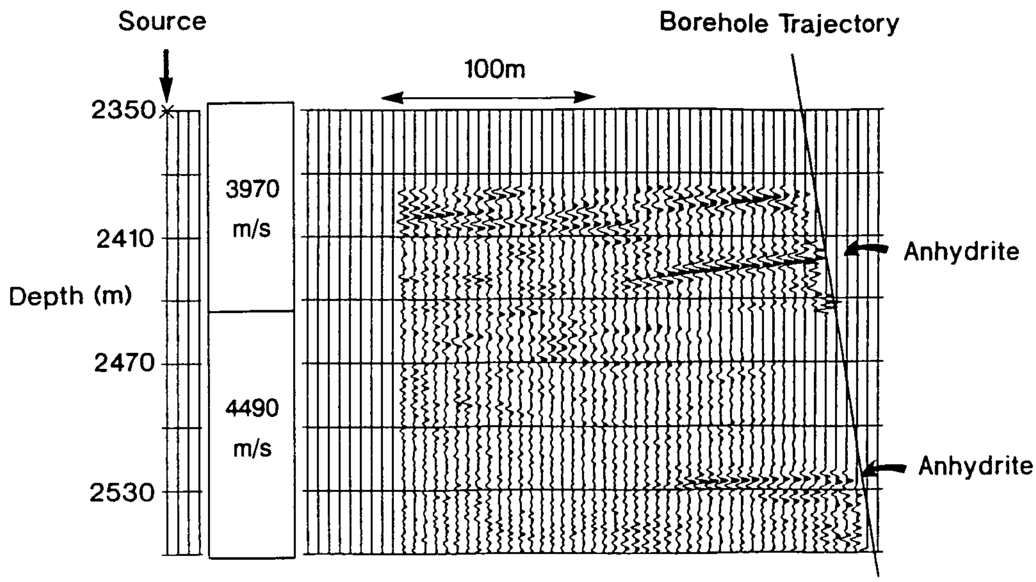

Figure 6.10 Migrated depth section of the CSG from 2350m depth. Migration velocity field also displayed. Trace spacing - 5m. SEG reverse polarity i.e. peak = compression.

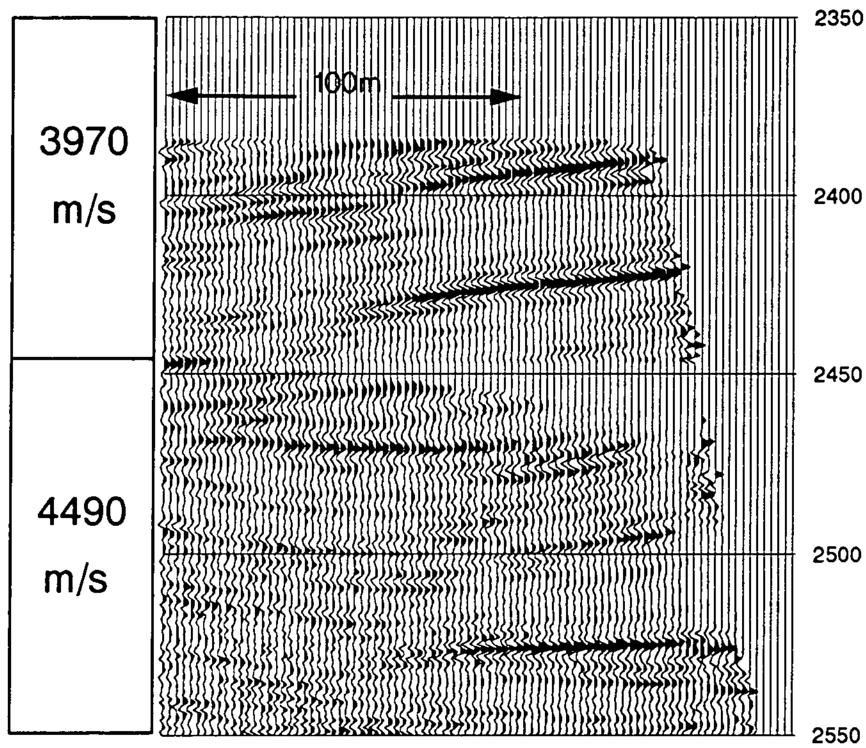

Figure 6.11 Partial migrated depth section of the CSG from 2350m depth. Migration velocity field also displayed. Trace spacing - 2m.

The anhydrite layer is known to have a normal fault with a throw of around 20m located between the two wells. The termination of the top anhydrite reflector at

2519m provides a lower limit for how close the fault could be to the receiver well (Figure 6.12).

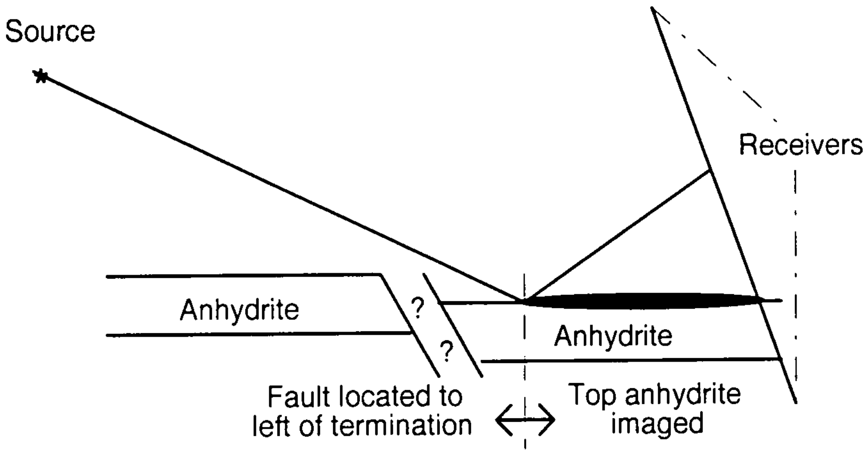

Figure 6.12 A simple ray diagram to show how the termination of the top anhydrite reflector provides a lower limit to how close the fault can be to the receiver borehole.

By raytracing, it has also been found that this termination may mark the transition from high amplitude post-critical reflections to lower amplitude pre-critical reflections (Tooley et al. 1965) (Figure 6.13).

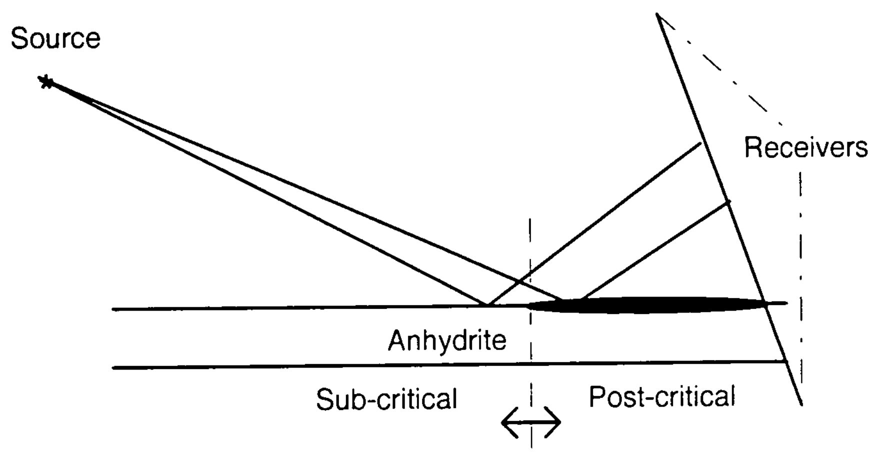

Figure 6.13 A simple ray diagram to show how the termination of the top anhydrite reflector could be due to the transition from post-critical to sub-critical reflections.

To confirm which of these is the cause of the termination, and of the termination of the reflector above, it would be necessary to acquire several common-shot gathers from widely spaced source locations in the left hand well.

Chapter VII
Physical Model Data

A crosshole survey with 51 source and 51 receiver positions was performed through a model using Durham University's physical modelling system. Here the datasets are described, and the successes and failures of various processing schemes discussed.

The acquisition of this dataset is described in §2.2. To recap, the model is made of seven layers of epoxy resin, it is 46.5mm wide, and the pre- and post-flood models were identical except for the inclusion of a low velocity flood zone in the 'reservoir' layer. Source and receiver spacing was 2.5mm, and the sampling interval was 250μs. All dimensions were scaled by a factor of 1000 to represent the case for real data (i.e. mm to m, ms to s).

7.1 Previous Work using these Datasets

These datasets have been subjected to full waveform inversion schemes, such as diffraction stack migration and frequency-domain acoustic and elastic wave equation imaging (Pratt and Goulty 1991, Pratt et al. 1991). The data have also been subjected to GRT imaging (Li and Worthington 1990). Traveltime tomography was used to provide the initial velocity field to be used for both imaging methods. However, to prevent errors in the tomographic stage feeding through to the later stages of processing, the known geometry and velocities were used to raytrace through.

Both traveltime and amplitude tomography have been performed on these data (Leggett 1992, Leggett et al. 1993); in that work, an initial estimate of the pre-flood velocity model was derived as if calibrated sonic logs had been run in both wells, this being a more realistic simulation of a real crosshole experiment. It was found that the velocity tomograms imaged the flood zone quite accurately, whereas amplitude tomography imaged the flood zone less precisely as an area of higher absorption.

7.2 Arrivals in the Wavefield

Figure 7.1 shows CRG 19 (depth 45m) from the post-flood model as a deconvolved gather with an agc of window length 50 samples applied to enhance the later arrivals. Figure 7.2 is an expanded view of the traces around the flood zone. Arrivals are labelled with letters. The characteristics of the arrivals are largely in accordance with those observed in datasets acquired at a West Texas carbonate reservoir (Van Schaack et al. 1992) and at the Devine test site in Texas (Smith et al. 1993).

7.2.1 P-wave Direct (Pd)

This is a high amplitude arrival, and its characteristics can be noted from the source signature approximation derived for deconvolution purposes (e.g. Figure 7.7). As noted by Leggett (1992), multi-pathing of the direct arrival around sharp corners of the flood zone occurs, and is not compensated for by the raytracing program. This effect is not clearly observed in CRG 19.

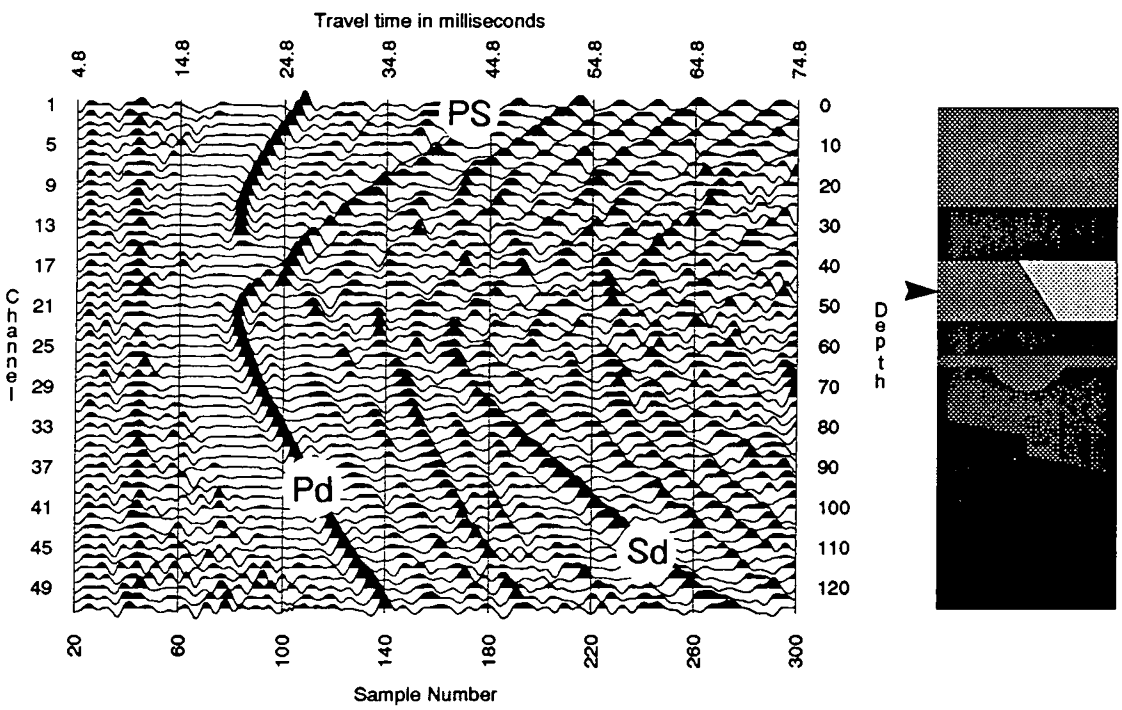

Figure 7.1 Post-flood CRG 19 deconvolved to zero-phase with arrivals marked (agc applied with window length of 50 samples). Position of R19 marked.

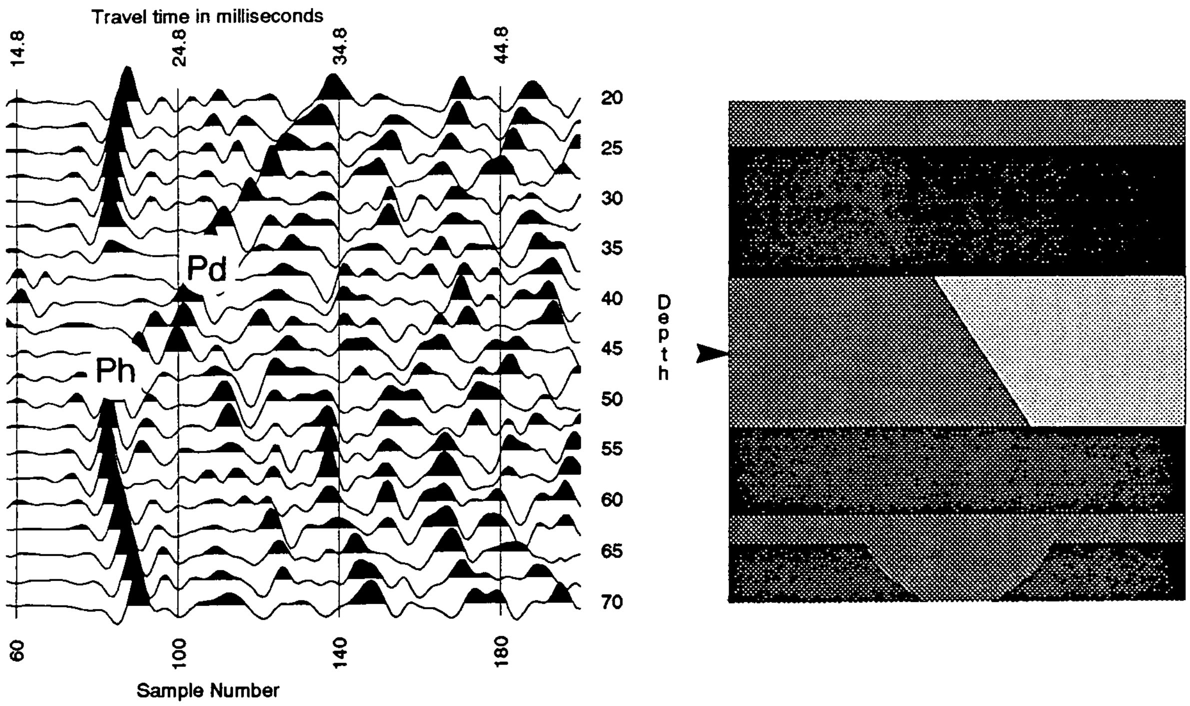

Figure 7.2 Expanded view of Figure 7.1. Position of R19 marked.

7.2.2 S-wave Direct (Sd)

This arrival is visible at low amplitude in the raw gather, but is greatly amplified by the AGC post-deconvolution as it lies outside the 50 sample window around the direct arrival. It interferes with the P-wave reflected arrivals on certain gathers, and will therefore be seen as noise in migrated images if not removed (see §7.4.1). The S-wave reflected arrivals could also be used for reflection imaging, though it was not considered worthwhile for such low amplitude arrivals.

7.2.3 P-wave Reflected

These arrivals are only noticeable in the raw data by paying attention to the variations in waveform of the direct arrival from trace to trace caused by interference with reflected arrivals. In the deconvolved data, the reflected events can be discerned by comparing moveouts with the direct arrival moveouts. They are seen much more clearly after wavefield separation when the direct wave has been removed (§7.4).

7.2.4 P-S converted wave (PS)

An arrival with S-wave velocity originates from the intersection of the P direct arrival with the top of the flood zone at 37.5m depth (Figure 7.2). It is interesting to note that this arrival would be passed in 2-D *f-k* filtering performed in CRGs (§7.4.1) since the arrival would have the moveout of a P-wave, whereas in CSGs it would lie outside the pass fan with the S-wave velocity. In 3-D *f-k-k* filtering (§7.4.3), the arrival would be muted.

7.2.5 P Head wave (Ph)

These can be seen arriving ahead of the high amplitude direct arrival which has travelled through the low-velocity flood zone. A sketch of the interpreted arrivals and raypaths is given in Figure 7.3. This interpretation can be checked by calculating the traveltime differentials of the raypaths. As first arrivals, these will cause problems if picked as direct arrivals for tomography, and in fact it is thought that the images obtained by Leggett et al. (1993) suffered from this. Head waves will tend to produce exaggerated tomographic estimates of velocity and attenuation locally since they have earlier traveltimes and lower amplitudes than the following direct arrivals.

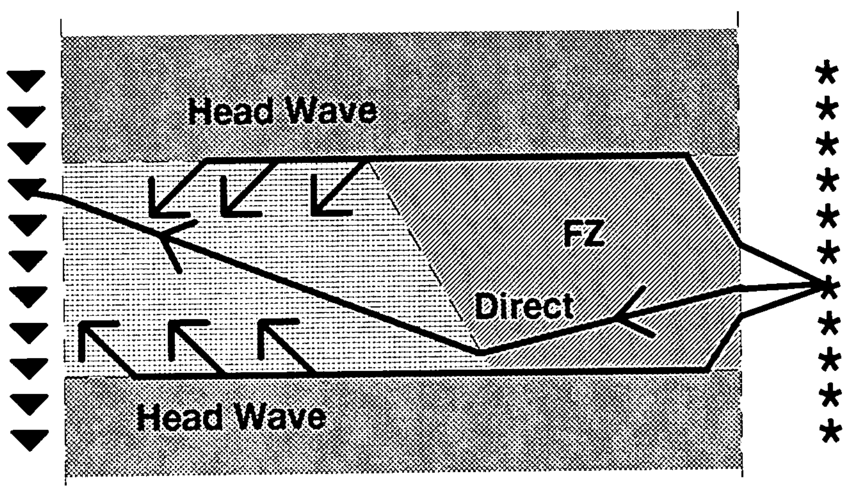

Figure 7.3 Sketch of interpreted direct and head wave raypaths in the region of the flood zone (FZ).

7.3 Processing - Removal of the direct wavefield

In many respects, the tank data proved more difficult to process than the field data, despite the advantages of having a known geometry and no anisotropy. The first and foremost reason for this was the relatively high amplitude and length of the direct arrival, which prevented any reflected arrivals being identified in the raw data. The reason for the predominance of the direct wavefield over the reflected wavefield was the weaker impedance contrasts of the interfaces in the epoxy resin models than in the real geological strata. The first task of processing was then the removal of the direct wavefield, to leave the reflected (scattered) wavefield. Several schemes have been tried.

7.3.1 Shaping and muting

The first method used was the basic one of shaping the source signature to zero phase and then muting it. For shaping, the simplest method involved using an averaged direct arrival through water, i.e. no model present, as input for designing an optimum Wiener shaping filter. The desired output was a zero-phase Butterworth wavelet with a signal bandwidth of 150-600Hz (Figure 7.4).

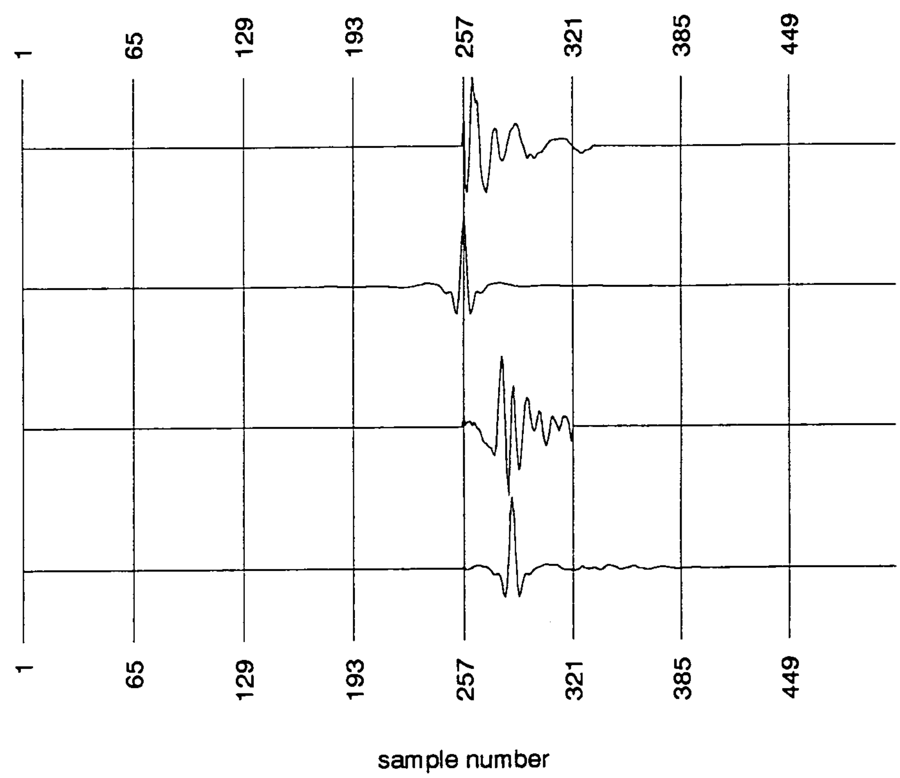

Figure 7.4 From top: source signature obtained by aligning and stacking wave through water, desired Butterworth output, deconvolution filter, actual output.

Choosing the desired output to be a zero-phase wavelet with the same amplitude spectrum as the source signature was also tried (Figure 7.5) to counter the ringing effects of a notch at 250Hz in the amplitude spectrum of the data. This was the method used by Leggett et al. (1993) for shaping the direct arrivals to a peak prior to first break picking for traveltime tomography.

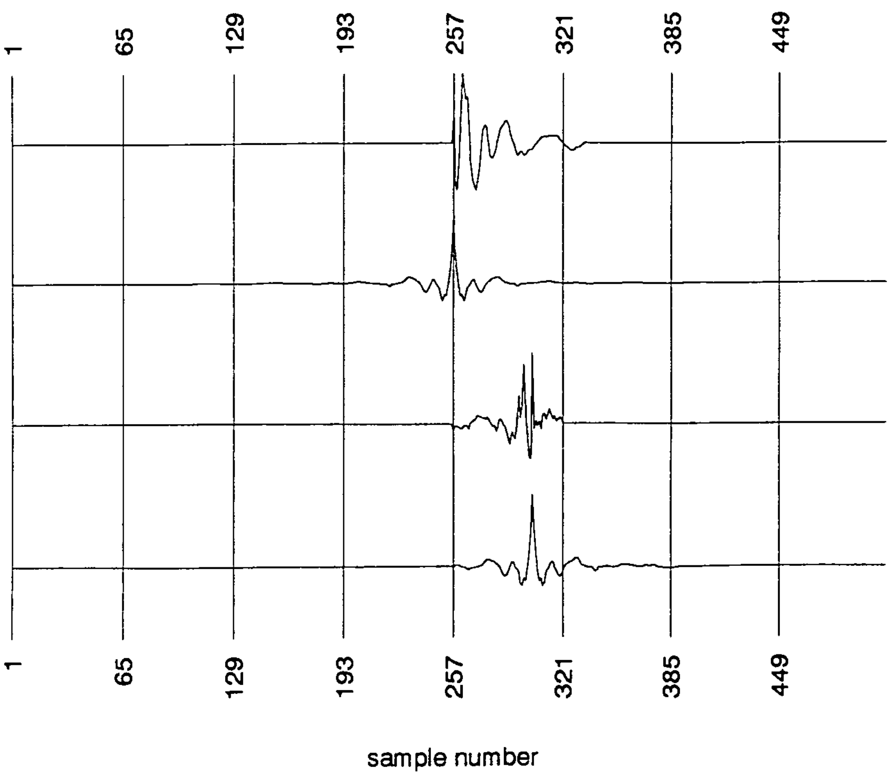

Figure 7.5 From top: source signature obtained by aligning and stacking wave through water, desired output with same amplitude spectrum as input spectrum, deconvolution filter, actual output.

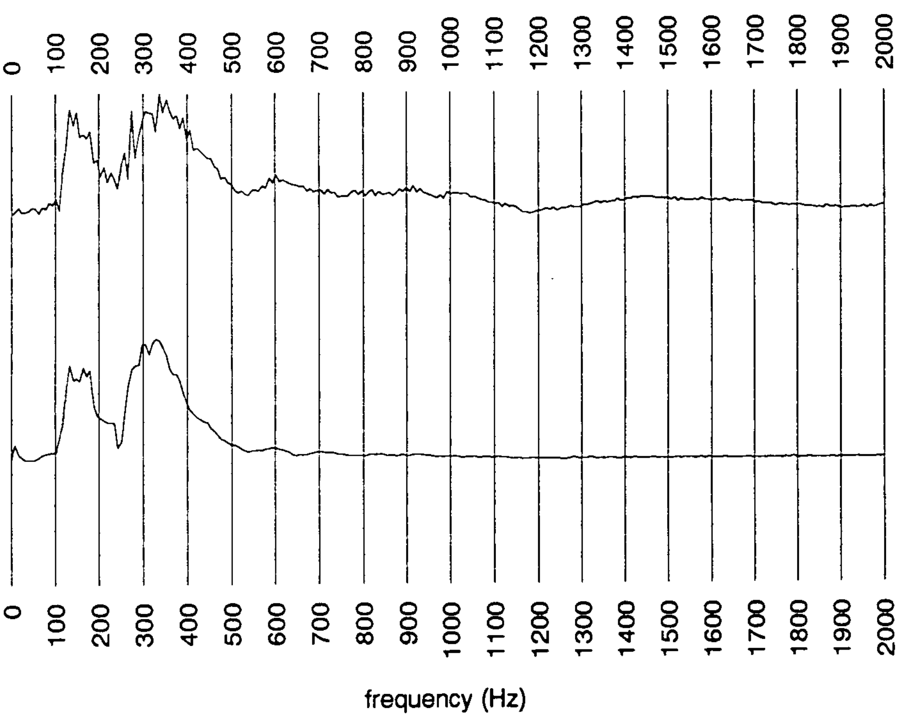

Figure 7.6 Amplitude spectra of the averaged direct arrival through water (top spectrum) and through the model (bottom spectrum).

These methods were found to be unsuitable for deconvolving the data because the arrivals passing through the model were of different shape to those passing through water alone, due to preferential attenuation of higher frequencies during passage through the model (Figure 7.6). By comparing the averaged direct arrival through water with the averaged direct arrival through the model (Figure 7.7), it is apparent that the relative amplitudes of the distinctive triplet of peaks has been changed by passage through the model. As a test, the filter designed with the water wavelet as input (Figure 7.4) has been applied to the wavelet through the model to produce the bottom trace in Figure 7.7. The filter has failed to bring all the energy of trailing peaks into the initial peak, and the peak is delayed behind the first break (sample 60). This is as expected since attenuation of the higher frequencies would cause a pulse to broaden and attenuation is causal. Applying the filter to the raw data can be seen to produce the same effects (Figure 7.8), with the initial peak behind the first break (marked with a dot), and later energy mimicking the moveout of the first arrival.

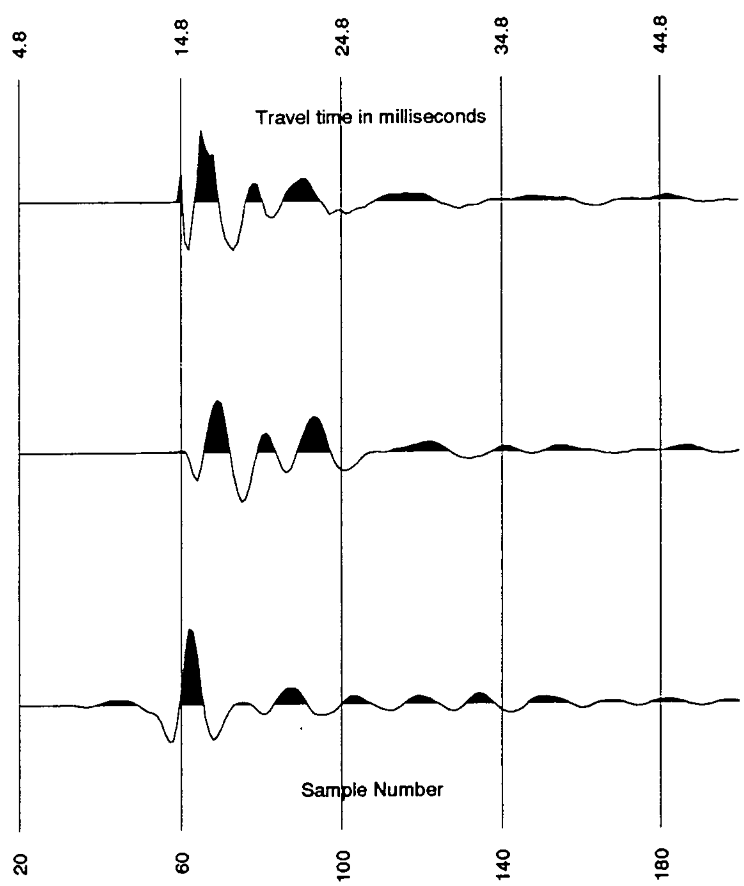

Figure 7.7 Input wavelet from aligning and stacking direct arrival through water (top), through model (middle), result of applying filter (bottom), designed with water wavelet (Figure 7.4), to model wavelet.

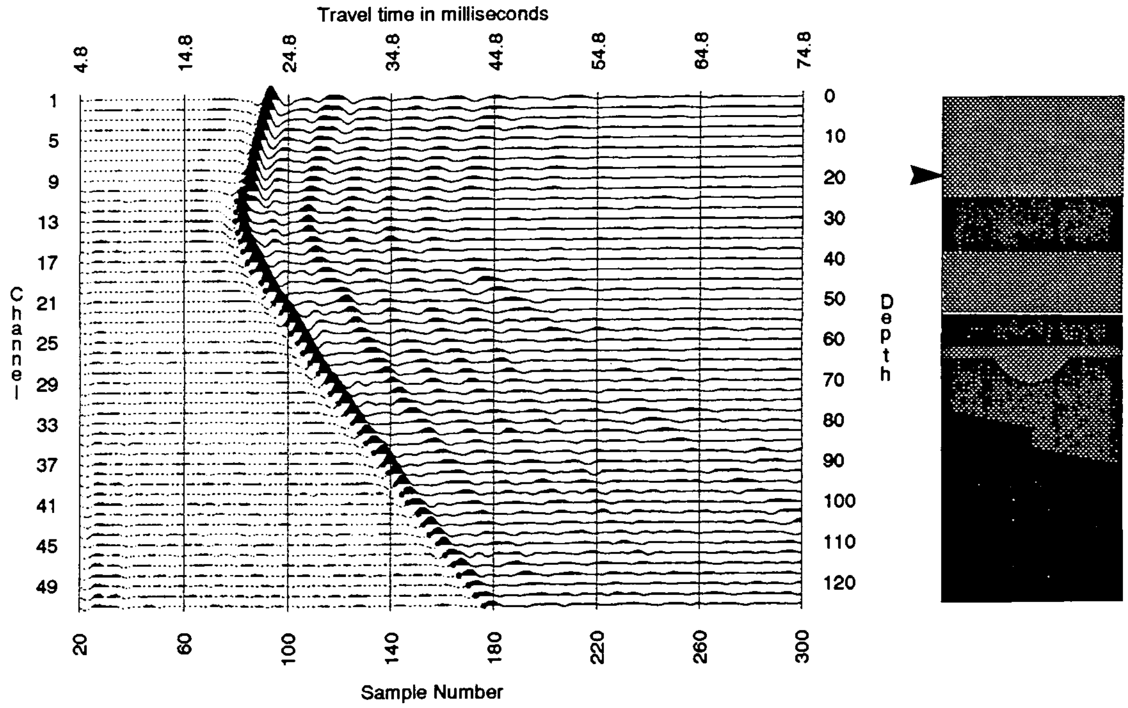

Figure 7.8 CRG 9 deconvolved using the filter in Figure 7.4.

Instead, an input wavelet was obtained by aligning and stacking the direct arrivals over a range of take-off angles through the model. Since stacking tends to attenuate high frequencies, the number of traces stacked was kept to the minimum necessary to give a good representative wavelet. This wavelet had a larger notch at 250Hz than the water wavelet and so both methods of defining an output wavelet used above (Butterworth wavelet and input amplitude spectrum = output amplitude spectrum) were tried (Figures 7.9 and 7.10).

It is evident that the 250Hz ringing due to the amplitude spectrum notch is more of a problem for this second input wavelet (compare Figures 7.4 and 7.9). The designed filter has had to inject more response at this frequency to achieve the desired output, resulting in ringing in the output wavelet at this frequency and hence difficulty in applying a mute to the direct arrival. The remedy suggested by Hatton et al. (1988) of adding in more white noise (the previous spectra having 2% white noise added) did not remove the ringing completely.

After each of the above shaping schemes, the direct arrival energy was muted out by applying a tapered mute after the picked first breaks. However, this muting scheme was limited in its usefulness because of the zero-phase wavelets:-

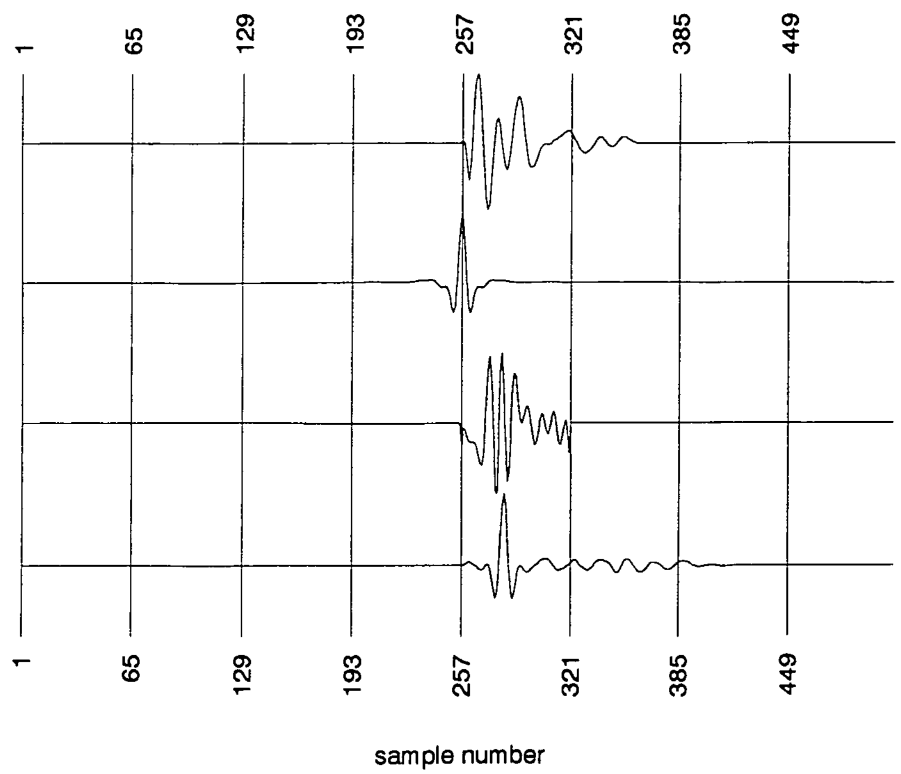

Figure 7.9 From top: input wavelet from aligning and stacking direct arrival through model, desired Butterworth output, deconvolution filter, actual output.

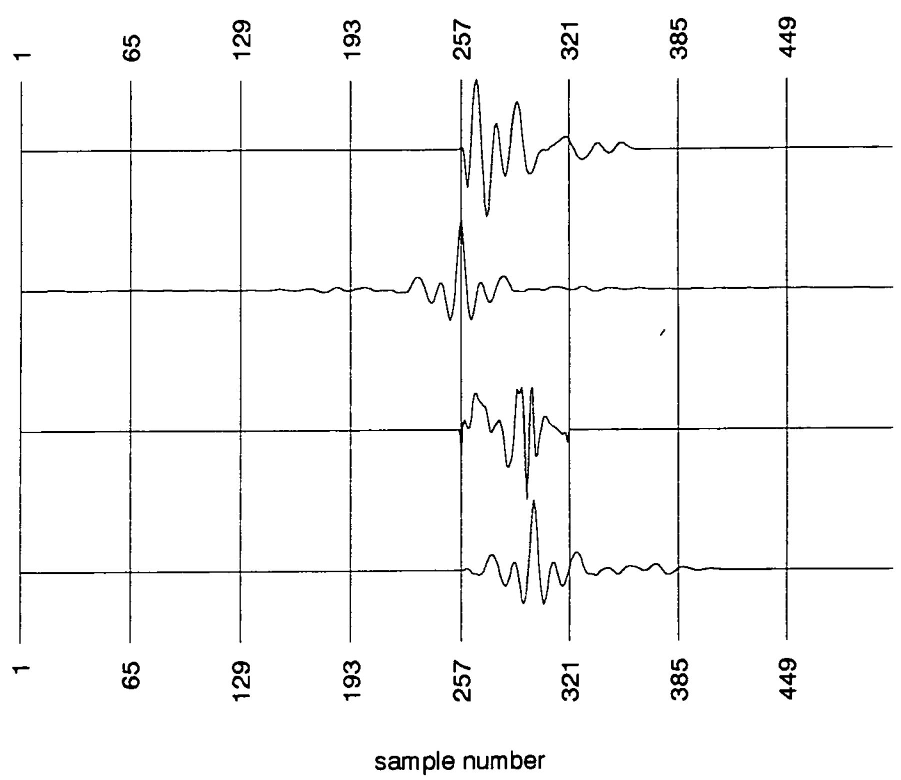

Figure 7.10 From top: input wavelet from aligning and stacking direct arrival through model, desired output with same amplitude spectrum as input spectrum, deconvolution filter, actual output.

• the filters designed using the arrivals through the water (Figures 7.4 and 7.5) left secondary peaks in the data.
• where the output amplitude spectrum was equated to the input spectrum (Figures 7.5 and 7.10), the output wavelet comprises a central peak with two prominent side lobes on each side.
• using a Butterworth as output (Figure 7.4 and 7.9) produced ringing at 250Hz, this being worse for the input trace derived through the model.

As a consequence, merely muting past the first break will prove inadequate for removing the direct arrival, unless a large mute past the first arrival is used to capture the two side lobes or the ringing also. This will also mute the reflected arrivals to an unacceptable degree. An alternative method of direct arrival removal was necessary.

7.3.2 Shaping to maximum phase, muting and inverse maximum phasing.

A novel method of maximum phasing before mute was tried, but proved unsatisfactory. Using the aligned direct arrival as a known wavelet in the seismogram, and making the minimum phase assumption, the spiking deconvolution filter for this wavelet was calculated. As this is the exact inverse of the minimum delay wavelet, it was inverted to give the minimum delay wavelet. By subtracting the phase spectra of the known wavelet and the minimum delay wavelet from that of the seismogram (Figure 7.11), the known wavelet was turned into a maximum phase wavelet in the seismogram (Figure 7.12). Note that if the phase spectrum of the minimum phase wavelet had been added instead of subtracted, the result would have been a minimum phase wavelet in the seismogram (Figure 7.11).

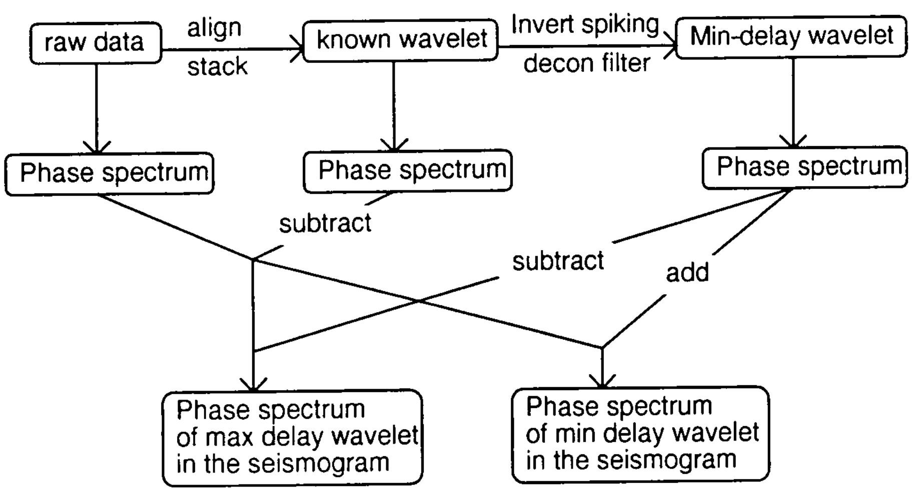

Figure 7.11 Flow chart of the maximum phasing philosophy.

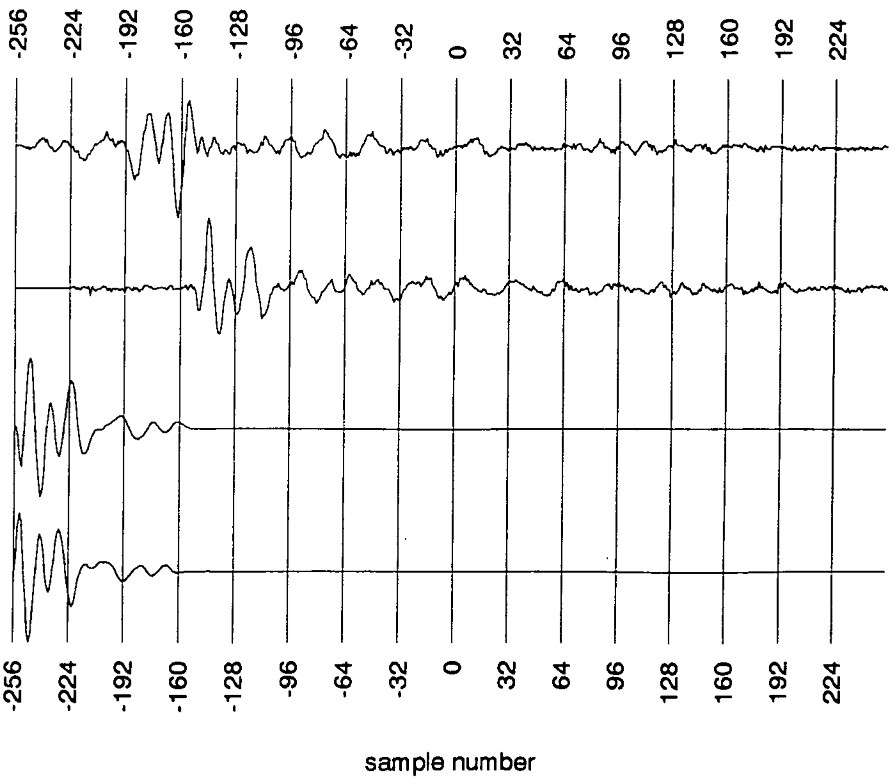

Figure 7.12 Maximum phasing. From bottom: minimum phase source wavelet, source wavelet from aligning and stacking direct wave, example raw trace - source 25 receiver 9, example trace following maximum phasing.

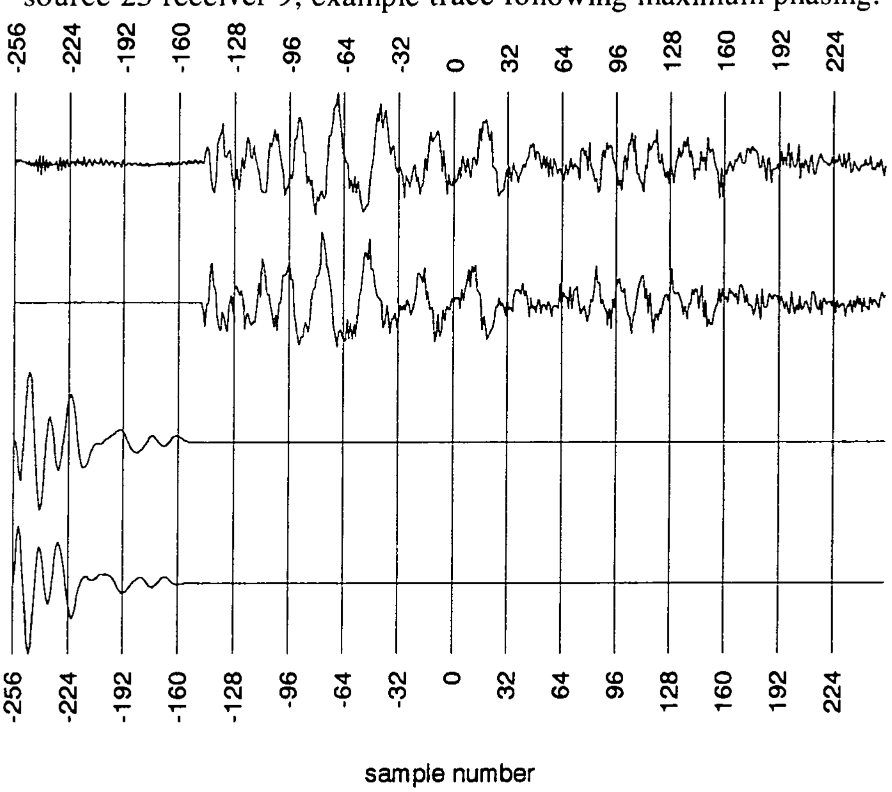

Figure 7.13 Inverse maximum phasing. From bottom: minimum phase source wavelet, source wavelet from aligning and stacking direct wave, example max-phased trace following muting, muted trace following inverse max-phasing.

The advantages of the maximum phase wavelet are clear. With the energy being greatest towards the end of the arrival with a low energy tail preceding it, it should be possible to mute just past the peak of the direct arrival, so removing the direct arrival energy and catching only the low energy tail of the reflected arrival. The inverse of the maximum phasing process was then performed (Figures 7.13 and 7.14), i.e. the phase spectra of the known wavelet and of the minimum-delay wavelet were added to the phase spectrum of the muted maximum-phased seismogram. It is also possible to zero phase the data by removing the influence of the phase spectrum of the minimum phase wavelet.

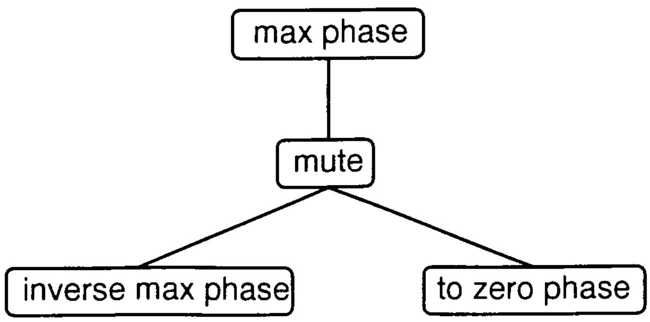

Figure 7.14 Scheme for direct wave removal by maximum phasing.

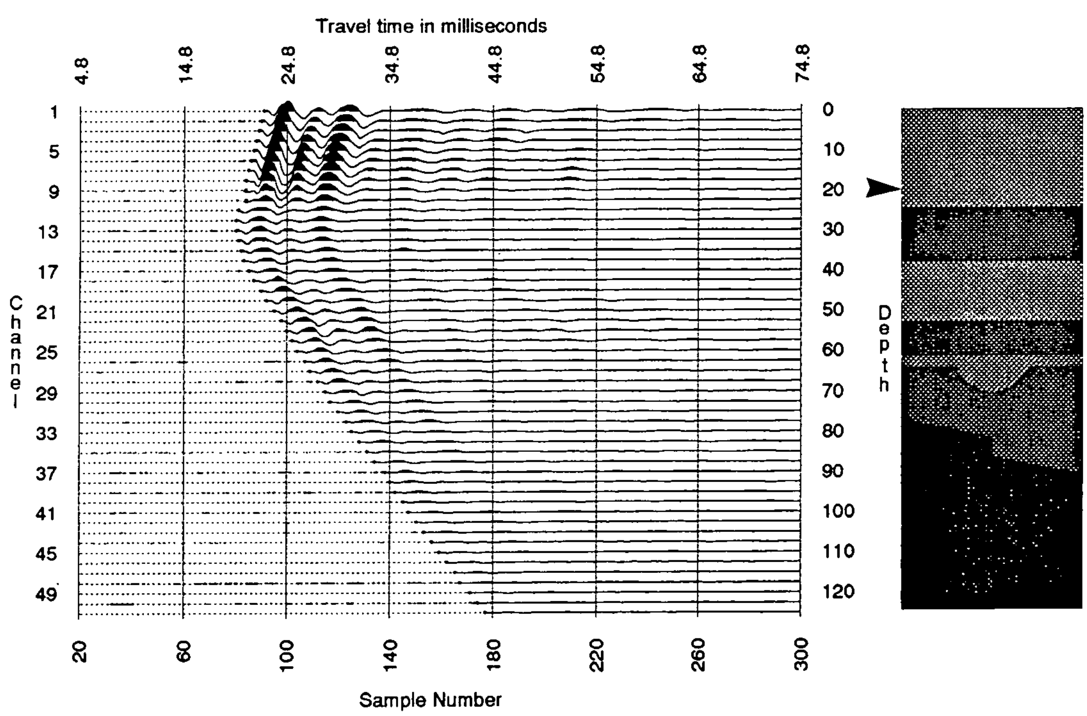

Figure 7.15 Raw data - example CRG from receiver 9 at depth 20m.

The results of max-phasing, muting and inverse max-phasing are illustrated with an example CRG from receiver 9 (depth 20m) in Figures 7.15 and 7.16. At first sight, the process has done a good job of removing the direct wave. However, several problems have arisen:-

- Upon closer inspection of the top two traces in Figure 7.12, it can be seen that residual direct energy has been left to the right of the first break upon max-phasing. It has therefore been necessary to mute past the first break to remove all the direct energy, and, as before, this will result in some reflected energy being removed. This is evidently the case in Figure 7.16, where the effect of the mute in removing energy past the first break is clearly seen.
- Several events can still be seen with the same moveout as the first breaks, e.g. a peak arriving 11.25ms (45 samples) after the first break, and high amplitudes on traces 1-10 arriving 3.75ms (15 samples) later.
- The process has introduced high frequency noise.
- A further problem is that the max-phasing technique cannot be employed for removing the shear wave arrivals, which have become prominent following the removal of the direct P-waves (Figure 7.16).

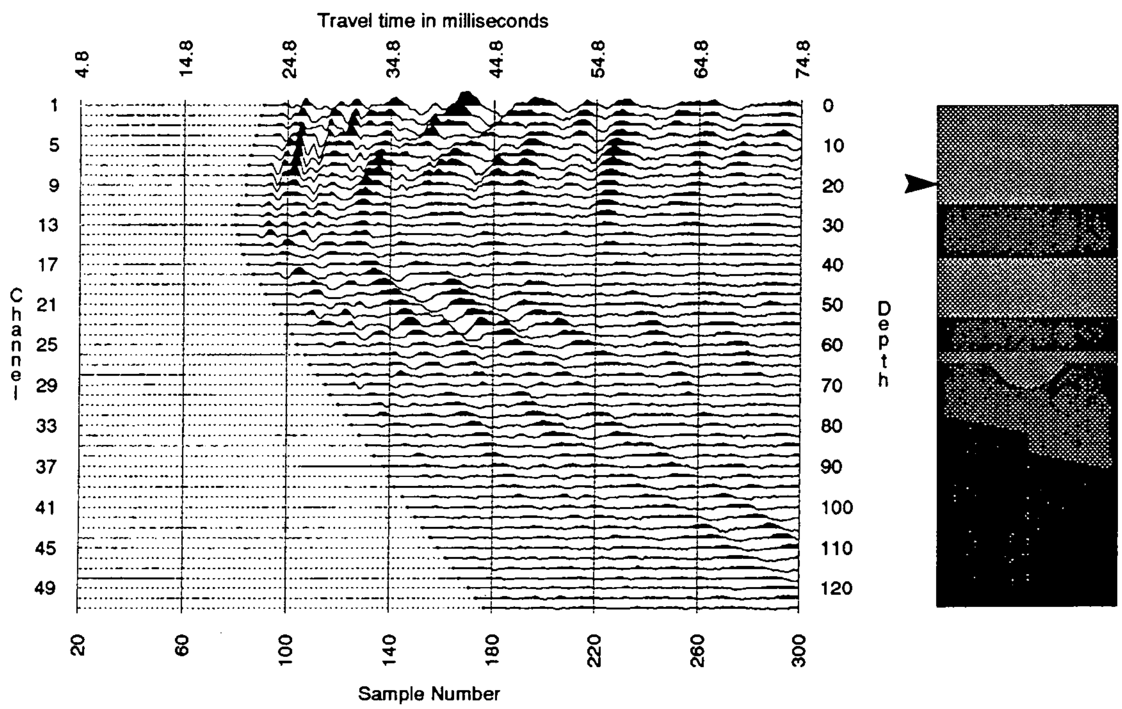

Figure 7.16 CRG 9 following max-phasing, muting and inverse max-phasing.

7.3.3 Common Ray Angle Gather

The method used by Pratt and Goulty (1991), processing the same dataset, was then tested. This method has been discussed in detail in §3.2.4 and §3.3.4. In order to estimate the direct wavefield, they regathered the data into common ray angle gathers (CRAGs), this reorganisation of the data allowing scattered (i.e.

reflected) and direct arrivals to be discriminated on the basis of time moveout. For each trace that was to be operated on, 11 common ray-angle gathers were selected (Figure 7.17).

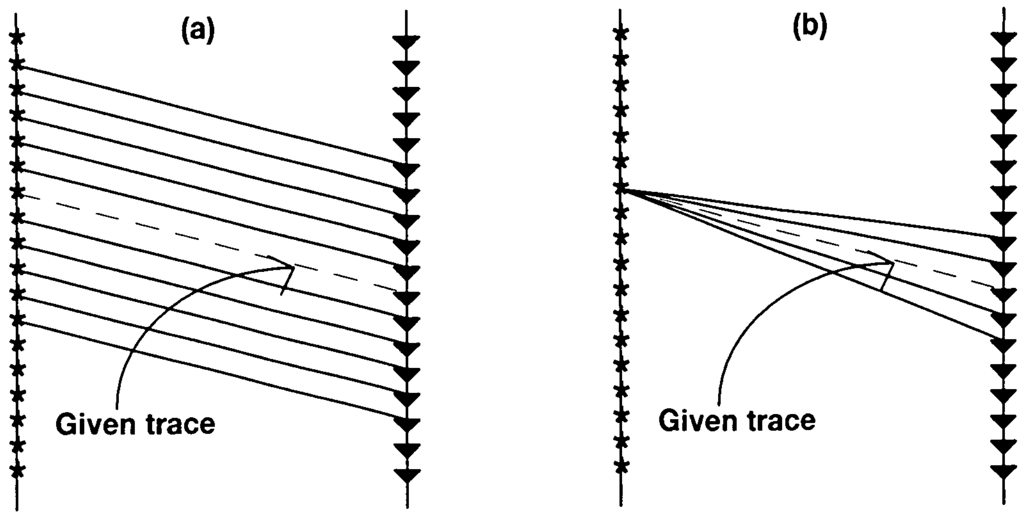

Figure 7.17 Traces contributing to the common ray angle gather - (a) Given ray trace and ten adjacent traces with same ray angle. (b) Given ray trace and four adjacent traces from same gather. These five traces are included from each gather in (a).

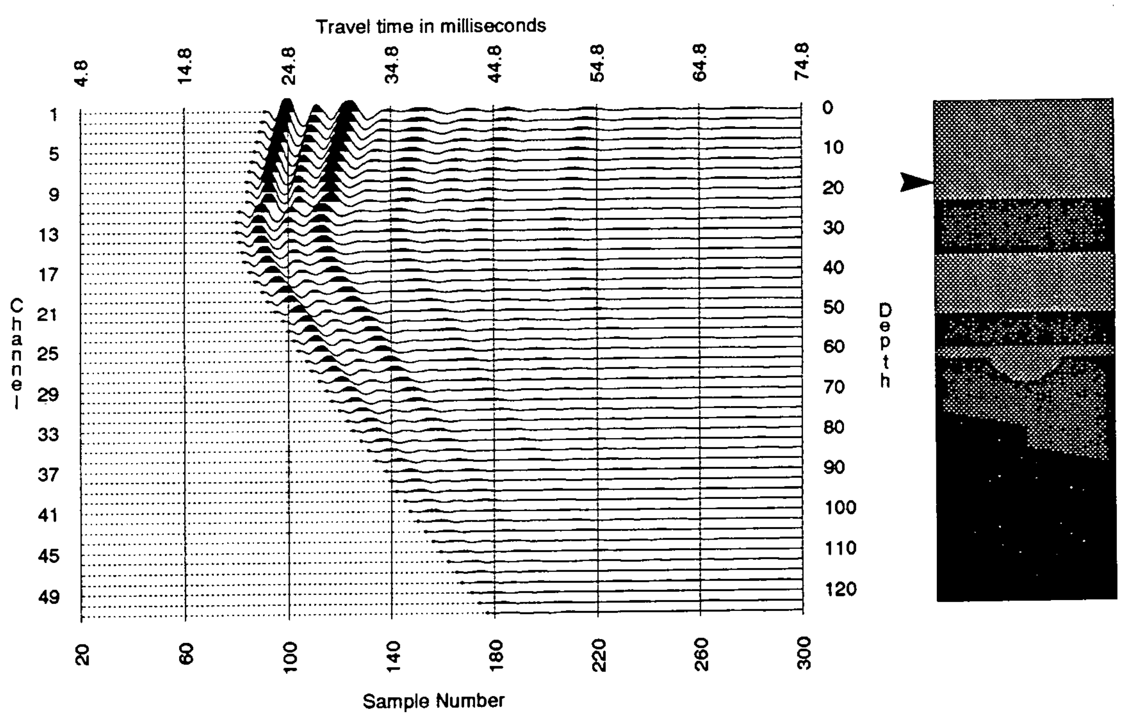

Figure 7.18 CRG 9 - direct wavefield estimate from CRAG (compare with Figure 7.15).

These were the gathers containing the given trace and the five adjacent ones on either side. From each of these gathers, five traces were selected: the traces with the same ray angle as the given trace, and the two traces on either side. Thus a total of 55 input traces were used for each output trace (except at the edges of the survey). The direct arrival first breaks were aligned and averaged by means of a running mean filter. This then gave an estimate of the direct wavefield for all 2601 traces (Figure 7.18).

This direct wave estimate provided a robust way of deconvolving the data. A Wiener shaping filter was computed and applied for each trace in the raw wavefield by using the equivalent estimate of the direct arrival as input (see §3.2.4). The desired output was a zero phase wavelet with the same amplitude spectrum as the input wave. The deconvolved data are shown in Figure 7.19.

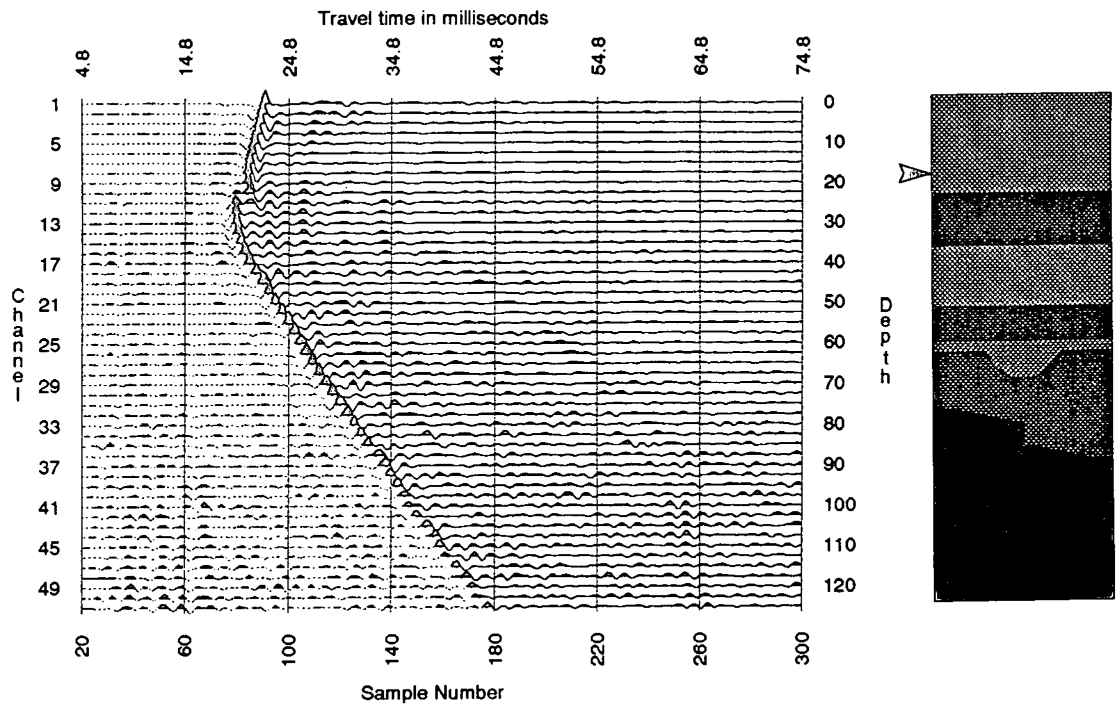

Figure 7.19 CRG 9 - trace-by-trace zero-phase deconvolved using direct wavefield estimate in Figure 7.18 as input.

Having deconvolved the data, two schemes for removing the direct wavefield were tried. These have been discussed in §3.3.4 and are illustrated in Figures 3.3 and 3.4. The second of these schemes produced the gather in Figure 7.20. The CRAG method was a great improvement on the methods for direct wavefield removal discussed previously, especially as it avoided all use of muting which had removed the desired reflected arrivals in previous methods. However,

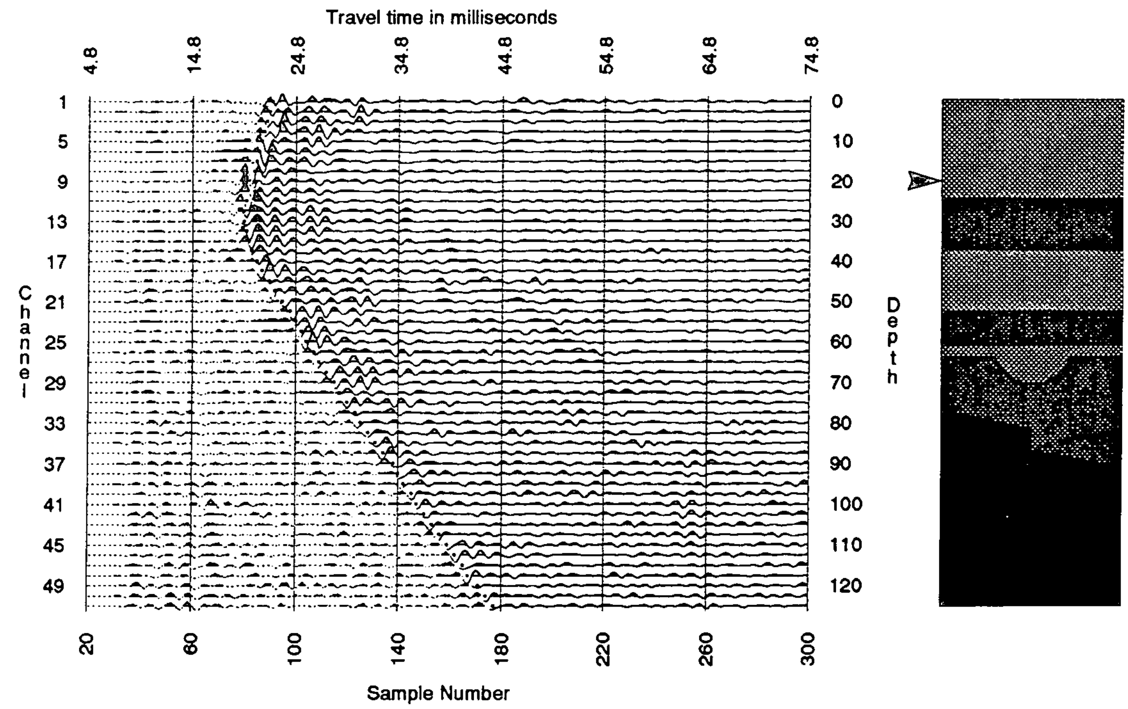

Figure 7.20 CRG 9 - zero-phase reflected wavefield obtained by subtracting zero-phase CRAG wavefield from zero-phase total wavefield (see Figure 3.4).

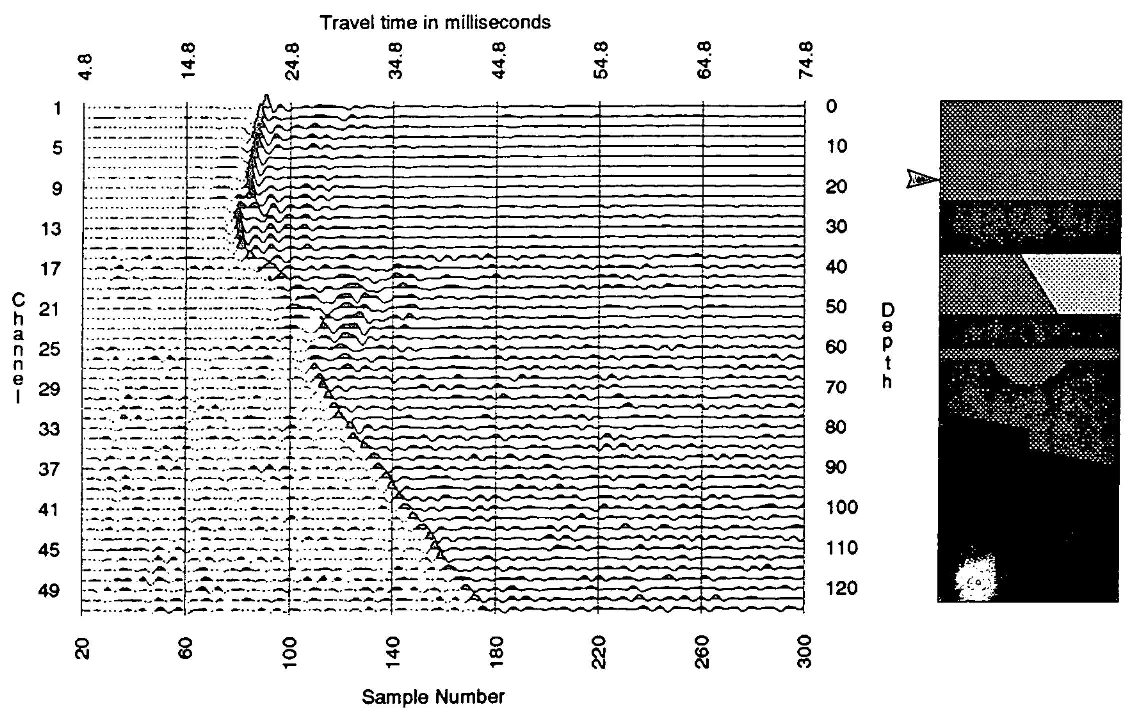

Figure 7.21 CRG 9 for the post-flood - trace-by-trace zero-phase deconvolved using pre-flood direct wavefield estimate in Figure 7.18 as input.

high-cut filtering and a short mute may be considered necessary for removing the high frequency noise and the noise around the first break where the direct wavefield has not been perfectly removed in Figure 7.20.

The CRAG method is less successful for the post-flood data because of the inconsistency of adjacent traces around the flood zone. This is partly due to high attenuation and partly due to multipathing of the direct arrival. This propagates through the processing sequence to produce ringing when the shot or receiver is located in the proximity of the flood zone. The situation is greatly improved by using the pre-flood CRAG data for deconvolving the post-flood data (Figure 7.21). The effect of multipathing in the flood zone could be removed by applying a severe mute before migration for sources positioned adjacent to the flood zone. This will mean that the reflector segments adjacent to the source array and bounding the flood zone will not be imaged from within the zone. However, these segments will be imaged from above/below the zone.

7.3.4 Deconvolution only (Direct removed by Wavefield Separation)

This was the preferred method of deconvolution. It is noted that the CRAG trace-by-trace deconvolution scheme above, is ray angle specific to rays travelling in a straight line between source and receiver, and hence inappropriate for scattered arrivals which could arrive at the receiver at any angle. Instead, a

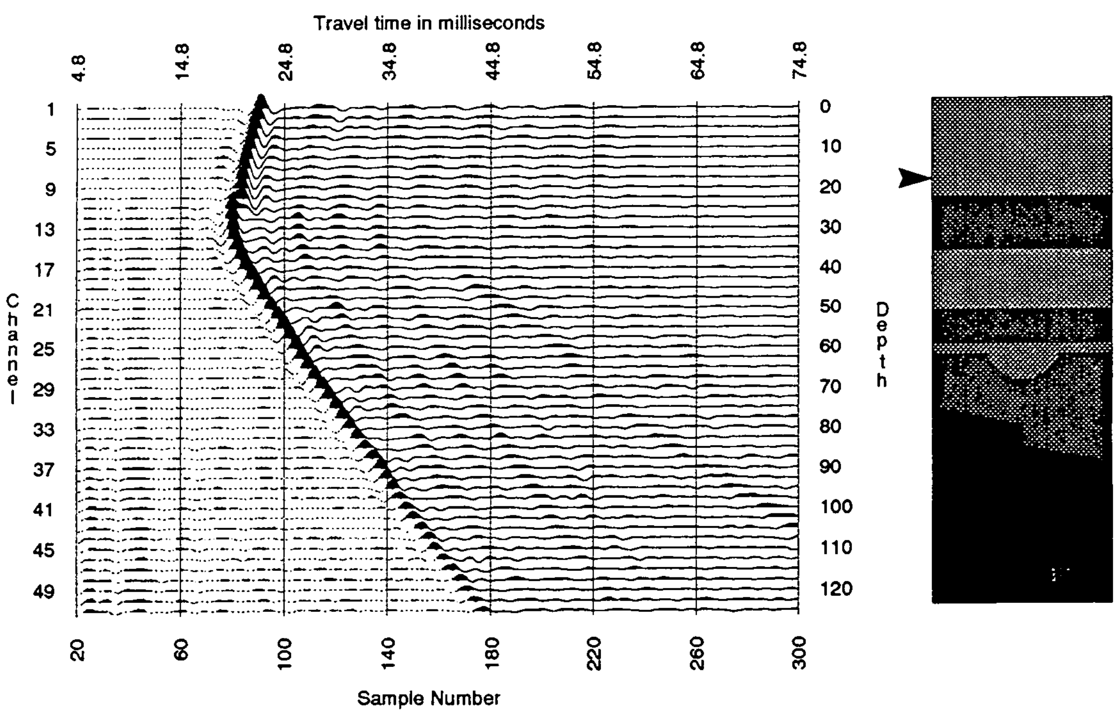

Figure 7.22 Zero-phase deconvolved CRG 9. Compare with the raw gather in Figure 7.15.

deconvolution filter was designed for the measured source signature through the model averaged over a range of take-off angles (as done in §7.3.1), as the transducers did not display much directivity. However, directivity effects could also be accounted for by applying a directional deconvolution filter in the *f-k* domain (Hubbard et al. 1984), or in the $f\text{-}k_s\text{-}k_r$ domain. The desired output was a 150-500Hz Butterworth wavelet. The direct arrival was removed by wavefield separation (§7.4.3). A deconvolved pre-flood gather is shown as Figure 7.22.

7.4 Processing - Wavefield separation

7.4.1 2-D *f-k* filtering

The basic method for wavefield separation described in §3.4 was the first to be tested. The data were transformed in CRGs to the *f-k* domain, following spatial tapering of the edge traces. Pie slice filters were applied to extract the upgoing and downgoing wavefields. Before transformation back to the *x-t* domain, edge wavenumbers were tapered. Time and frequency tapers were unnecessary (see §3.4.2).

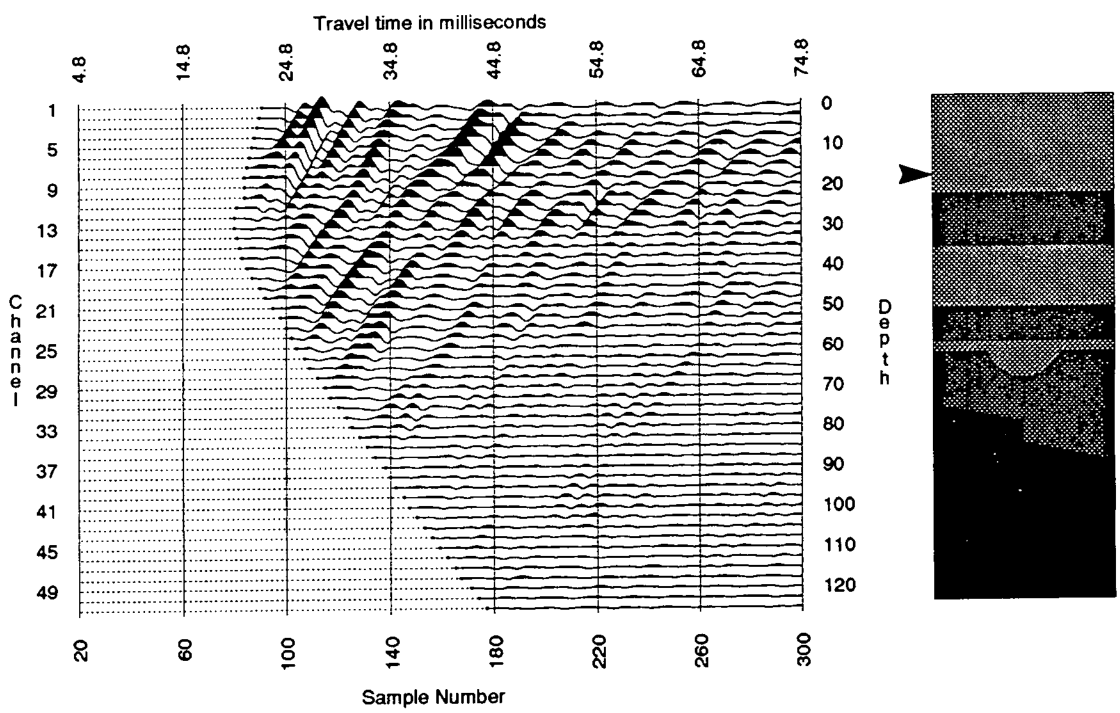

Figure 7.23 CRG 9 following wavefield separation by 2-D *f-k* filtering of zero-phase reflected wavefield in CRGs (Figure 7.22).

Several problems were encountered with this method. 2-D *f-k* filtering is a non-reciprocal process. Simply put, *f-k* filtering in CSGs is not equivalent to *f-k* filtering in CRGs. The effect of this is to produce non-symmetry of image quality in the final migrated sections. The upgoing wavefield (Figure 7.23), separated in CRGs, is re-sorted into CSGs, where the non-reciprocity becomes apparent (Figure 7.24), and reflected arrivals are much less clear. The migrated image will therefore be better resolved on the shot array side of the image which is imaged by the CRGs, whereas the receiver array side of the image, imaged by the CSGs, shows poor imaging.

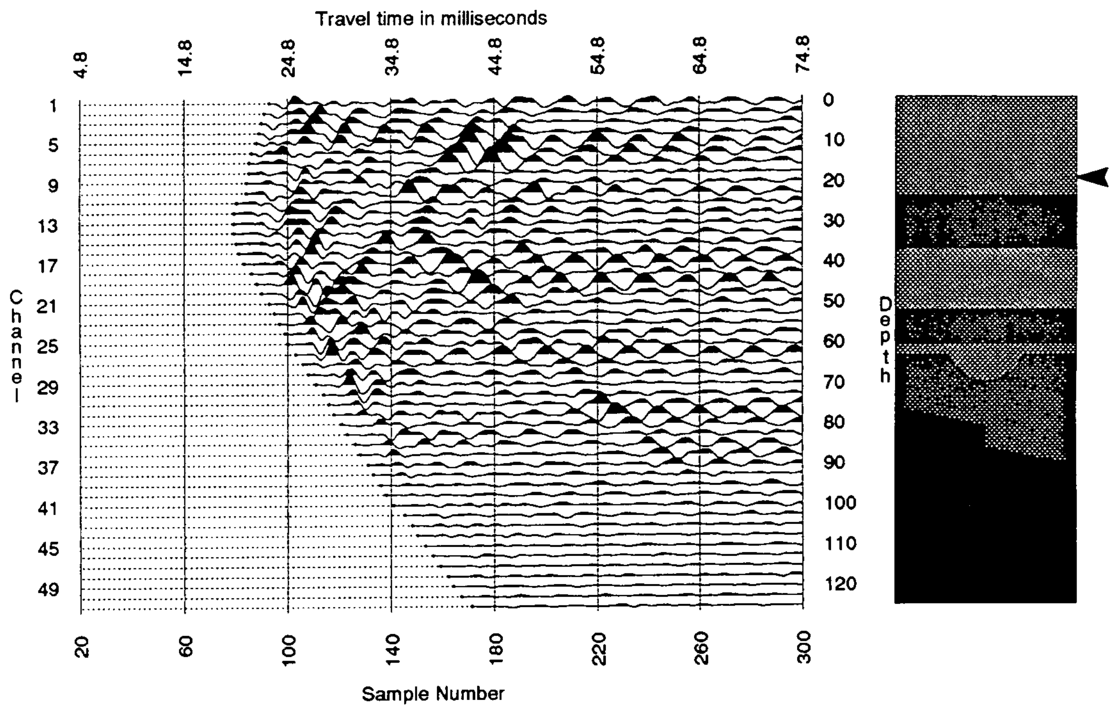

Figure 7.24 CSG 9 following wavefield separation of zero-phase reflected wavefield in CRGs (compare with Figure 7.23).

Performing *f-k* filtering in CRGs followed by resorting and *f-k* filtering in CSGs may be thought to be the solution to this problem. Alternatively, processing in CRGs to image the shot side of the image, and in CSGs to image the receiver side could be considered (§7.4.2). However, 3-D *f-k-k* filtering is presented as a more satisfactory solution (§7.4.3).

A further problem with 2-D *f-k* filtering has been highlighted when the separated CRGs are displayed in CSGs. In Figure 7.24, the S-wave arrival has become more apparent, and by comparing back to Figure 7.23, it is clear that S-wave direct arrivals have been caught up in the *f-k* filter and are interfering with the reflections from about 60-70m depth (see §5.2.1 for discussion of why downgoing waves are seen in the CSGs). It would not be possible to remove the S-waves by raising the low-cut velocity of the pass region as the S arrivals have a range of apparent velocities up to infinity in the crosshole case. The S-waves will appear as curving noise on the migrated depth sections at depths 50-60m below the receiver location, and on the receiver side of the section. The noise is curving since the S-wave has a lower apparent velocity than the reflection at the same position in *x-t* space. This is shown in Figure 7.25 where only the top ten CRGs (depths 0-22.5m) have been migrated.

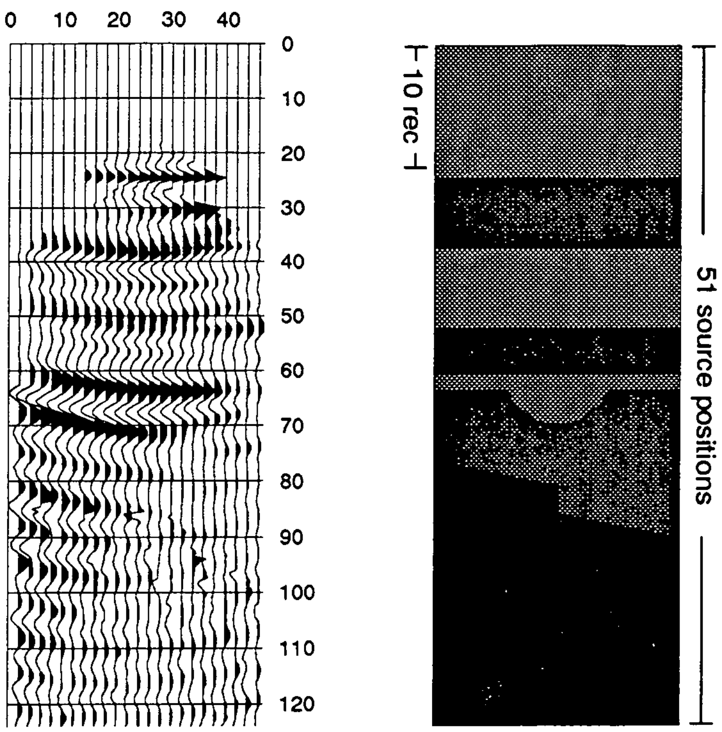

Figure 7.25 Migrated depth section of the top 10 CRGs (depths 0-22.5m) following 2-D *f-k* filtering in CRGs. Note the curved noise to the left of the section below 60m depth. Linear ramp with depth applied.

To remove the shear arrivals, Pratt and Goulty (1991) have suggested using the CRAG method with shear wave first arrival times. However, the accurate picking of S-wave first arrivals is not an easy task, especially for sub-horizontal raypaths (see for example, Figure 7.16). Pratt et al. (1991) and Li and Worthington (1990) suggest the use of recursive dip filtering (Hale and Claerbout 1983) to remove the direct, scattered or mode-converted S-wave energy. Also considered was the muting to zero of all energy past the S-wave arrival for all traces, but this was deemed to be somewhat brutal as it removed reflected arrivals also. Instead, the use of 3-D *f-k-k* filtering (see §7.4.3) is advocated for S-wave removal.

7.4.2 2-D *f-k* filtering in CSGs and CRGs for imaging different sides

To address the problem of poorer image quality on the receiver side of the migrated section following wavefield separation in CRGs, the data were *f-k* filtered separately in both CSGs and CRGs to produce two datasets each for the upgoing and downgoing wavefields. The migration program was therefore adapted to read in data from the CSG separated dataset for image points in the receiver borehole half of the migrated section, and from the CRG separated dataset for image points in the source borehole half. This method is similar to that used by Lazaratos et al (1992). The migrated images (Figure 7.26) are still affected by the S-wave noise problem, and 3-D *f-k-k* filtering (§7.4.3) is preferred.

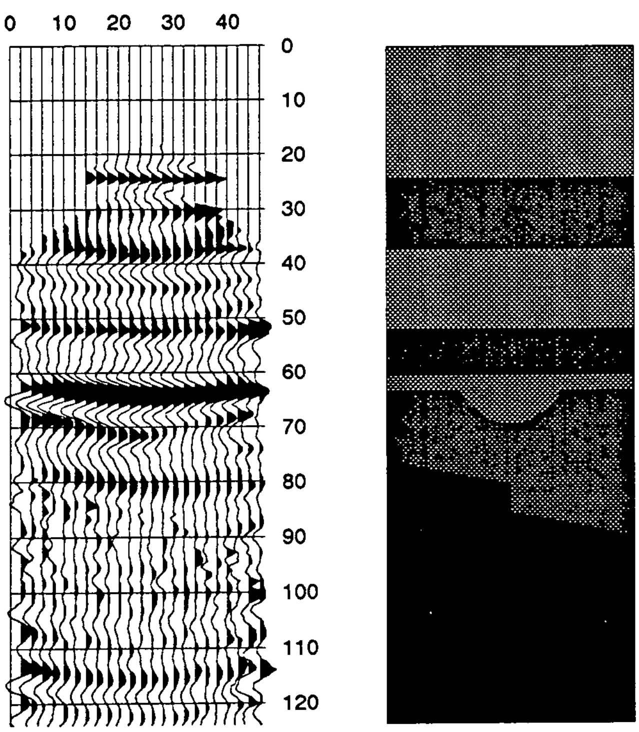

Figure 7.26 Migrated depth section following 2-D *f-k* filtering in CSGs (left half of section) and in CRGs (right half). Traces normalised - linear depth ramp.

7.4.3 3-D *f-k-k* filtering

This method has been described fully in §5.2. Here comparisons with the above methods and the specific advantages of the technique as applied to the tank dataset are discussed.

One main advantage of this method is that not only does it separate the upgoing and downgoing reflections, but also discriminates between reflected and direct arrivals (both P and S) as they lie in different k_s-k_r quadrants of f-k_s-k_r space (see §5.2.3). A mute of the high amplitude direct arrival, which would also remove reflected energy lying close behind the direct arrival, is therefore unnecessary. Examples of the data quality following zero-phase deconvolution (§7.3.4) and selection of the upgoing reflected wavefield by 3-D *f-k-k* filtering are given as CRG 9 (Figure 7.27) and CSG 9 (Figure 7.28) for comparison with Figures 7.23 and 7.24, respectively. Notice how the two gathers are now of similar data quality, how the S direct arrival has been removed from the data, and how the reflected arrivals reach the first break, not having been caught in a direct arrival mute.

An advantage of 3-D *f-k-k* filtering over two-pass 2-D *f-k* filtering (*f-k* filter in CRGs, then in CSGs) is that the wavefield separation is performed in one pass, requiring transformation into and out of the *f-k* domain, and hence edge tapering, only once.

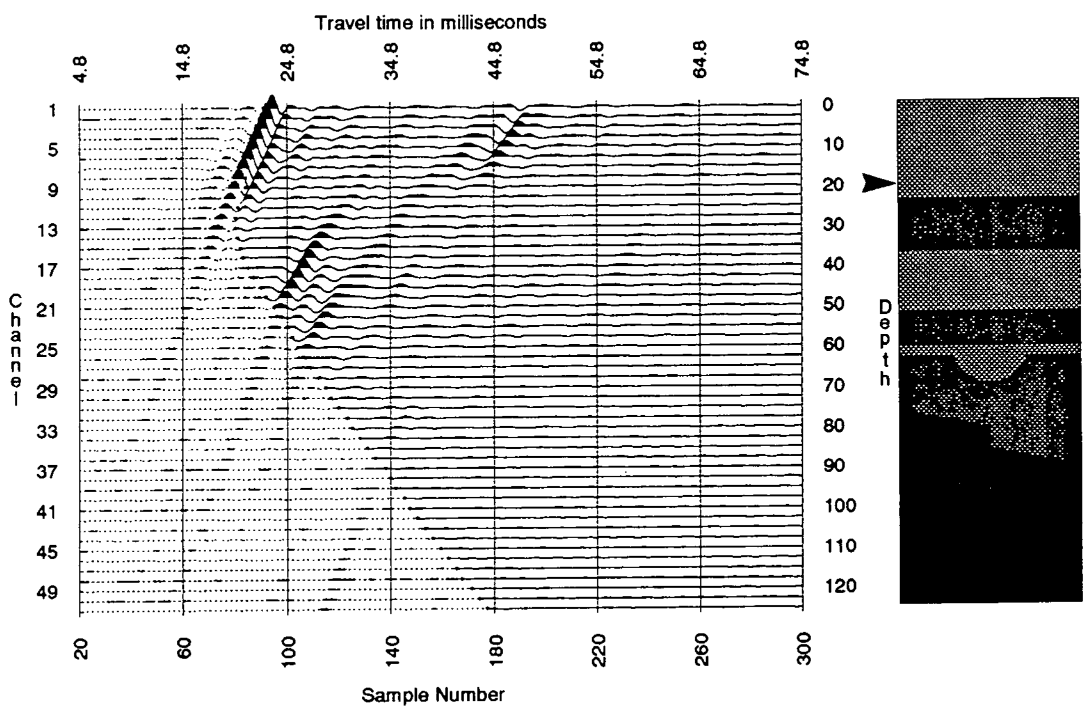

Figure 7.27 CRG 9 following wavefield separation by 3-D *f-k-k* filtering of zero-phase wavefield in Figure 7.22 (compare with Figure 7.23).

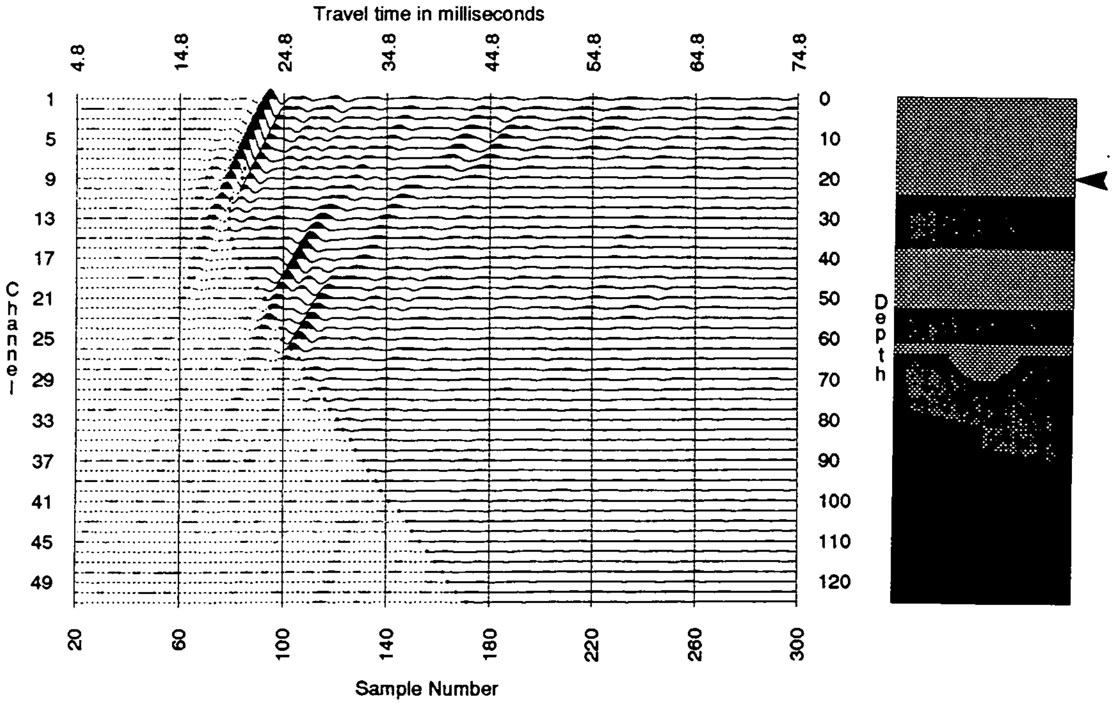

Figure 7.28 CSG 9 following wavefield separation by 3-D *f-k-k* filtering of zero-phase wavefield in Figure 7.22 (compare with Figure 7.24).

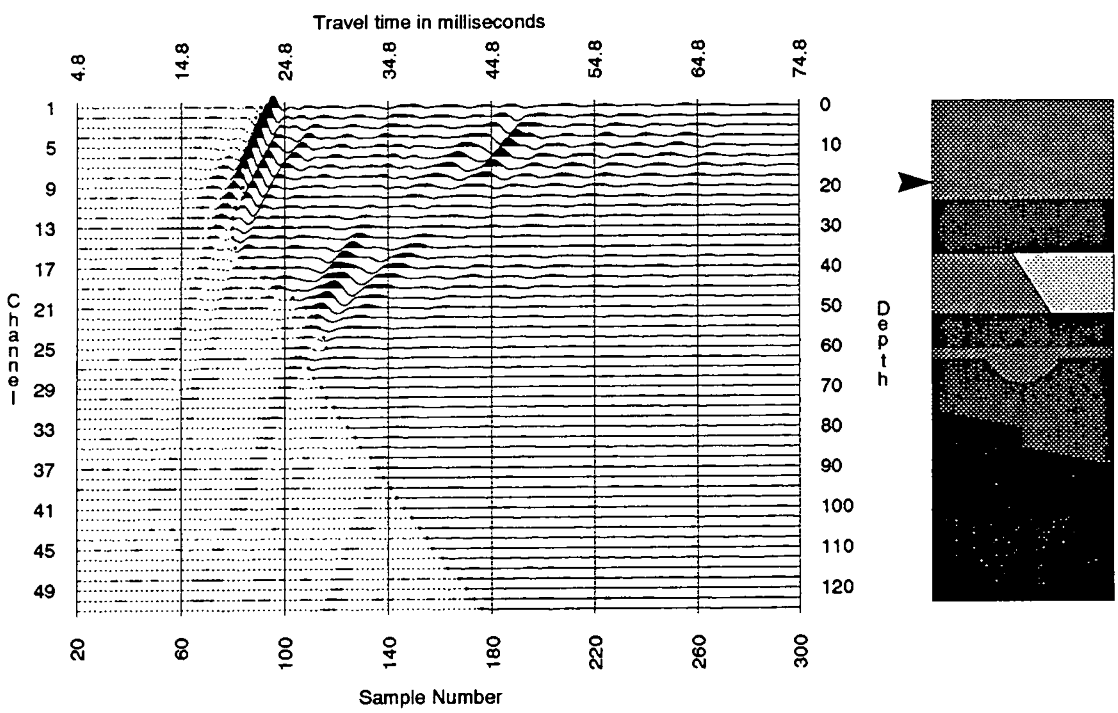

Figure 7.29 CRG 9 following wavefield separation by 3-D *f-k-k* filtering of post-flood zero-phase wavefield.

7.4.4 3-D *f-k-k* filtering with muting for the post-flood wavefield

Head waves existed in the post-flood dataset (see §7.2.5). These have the same moveout characteristics as reflected arrivals: i.e. downgoing on receiver array, upgoing on source array and vice versa. They will therefore still be present in the wavefield separated datasets (energy on traces 20-25 at sample numbers 105-115 in Figure 7.29), and appear as noise on the migrated sections (Figure 7.30). In order to reduce this problem, muting of the source and receiver traces level with the flood zone was performed, with muting past the first arrivals performed for traces above and below.

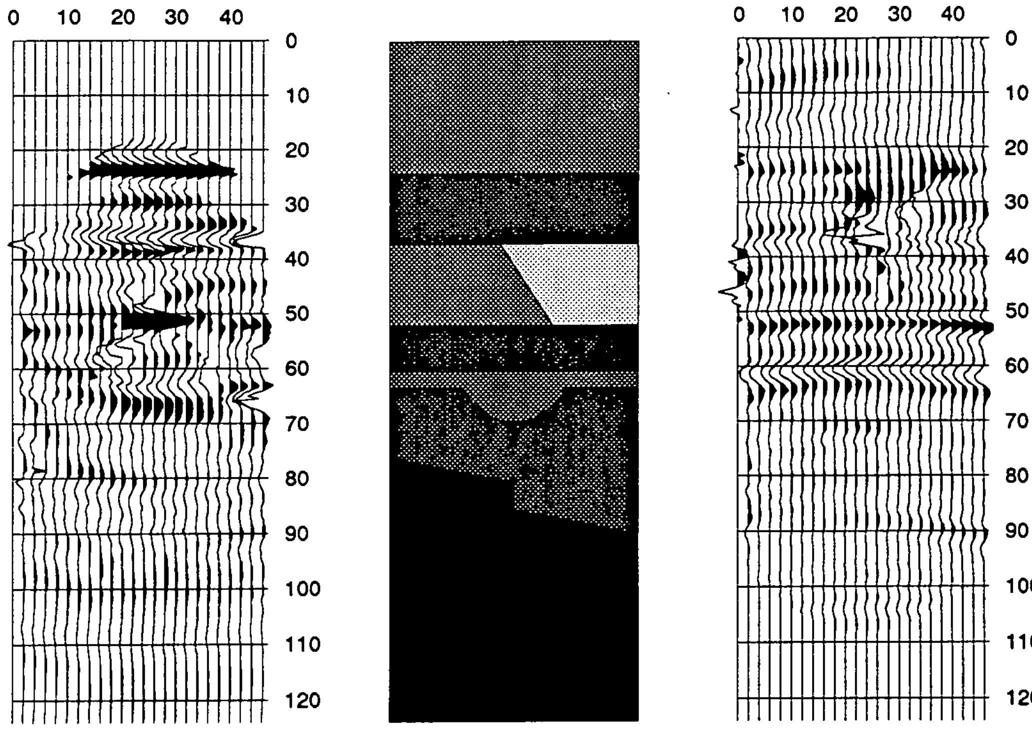

Figure 7.30 Up and downgoing migrated depth section following 3-D *f-k-k* filtering of post-flood data (Figure 7.29). Notice the large amplitude noise below 50m depth on the upgoing and above 40m depth on the downgoing section. Linear ramp with depth applied.

7.5 Processing - Migration

Both GK and GB migration (§5.1), which are Kirchhoff integral migration methods, have been performed on the wavefield separated datasets. In addition, two raytracing methods have been used for obtaining traveltimes and distances from image points to sources and receivers.

7.5.1 Static corrections

As discussed in §2.2.2, the source signal was generated on a cylindrical source, and to regard this as a point source would require a static shift increasing traveltimes by 9.54/9.52 samples (pre/post-flood). A further static shift is required for the first raytracing method (§7.5.3), since the source and receiver positions are assumed to be at the edge of the model. This static shift reduces the traveltime of the rays by the time taken for the rays to travel from the active elements of the piezoelectric elements to the side of the model through the water. Again from §2.2.2, this would amount to 21.81/21.66 samples (pre/post-flood), giving a combined reduction of traveltime 12.27/12.14 samples.

7.5.2 Positioning errors of the source and receiver arrays

By considering the traveltimes of the arrivals through the water with the model removed, it was determined that the source and receiver arrays were not exactly parallel (Leggett 1992). The deviation calculated was from 54.1m horizontal separation of source and receiver at the top of the model to 55.1m separation at the bottom, with the source array being 1m lower than that of the receiver array. By inspection of the data shot through the model, it was also calculated that the model was placed 1.5m higher than intended with respect to the arrays. The deviation was applied to the source array for ease of raytracing, making the end coordinates of the receiver and source arrays (0.0,1.5)→(0.0,126.5) and (54.1,2.5)→(55.1,127.5), respectively, where the top of the model was at 0.0m depth. These positioning errors are not surprising when it is noted that the deviation amounts to a 1mm error over a distance of 125mm in the tank.

7.5.3 Raytracing - Horizontal layers

The first method of raytracing was that used for the Coal Measures and Groningen datasets in this study. A horizontal layered velocity model is defined; different horizontal and vertical velocities can be defined to model anisotropic media, though since the epoxy resin layers were uniform, isotropic layers were assumed. The migration plane is covered by a user-defined grid of image points. Each image point is considered in turn, and raytracing from each receiver position to the point and from each source position provides a total travel time and distance travelled for each source-receiver combination, together with the angles of take-off from source and receiver and incidence at the image point.

7.5.4 Raytracing - Boxel method

This second method was adapted from the raytracing method used by Leggett et al. (1993) for tomography. A grid of 'boxels' is defined, each with an associated velocity. Refraction calculations using Snell's law therefore are performed at each cell boundary. As above, a grid of image points is considered in turn, and raytracing provides traveltime, distance travelled and angles. It was found necessary to use this second more complex method of raytracing because of the geometry of the experiment (Figure 2.3). A simple static correction to reposition the source and receiver at the edge of the model will fail because:-

(a) the static correction will vary between rays, since rays leave the source and receiver at different angles for different image points and hence travel different distances through the water.

(b) the entry points into the model (i.e. the positions to which ideally the static shift will move the source to) will depend on the ray angle. The boxel raytracing method overcomes these problems as it is possible to define a thin water layer velocity down the sides of the model to raytrace through.

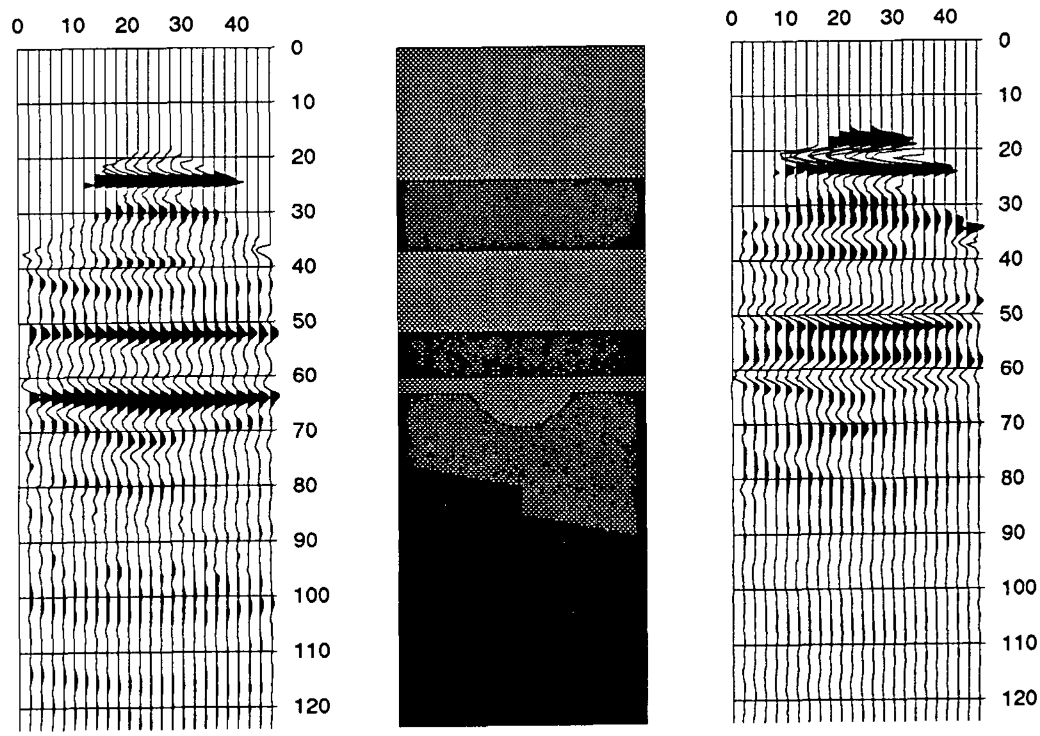

Figure 7.31 Upgoing depth sections migrated through (left) a boxel velocity model and (right) a layered velocity model. Linear ramp with depth applied.

A comparison between migrated images produced by identical processing schemes except for the method of raytracing (Figure 7.31), emphasises the advantages of the second method of raytracing. The boxel raytracing image shows a much better resolution implying that the rays have more precise travel path parameters.

On a practical note, the adaptations to the raytracing program used by Leggett et al. (1993) were as follows:-

(a) Tomography requires shooting from source to receiver, hence the program was written to shoot only from left to right i.e. increasing in the horizontal coordinate. For reflection imaging, it was therefore necessary to call the program once for the source positions, reverse the receiver and image point horizontal coordinate about the vertical bisector of the model, and call the program again for the receiver-image point ray. Angles would obviously need to be flipped by $\pi/2$ for this receiver ray.

(b) The ray capture criterion specified in the tomography program is for the ray to pass within a certain vertical distance of the receiver position. By default, this criterion would therefore be relative to the image point. However, if we consider a typical raypath, it is realised that the rays are more horizontal in the water layer in the vicinity of the source and receiver than after being refracted into the model. To assist capture of the rays therefore, shooting was performed from image points to source and receivers, and angles adapted accordingly.

7.5.5 Estimation of the velocity field

Several methods were used in conjunction to estimate the velocity field to raytrace through. For the pre-flood data, Leggett's work (Leggett et al. 1993) was followed closely. It was a simple matter to assume a 1-D velocity field within the model, except for the sloping boundary which was assumed linear, with layers of constant velocities derived from the tomogram and from horizontally travelling raypaths. Of course this method would not account for either the fault on the dipping interface, or the layer containing the channel feature, which is seen as a thin low-velocity layer at the boreholes. The raw tomogram data was not used directly to raytrace through because of the possibility of the existence of artefacts within the tomogram and because it was sensible to constrain the velocity field with all the known information, namely the hypothetical 'sonic log' run in each 'borehole'. In a field environment, the obvious information to use would have been the sonic log data together with uphole shots, though this would have proved too difficult to collect for the tank model, bearing in mind the static problems of the source and receiver.

The post-flood model proved more of a problem to create a velocity model for. If a sonic log had been run down the sides of the model, the same velocity field as for the pre-flood would have been created except for the anomalous velocity detected adjacent to the flood zone. The reservoir layer could then be modelled in several ways:-

(i) A homogeneous layer of the average velocity of the flooded and non-flooded reservoir.

(ii) As two rectangular blocks of the extreme velocities with a vertical interface in the middle of the model.

(iii) As two blocks of the extreme velocities with the correct sloping interface in the middle of the model.

(iv) A gradational velocity change from the two extreme velocities measured at the sides of the model.

(v) As obtained from direct arrival traveltime tomography.

However, it should be borne in mind that sonic logs were not run, and so it was necessary to estimate the velocities of the media from the tomography results and from the horizontal raypaths. The tomography of Leggett was found to be in error as the earlier arrival times of head wave arrivals (§7.2.5) had been picked and inverted instead of the direct arrivals. This would cause the tomogram to overestimate the velocity of the flood zone. The actual flood zone velocity was estimated by correctly picking the direct arrival through the flood zone, and taking account of refraction at all interfaces when considering the sub-horizontal raypaths. The tomography was rerun with improved picks (§7.2.5), using the starting models (ii), (iii) and (iv) mentioned above. Note that these velocity models have also been compared in the migration scheme (§7.6.2). The resultant tomograms (Figure 7.32) are displayed after 20 iterations, and no smoothing (Krajewski et al. 1989) has been applied between iterations. For details of the tomography programs used, refer to Leggett (1992).

Several points are raised by the tomography results. Firstly, the difference between the tomograms obtained with the corrected picks and that of Leggett's (1992) is very clear, with the flood zone velocity for the corrected picks being lower. The second point is that the choice of initial velocity model can be crucial to the results. This is evident in Figure 7.32, where the expression of the initial velocity model has been kept in the final tomogram.

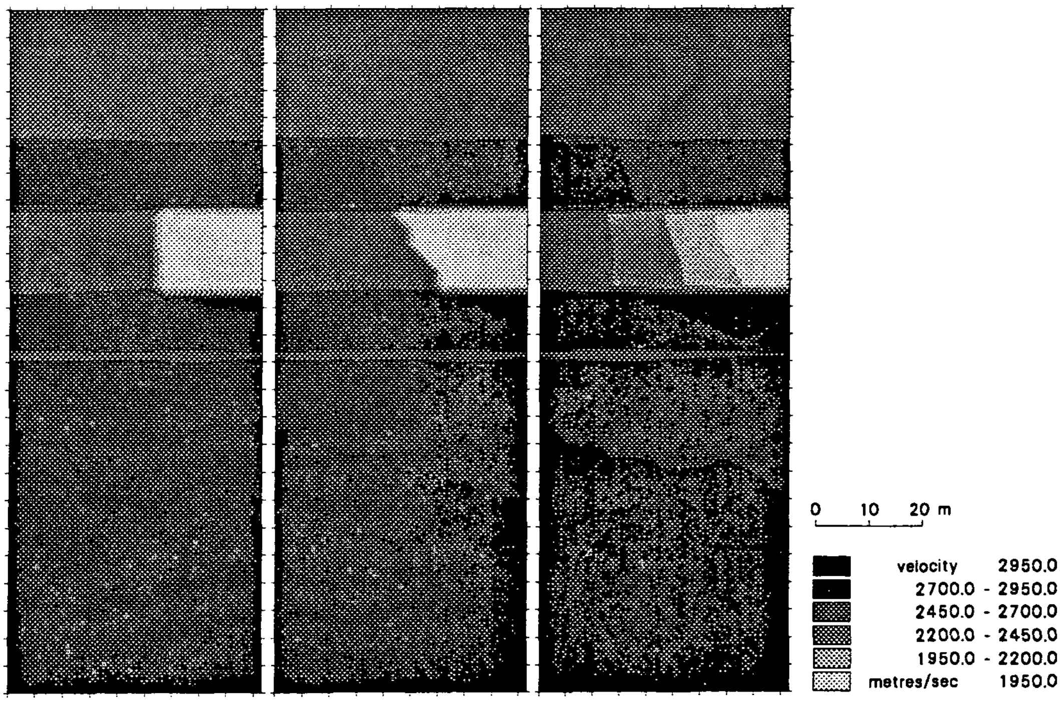

Figure 7.32 Velocity tomograms obtained with improved traveltime picks using initial velocity models (ii), (iii), (iv).

By using the pre-flood model as the initial velocity field, the velocity of the flood zone is greatly overestimated. This has major implications if tomography alone is used for estimating the velocity field without the use of sonic logs. For both the block models as initial velocity estimates, the shape and velocity of the flood zone has been largely unaffected by the tomographic process. One reason for this is that the sloping interface of the flood zone has been inadequately modelled by the cell raytracing approach in both cases so that non-physical rays have been traced, such as a horizontally travelling ray which should be refracted by the sloping boundary but passes undeviated through a vertical interface. Using the gradational velocity model as the initial estimate, the shape and velocity of the flood zone has developed most clearly, though the remnant gradational nature is still present in the tomogram.

The preceding paragraph emphasises the need to be wary of tomography results, and to use common sense in creating the initial velocity field. The geometry of the flood zone can be obtained from the tomograms, and the velocity estimated from near horizontal raypaths with refraction effects at the sloping interface taken into account. Of course the reflection imaging should also be used as a feedback process for updating the velocity models, since the stacking of reflected arrivals can be checked, and their depth are known from their intersection with the direct arrivals.

7.5.6 GK or GB algorithm?

This question has been fully covered in §5.1. One of the salient points relating to the tank dataset is how large the near-field is considered to be. With a signal bandwidth of 50-500kHz, a model velocity of 1750-3000m/s, the borehole separation of approximately 50mm is equivalent to 1-15 wavelengths. Assuming the near-field to be within one wavelength, it is noted that the sides of the model are in the near-field for some frequencies. The GB algorithm which includes the near-field term is therefore favoured. A comparison of the effect of the two algorithms is shown in Figure 7.33, where migrated images have been produced by identical processing routes apart from the algorithm used. As can be seen, the two images display only slight differences, although side traces should be truer with the GB image.

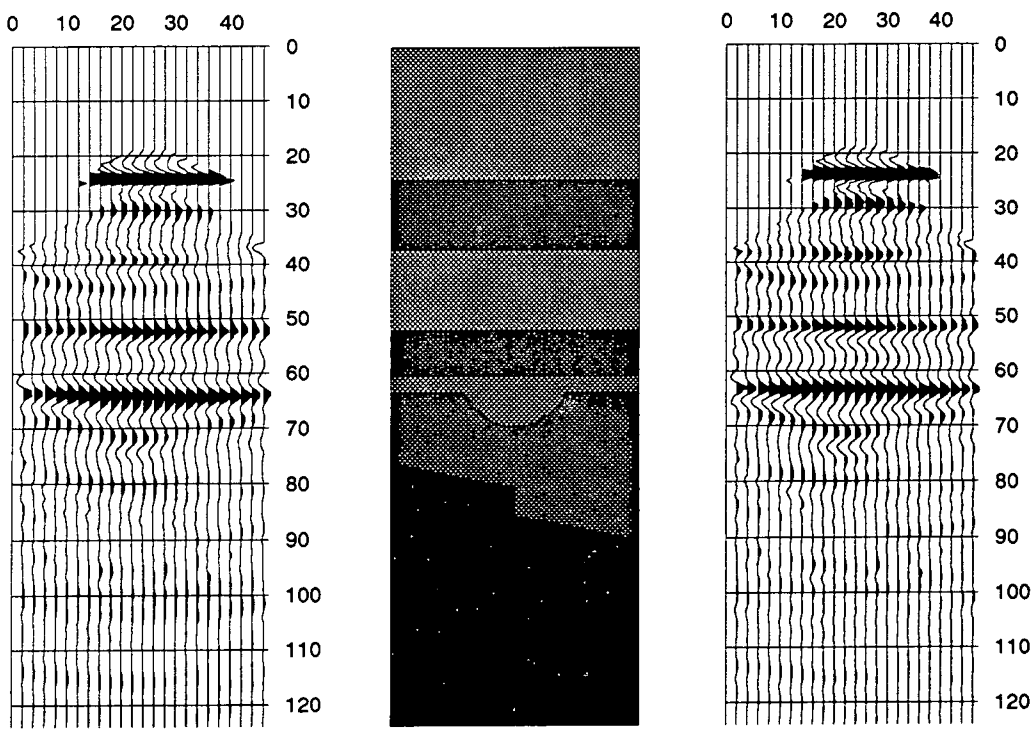

Figure 7.33 Upgoing migrated depth sections using (left) GB and (right) GK. Linear ramp with depth applied.

7.5.7 Aperture

The dipping faulted reflector has a dip of about 11°, the fault dips at 90° and the channel feature will have dips from 0-90°. Following the recommendations of

Findlay (1991), an aperture of 45° (±22.5° about the horizontal) was utilised. For comparison, an aperture of 4° (±2°) was also tested (§7.6.1).

7.6 The Migrated Results

7.6.1 Pre-flood

Reflected events in the pre-flood images of Figure 7.34 correlate well with the interfaces in the model (SEG reverse polarity, i.e. peak = compression. Linear ramp with depth applied); inadequate dip aperture appears to cause discrepancies over certain zones of the image. The bottom of the channel has been imaged by both up and downgoing wavefields, though its flanks are dipping too steeply to be sampled by the crosshole geometry. These images have been obtained by using a migration aperture which includes dips of ±22.5°, which experience shows will produce an optimally focused migrated image. When a narrow aperture of ±4° is used (almost in the limit of reflection point mapping), the image quality deteriorates (Figure 7.35). As expected, this affects the dipping interface and channel feature most, with the channel displaying some 'bow tie' characteristics on the upgoing image.

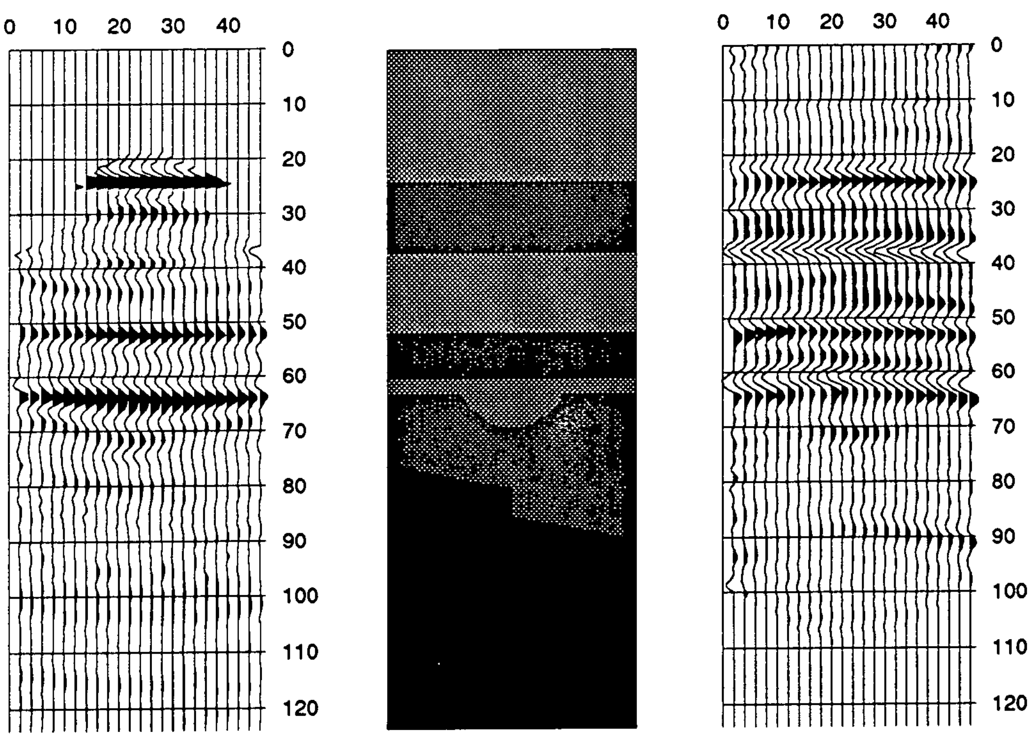

Figure 7.34 Pre-flood up and downgoing migrated depth sections. Aperture ±22.5°.

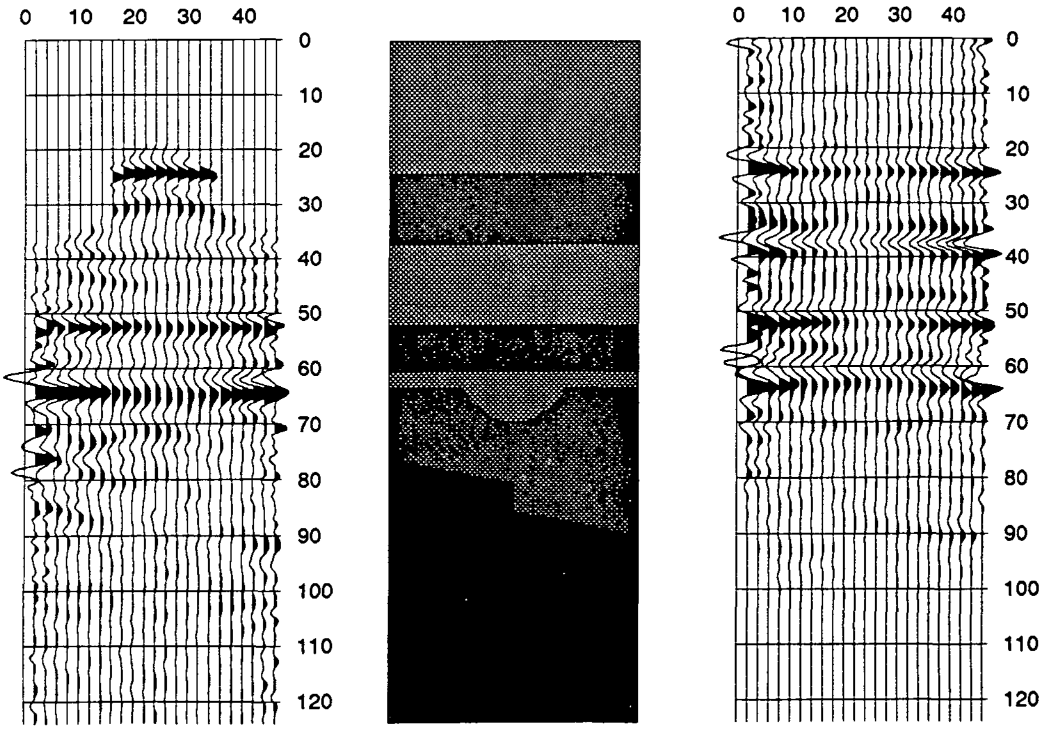

Figure 7.35 Pre-flood up and downgoing migrated depth sections. Aperture ±4°

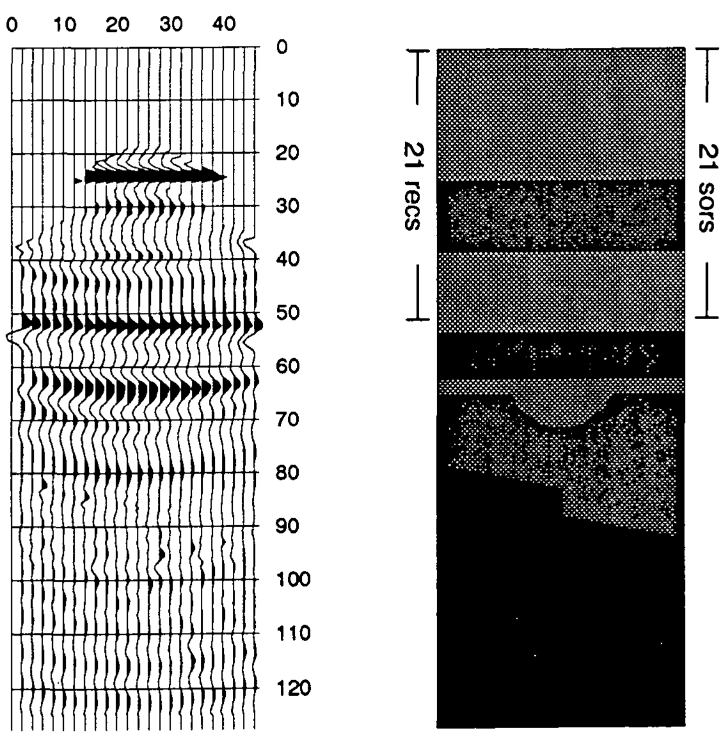

Figure 7.36 Pre-flood upgoing migrated depth section for 21 source and 21 receiver positions. Aperture ±22.5°. Linear ramp with depth applied.

In a production environment, it is likely that wells will bottom out in the reservoir layer. To model the corresponding crosshole experiment, only those 21 source and 21 receiver positions above the bottom of the reservoir layer have been used to produce the image shown in Figure 7.36. Of course it is necessary to repeat the 3-D *f-k-k* wavefield separation process for the 21 source and 21 receiver positions to prevent information from the lower traces contaminating the data. Again the migration aperture used is ±22.5°. The migrated image is still clear, although without the corresponding downgoing reflected image, it is difficult to interpret the channel feature. In comparison to the image using all 51 source and receiver positions (Figure 7.34), the lower reflectors are seen to be less well focused and to display smiling towards the side of the image. However, tomography in this case would only provide an image down to the level of the bottom source and receiver position.

7.6.2 Post-flood

To demonstrate the effects that the velocity model can have, the post-flood results are displayed in Figures 7.37 and 7.38 following raytracing through velocity models (ii) (two rectangular blocks) and (iv) (gradational) from §7.5.5, respectively. Velocity model (iii) (2 blocks with correct sloping boundary) is not shown as results are similar to model (ii). The images in Figure 7.37 show disruption around the sharp vertical boundary, where rays have been lost. One would expect the rays passing through the reservoir layer in model (iv) to be travelling too slowly on the left hand side of the model and too fast on the right hand side. The upper surface of a horizontal reflector illuminated solely by rays passing through the reservoir then would be imaged lower on the left and higher on the right than its physical position, and vice versa for the downgoing image. This effect is seen quite clearly on both the up and downgoing images in Figure 7.38. For this reason, velocity model (ii) is preferred, because, although major disruption has occurred, the depth section is nearer to reality.

The post-flood images (Figure 7.37 and 7.38) demonstrate the expected changes in reflectivity due to the flood zone. Of most note is the amplitude of the top and bottom of the flood zone on the up and downgoing images respectively. These plots, together with traveltime and amplitude tomography (Leggett et al. 1993) could be used to trace the progress of the flood front, using time-lapse repeated surveys.

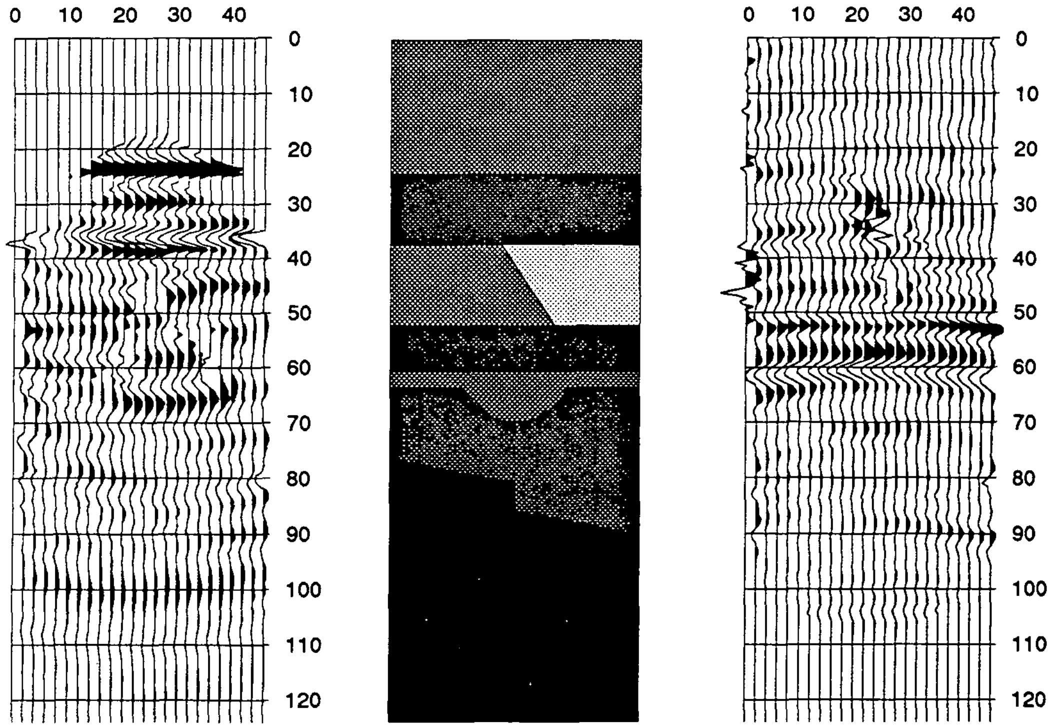

Figure 7.37 Post-flood up and downgoing migrated depth sections. Velocity model (ii). Aperture ±22.5°. Linear ramp with depth applied.

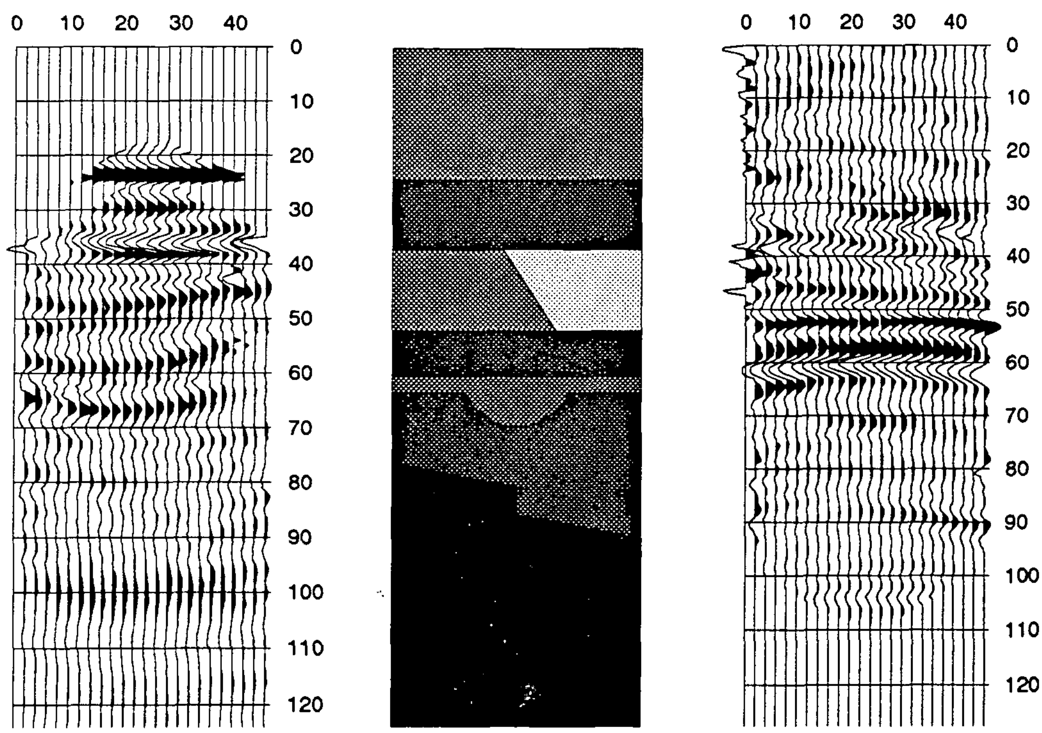

Figure 7.38 Post-flood up and downgoing migrated depth sections. Velocity model (iv). Aperture ±22.5°. Linear ramp with depth applied.

Chapter VIII

Coal Measures

The two surveys presented here, surveys 3438-3437 and 3500-3496 from the Lowther South site, have both been processed by Findlay (Findlay et al. 1991) and their acquisition is described in §2.3. They have been reprocessed in this study for different reasons.

- Survey 3438-3437 should have revealed a small fault close to borehole 3437, detected by an RVSP survey in the hole (Kragh et al. 1991), but Findlay's processing failed to image it. An investigation into why this was so, including some reprocessing, has been performed, and it is shown how survey geometry can limit imaging capability close to the boreholes and even in the middle of the section between the boreholes.

- Drillers' reports and wireline log information indicated that a normal fault zone passed between the two boreholes 3500 and 3496. Findlay's processing, including *f-k* filtering in CSGs, imaged the fault. The dataset has been reprocessed with 3D *f-k-k* filtering to compare the results of this advanced processing technique.

8.1 Survey 3438-3437 - Imaging capability of crosshole surveys

8.1.1 Background

The starting point for this study was a comparison between two different types of seismic reflection depth section generated along the same line of three boreholes (Figure 8.1a) at the Lowther South opencast coal exploration site in Yorkshire. These boreholes were 3436 (A), 3437 (B) and 3438 (C). The first section was produced by processing reverse vertical seismic profiles (RVSPs) shot in each borehole to give continuous coverage (Kragh et al. 1991). These RVSPs were shot using explosive charges downhole and a line of 24 geophones at the surface in the plane of the section. They showed two small faults cutting the coal seam at 50m depth, just to the right of boreholes B and C (Figure

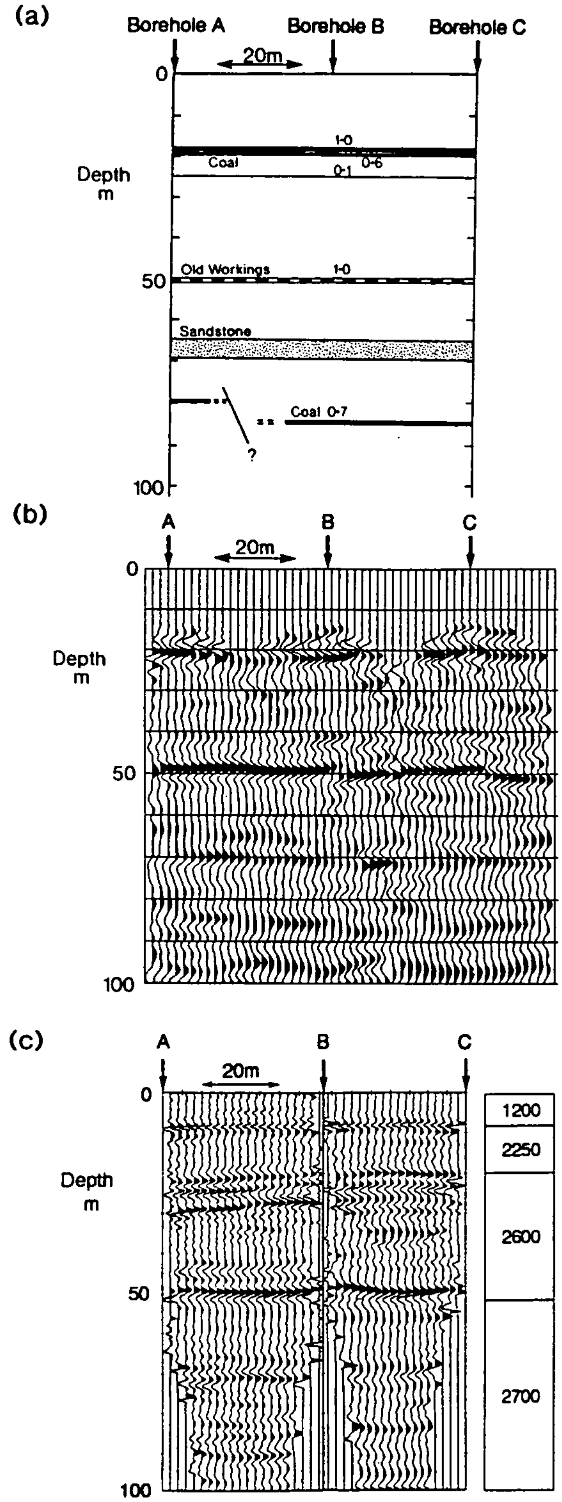

Figure 8.1 (a) Coal seam stratigraphy proved in boreholes A, B and C. (b) Hole-to-surface migrated depth section (from Kragh et al. 1991). (c) Crosshole migrated depth section with velocity field (m/s) used for migration (from Findlay et al. 1991). Sections plotted SEG normal polarity (peak=rarefaction).

8.1b). The second section was produced by processing crosshole seismic reflection surveys shot from the boreholes A and C into the central borehole B (Findlay et al. 1991). Single detonators were used as sources and hydrophones as receivers. The crosshole survey between boreholes C and B should have revealed the small fault just to the right of borehole B at 50 m depth, but it failed to do so (Figure 8.1c).

Before investigating why the crosshole processing had failed to image the fault, it was necessary to consider whether the fault imaged on the RVSP section from borehole B really exists, and whether it is accurately located. As described by Kragh et al. (1991), great care was taken in calculating static corrections for the surface geophones. No particularly anomalous values were found. After wavefield separation and deconvolution, each common source gather (CSG) was migrated separately, so that the resulting images could be examined before stacking. The fault appeared at the same location on each prestack image, which would not have been the case if the discontinuity in the reflector had been caused by inaccurate static corrections. Confidence in the integrity of the processing scheme has been enhanced by successful results of RVSP profiles from other lines of boreholes (Kragh et al. 1992). The possibility of significant error in lateral positioning of the RVSP image was also ruled out, since a verticality survey in borehole B showed a deviation of less than 0.5m at 50m depth in the plane of the survey.

Having concluded that the fault is real, why had the crosshole survey failed to image it? The spatial variation of image quality in crosshole reflection surveys has been explained in Chapter IV. Here the discussion of image quality concentrates on this particular crosshole reflection survey, and some general conclusions for all such surveys are drawn.

8.1.2 Initial processing

A typical CSG from the dataset is shown in Figure 8.2. The seismic signals had a bandwidth of 100-700Hz with a peak at about 200Hz. In the initial processing of these data to generate the section of Figure 8.1c, both uphole and crosshole direct-arrival traveltimes were used to make a first estimate of the velocity field. This was adjusted by a process of trial and error. The velocity field found by a tomographic inversion of the crosshole direct-wave traveltimes was unsatisfactory because of anisotropy.

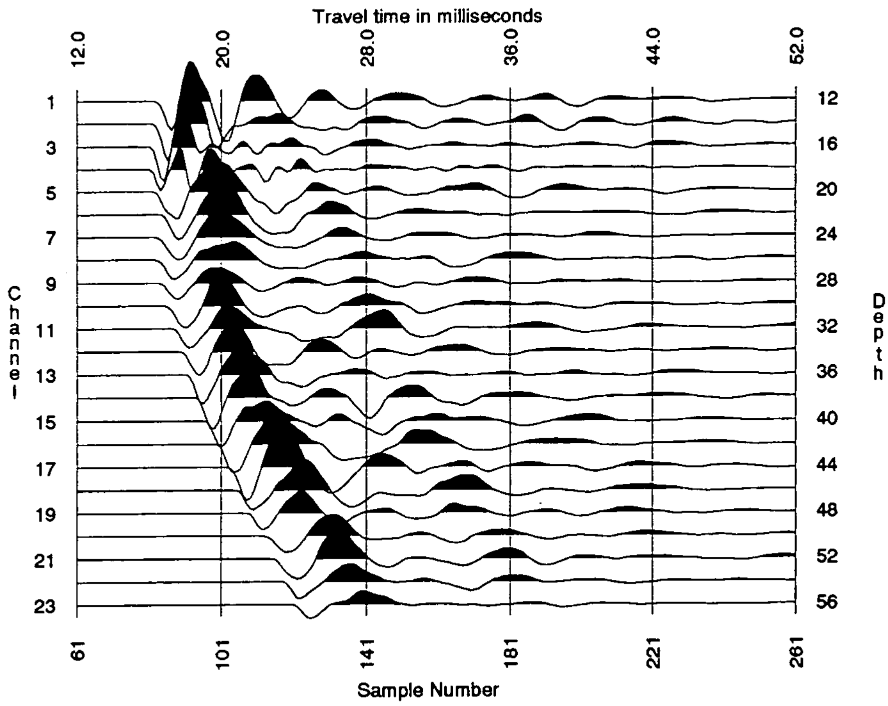

Figure 8.2 Raw data - the common source gather from 10m depth.

The direct arrivals were muted out and up- and downgoing wavefields were separated by filtering CSGs in the *f-k* domain. A wavelet was estimated from the data by averaging the autocorrelation functions of all 23 traces and making the minimum-phase assumption. Then a Wiener shaping filter was designed to shape the estimated wavelet into a zero-phase Butterworth wavelet of bandwidth 150-700Hz, and applied to the data. The GK migration scheme was used to migrate both up and downgoing wavefields, the polarity of the migrated downgoing wavefield was reversed, and the two migrated wavefields were merged together.

8.1.3 Reprocessing

A limited amount of reprocessing was undertaken to try to image the small fault, but did not result in significant improvement.

Muting was avoided in the reprocessing scheme because it will remove reflected events arriving shortly after the direct waves in real, band-limited datasets. The known location of the fault is close to the receiver borehole B (3437) and towards the bottom of the receiver array. The seismograms which contribute most to the image in this vicinity are those recorded by receivers just above the fault from the shallowest sources. The direct waves are downgoing at the receivers on these seismograms, whereas the primary reflections required for imaging are upgoing. Thus these wavefields could be separated in *f-k* space for each CSG (§3.6.3).

The GK migration method used in the initial processing is a far-field approximation, resulting from dropping the near-field term in the Kirchhoff integral, which might be important for imaging close to the boreholes. Accordingly, the GB migration scheme (§5.1) has been used in the reprocessing scheme. The difference between the results of using GK and GB migration is illustrated in Figure 8.3, where only the region around the small fault (expected location 50m depth, 34m offset), close to the receiver borehole, has been imaged.

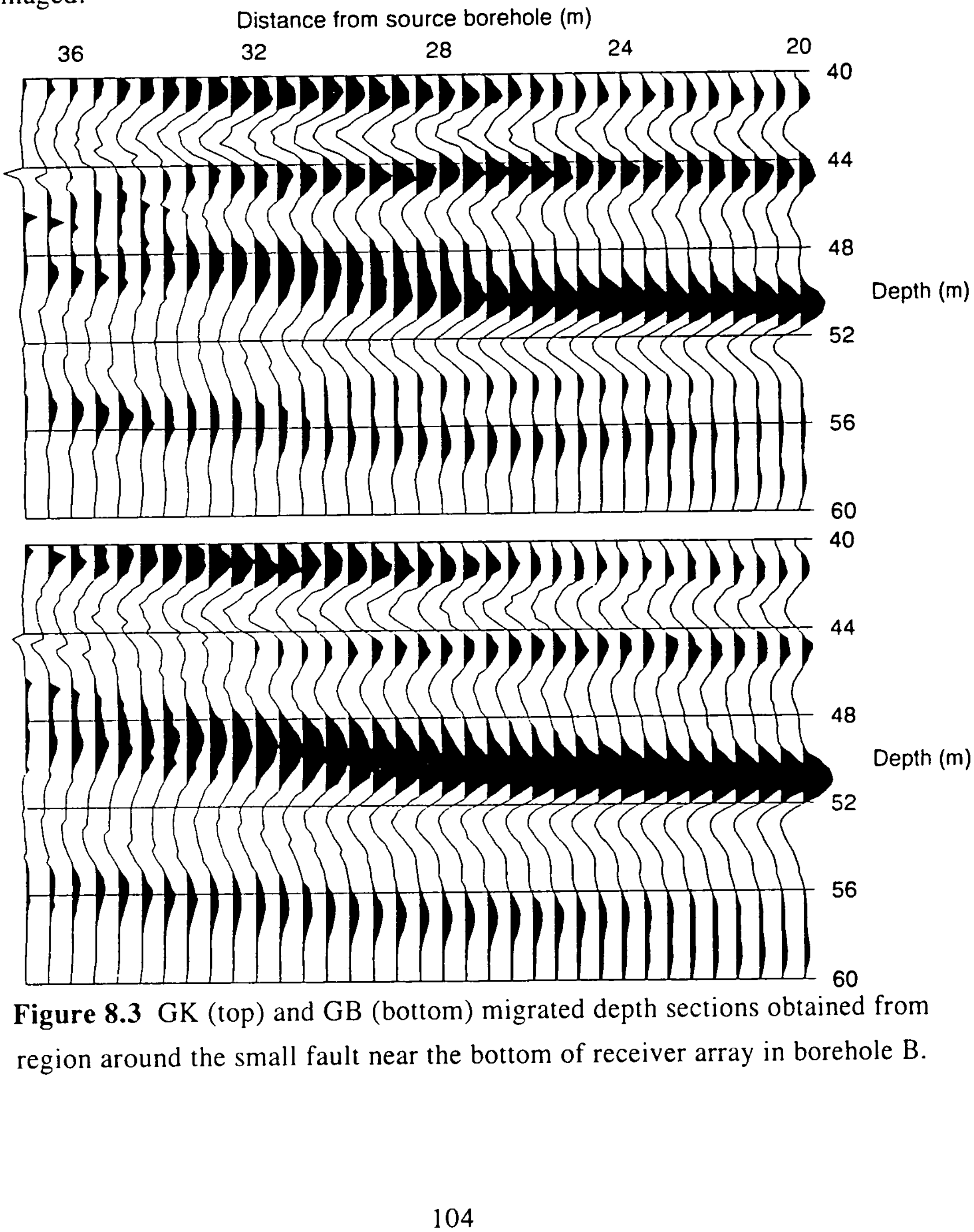

Figure 8.3 GK (top) and GB (bottom) migrated depth sections obtained from region around the small fault near the bottom of receiver array in borehole B.

There is better continuity of reflection character in the GB image, which is preferred as being in better accord with the stratigraphy, but it is still quite impossible to identify a fault.

8.1.4 Dip sampling at image points - shot and receiver spacing

At this point it was concluded that there was nothing further that could be done in processing to resolve the small fault. Instead the effects of the restricted lengths of source and receiver arrays and the element spacing in each array were considered (§4.2 and §4.3). By comparing the rose diagram in Figure 4.8 (reproduced here as Figure 8.4) with the depth migrated sections in Figure 8.3, it is plain to see that gaps exist within the range of dips sampled around the fault zone due to the coarse receiver spacing.

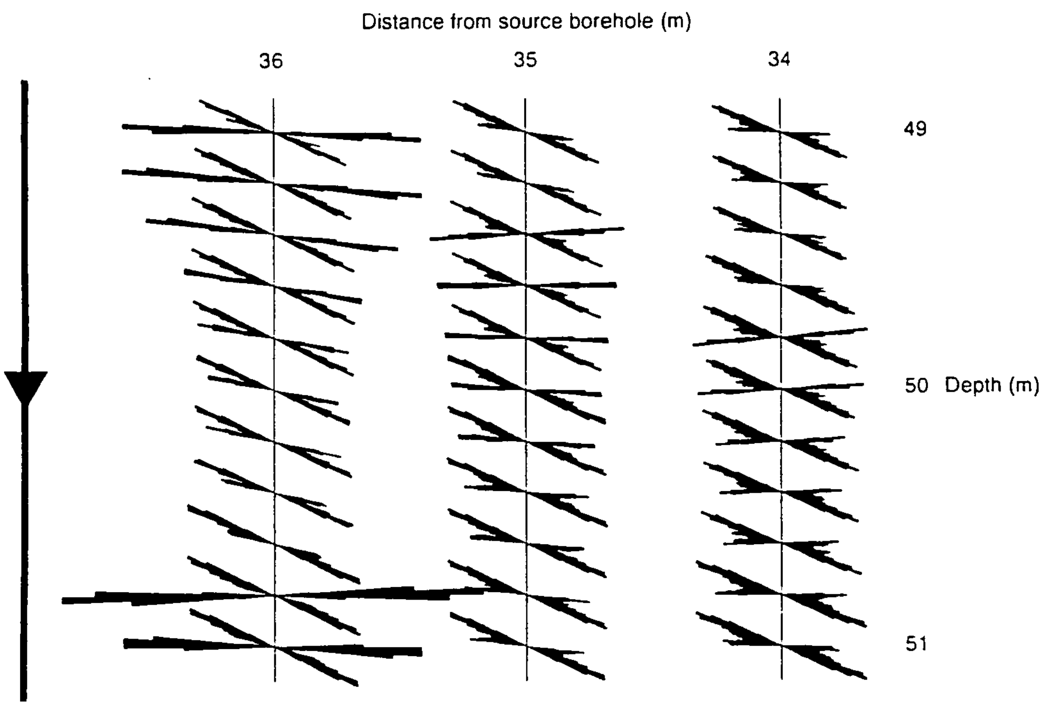

Figure 8.4 Rose diagram showing the effect of discrete spatial sampling on the distribution of dips sampled at image points spaced 0.2 m apart vertically over a 2 m interval at distances of 1-3 m from the receiver borehole. (This Fig. is identical to Fig. 4.8)

For image points close to either the source or receiver array, it is the combination of spatial sampling in the near borehole (§4.3) and the extent of the array in the far borehole (§4.2) which govern the ability to form an accurate image. Rose diagrams such as in Figure 8.4 clearly show where spatial sampling and array lengths are adequate over the section. Where they are inadequate, the image will inevitably be smeared, whatever is done in processing. It is believed that this is the cause of the failure to image the small fault close to borehole B in the crosshole survey between boreholes C and B. In order to image this small fault, which should appear as a step in reflecting horizons, dips should have been sampled smoothly around the horizontal on both sides of the fault.

8.2 Survey 3500-3496 - *f-k* versus *f-k-k*

As a demonstration of the *f-k-k* technique (§5.2), survey 3500-3496 has been reprocessed using two processing routes, identical except for performing wavefield separation by *f-k-k* rather than *f-k* filtering.

These data were previously presented by Findlay et al. (1991) with wavefield separation carried out by *f-k* filtering of common shot gathers. The boreholes used in this survey were 32m apart. Drillers' reports and wireline log information indicated that a normal fault zone passed between the two boreholes with a vertical throw of some 22m. Shots were fired at 2m intervals from 10m to 52m depth in one borehole, and receivers were deployed at 2m spacing from 9.79m to 53.79m depth in the other (Figure 2.4).

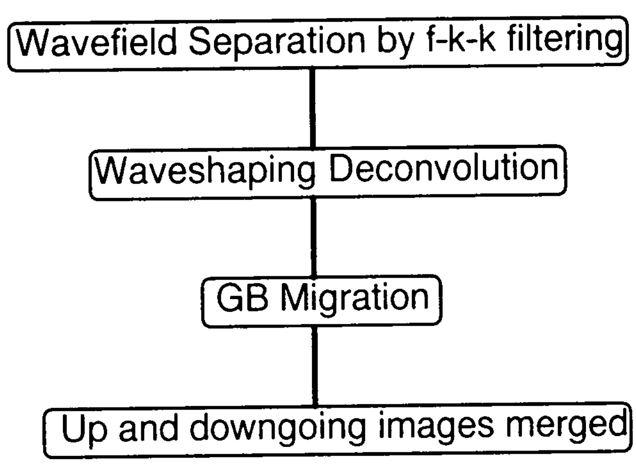

Figure 8.5 The crosshole reflection processing sequence used to produce Figure 8.7b. Compare this with Figure 3.1.

In reprocessing these data, wavefield separation was performed by 3-D *f-k-k* filtering (§5.2). Figure 8.6 shows the data volume depicted before and after 3-D *f-k-k* filtering as *f*-slice contour plots in the *f-k-k* domain. Following wavefield separation, deconvolution and Kirchhoff migration using the Generalised Berryhill algorithm (§5.1) were performed (Figure 8.5). Up and downgoing wavefields were migrated separately, the polarity of the downgoing reflections reversed, and the images merged (Figure 8.7b). Results obtained by a processing scheme involving 2-D *f-k* filtering in common shot gathers, which differed only in the wavefield separation step, are shown as Figure 8.7a for comparison. An AGC over 30m has been applied to both images.

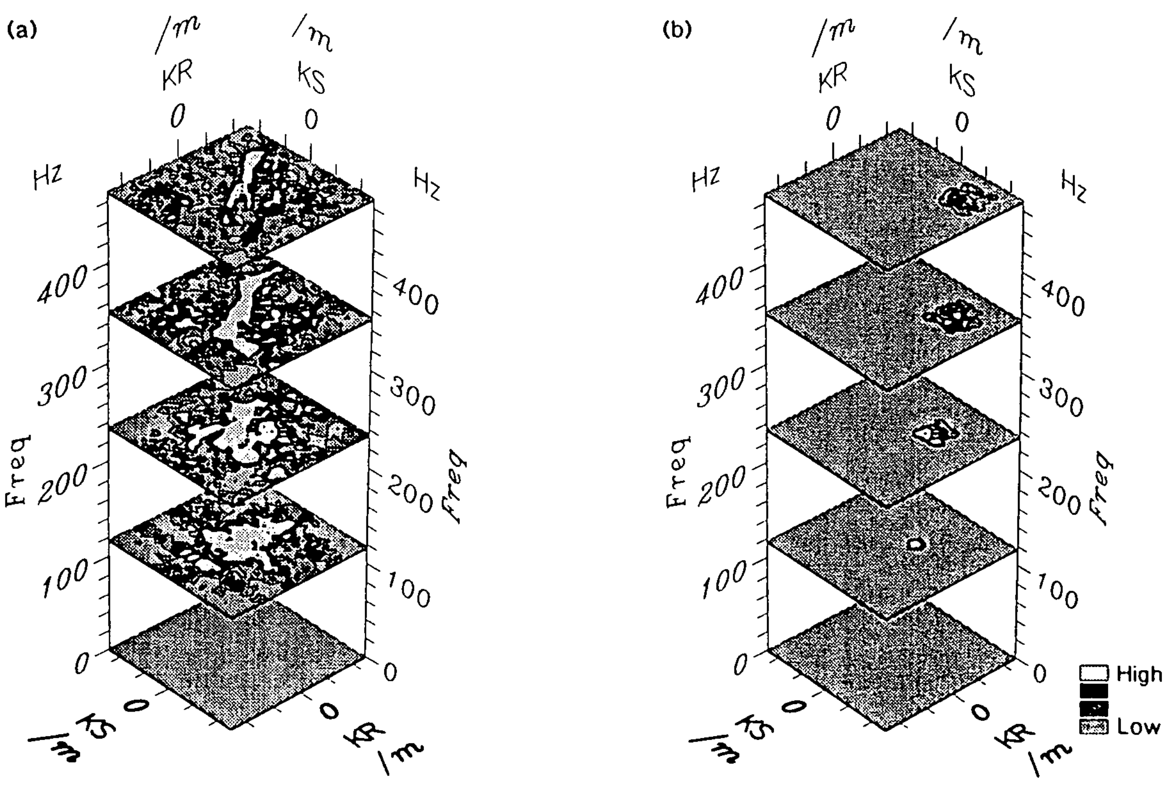

Figure 8.6 The data volume transformed into the *f-k-k* domain, and depicted as *f*-slice plots with a linear amplitude scale: (a) before filtering, and (b) after 3-D *f-k-k* filtering.

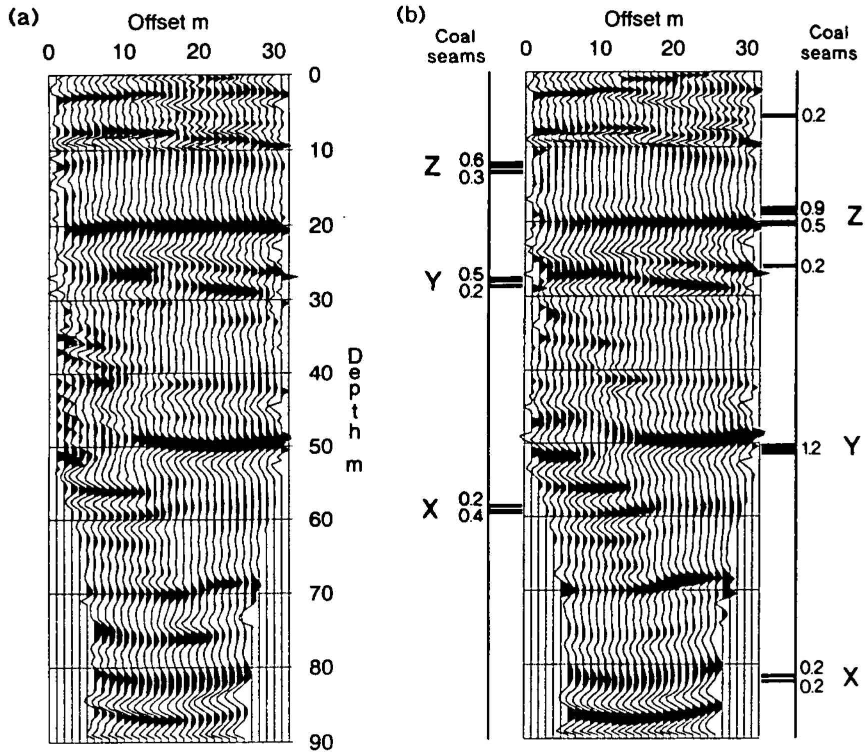

Figure 8.7 Migrated depth sections following (a) 2-D *f-k* wavefield separation, and (b) 3-D *f-k-k* wavefield separation.

Both processing schemes have clearly imaged the fault. The reflection from coal seam Z is continuous across the section from the borehole on the right, so there must be a fault with 7m throw at this horizon very close to the left borehole. A larger fault, of some 15m vertical throw, cuts the left borehole between seams Z and Y. The truncations of reflections, from seam Y to the right and from seam X to the left, locate the fault zone in the body of the data. The most striking difference between Figures 8.7a and 8.7b is the nature of the migrated image on the left side of the section. The continuity of reflectors is poorer in the section with 2-D wavefield separation, especially near the source borehole, and the level of coherent noise is higher. With the improved image produced by the 3-D *f-k-k* processing a more confident interpretation of the location of the fault can be made.

Chapter IX

Conclusions and suggestions for future work

9.1 Conclusions

Crosshole seismic reflection imaging has been shown to be a high resolution imaging technique. Major limitations of the method have been discussed, and advice given on how the produced depth sections should be interpreted. New processing techniques have been developed and implemented to overcome specific problems with previous processing schemes, and their success has been clearly demonstrated using example datasets. Finally, three types of dataset have been processed and the results discussed in terms of the limitations of the method, of the processing techniques developed, of the acquisition geometry used, and of the quality of image.

9.1.1 Imaging capability

Effective imaging in any seismic survey is restricted to that part of the subsurface section where the image is not smeared. Image quality depends on the range of dips sampled around the local structural dip and on the distribution of dip angles within that range. For image points close to either array, the distribution of dips sampled within the range can contain gaps if the element spacing in that array is too coarse. The overall range of dips sampled depends on the lengths of source and receiver arrays, which should be comparable with the borehole separation in order to image a horizon at the base of the arrays. It follows that in order to image a horizon level with the centres of the source and receiver arrays, the array lengths need to be twice the borehole separation.

As regards improving the imaging capability of crosshole reflection surveys, the Generalised Berryhill algorithm and the use of 3-D *f-k-k* filtering are recommended. Close to the boreholes, provided that the spatial sampling interval is small enough to give a smooth distribution of dips, it is believed that GB migration will have better imaging capability than GRT or GK migration because it includes the near-field term in the Kirchhoff integral. Coherency problems in CRGs (or CSGs) have been shown to be produced by performing wavefield separation on CSGs (or CRGs) by conventional 2-D *f-k* filtering. This migrates to coherent noise on the source (or receiver) array side of the depth section. Addressing this problem, *f-k-k* filtering has been shown to be the most satisfactory solution, such that both sides of the migrated section have equal image quality and fidelity.

9.1.2 Results

It has been shown that a restricted survey involving a single common shot gather from the Groningen experiment can generate an image between wells 300m apart with a signal bandwidth of over 1kHz. The potential for high resolution imaging of a more extensive survey from a producing field is evident.

We have shown an innovative processing scheme for crosshole seismic reflection imaging. The use of 3-D *f-k-k* filtering has made the muting of direct arrivals in the time domain superfluous, since the separation of direct and reflected wavefields and of up and downgoing reflections is achieved in one operation. The migrated images produced are of high resolution and can be used to monitor the progress of the flood front during EOR.

Conventional 2-D *f-k* filtering for wavefield separation of common shot gathers in crosshole reflection data has been problematical, primarily because of the direct and reflected wavefields overlying each other in the *f-k* domain, and because of poor coherency of the separated wavefields when viewed in common receiver gathers. Coherent noise has resulted, and we have demonstrated with real data that this may be reduced through use of a one-pass 3-D *f-k-k* filter for wavefield separation. Such filters can reject both compressional and shear direct waves and also the strongest multiple arrivals in crosshole data, but not head waves.

9.2 Future work

9.2.1 Further comparison of standard and novel processing techniques

Further research is required into the applicability of the processing techniques presented in this thesis, and this is only possible with the acquisition of more good datasets, either from the Coal Measures or from producing fields. It would thereafter be possible to provide more comprehensive guidelines on the conditions governing the suitability of each scheme.

9.2.2 Quantification of the quality of images

Thus far it has not been possible to provide a measure of the quality of an image, and therefore the success of a processing scheme, beyond a simple statement of how well the migrated image qualitatively matches the assumed geological cross section, or the known model geometry in the case of the tank data. Work is needed then in providing a technique for quantification of images, possibly by some sort of correlation philosophy, in order to state categorically whether one image is a better representation of the section than another.

9.2.3 Detailed amplitude interpretation of crosshole reflection images

Following on from the point above (§9.2.2), special attention should be given to the preservation of amplitude values during recording and processing of crosshole data in order to maintain amplitude information in the migrated sections. Non-linear practices such as normalising the data and performing AGC should be avoided wherever possible. Future work could then involve the study of reflector strength for lithology analysis (§9.2.10).

9.2.4 Resolution of raytracing problem

One problem still requiring attention is the limitation of the boxel raytracing, especially when sloping boundaries are required. The proposed solution is a raytracing program based on triangular velocity cells, as these would be able to cope with any velocity polygon.

9.2.5 Integration of amplitude tomography into raytracing

Leggett et al. (1993) demonstrated the possibilities of amplitude tomography. It would be possible to integrate the results of an amplitude tomogram into the raytracing for crosshole reflection imaging, by calculating the total attenuation along each reflected raypath. This would then give an amplitude correction factor for each source-receiver-image point combination to compensate for the attenuation through the media.

9.2.6 Further comparison of GRT and GB migration

Recent publications have described the extension of the scalar inversion problem of reconstructing a velocity perturbation in a constant density acoustic medium to a solution of the vector inversion problem of material parameters. Beylkin and Burridge (1990) have developed an algorithm for multi-parameter inversion of surface seismic reflection data based on the inverse Generalized Radon Transform. Miller and Burridge (1992) have recast this algorithm in terms of a

GRT-based dip-moveout operator. The removal of the constant density restriction of GRT imaging is of interest to this work, and some thought should be given to the implications of the above two publications to the work presented in this thesis, possibly by comparing the results of the physical model dataset using GB, GRT and multi-parameter GRT imaging.

9.2.7 Novel acquisition geometries

Proposed future experiments include acquiring data with shots and receivers in the same borehole, either vertical or with a 45° trajectory. Major consideration will have to be given to acquisition problems such as the damping of tube waves. However, it is hoped that this technique will be attractive for hydrocarbon reservoir definition with inclined or horizontal boreholes.

9.2.8 One-pass total processing

Possibly mere speculation, though on purely aesthetic grounds, a one pass total processing philosophy for crosshole seismic reflection processing could be a final goal. Since conceiving the idea of 3-D wavefield separation in the f-k_s-k_r domain, other processing steps have presented themselves as being suitable for application in f-k_s-k_r, such as directional deconvolution with respect to source and receiver and trace interpolation. What could be considered then is:-

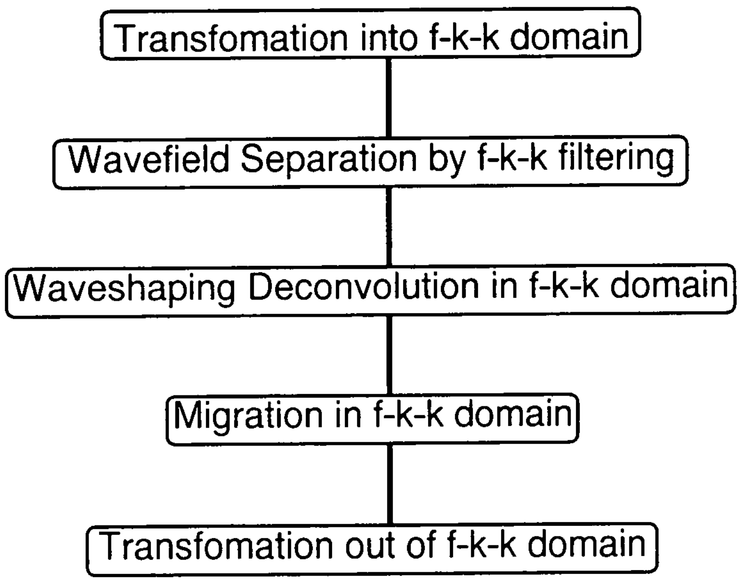

Figure 9.1 Speculative one-pass crosshole reflection processing sequence.

9.2.9 Imaging of all modes

Previous workers have only mapped P-P (e.g. this study) and/or S-S modes (Lazaratos et al. 1992, Becquey et al. 1992), though four modes are always available. The first arrival P-P reflections are the obvious first choice for imaging, since they are uncontaminated by other reflection arrivals. What has not been tried on real data so far (although Balch et al. (1991) have migrated different wave modes in laboratory model data) is the development of an imaging scheme for integrating the information from up and downgoing images for both P-P and S-S reflected wavefields, as well as mode-converted arrivals such as P-S and S-P. For this, a dataset of the highest quality would be required. Once cross-sectional images had been produced, the initial approach could be that of inverting each image (P-P, S-S, P-S, S-P) separately. Note that once P-P and S-S images had been produced, it would be a simple matter to construct the images for mode-converted P-S or S-P waves, as the raypaths to the image points would have already been computed. The images produced would be compared and updated accordingly in an iterative process. The four reflectivity images could then be inverted together to obtain P and S velocity images, provided some assumption is made about density.

By this scheme, quantitative information (lithology derived from reflection amplitude) would be extracted to complement the previous qualitative crosshole images of bed geometry. Once a target reflector (thin bed, faulted interface, reservoir rock) had been identified, the nature and continuity of reflectivity along it could be traced to assess small-scale changes in rock character. The success of these schemes could be rigorously tested by comparing the results with known properties derived from well logs in the field.

Since the image amplitudes are critical to the success of the inversion process, considerable attention will be given to the validity of the image amplitudes in each imaging technique used, to the dependence of amplitude on angular coverage of the reflecting interface and to any possible source directivity.

9.2.10 Integration of high-resolution seismic data into inversion schemes for deriving reservoir properties.

Seismic data have been of insufficient resolution to delineate porosity and permeability heterogeneities on the scale required by reservoir engineers for modelling fluid flow. However, the recent improvements in 3D seismic data quality and the development of crosshole seismic and VSP methods suggest an investigation of the detail that can be obtained by integrating this higher resolution seismic data in an inversion scheme with wireline log and core data from wells. The crosshole and VSP data act as a node of reference between the two extremes of the areal extent of the 3-D seismic and the vertical resolution of the core and wireline data. The inversion would start with known properties at one well, and proceed through the seismic data volume to an adjacent well. The success of an inversion scheme would be rigorously tested by comparing the seismic inversion results with the known reservoir properties in the second well.

References

Albright, J.N. and Johnson, P.A. 1990. The use of vertical seismic profiles in seismic investigations of the earth. *Geophysical Prospecting* **38**, 607-620.

Balch, A.H., Chang, H., Hofland, G.S., Ranzinger, K.A., and Erdemir, C. 1991. The use of vertical seismic profiles in seismic investigations of the earth. *Geophysical Prospecting* **39**, 887-913.

Balogh, W.T., Owen, T.E., and Harris, J.M. 1988. New piezoelectric transducer for hole-to-hole seismic applications. 58th Meeting, SEG, Expanded Abstracts, 155-157.

Baria, R., Jackson, P.D., and McCann, D.M. 1989. Further development of a high-frequency seismic source for use in boreholes. *Geophysical Prospecting* **37**, 31-52.

Beattie, K.E. 1990. A comparison of receiver types in crosshole seismics. Unpublished MSc dissertation, University of Durham.

Becquey, M., Bernet-Rollande, J.O., Laurent, J., and Noual, G. 1992. Imaging reservoirs - a crosswell seismic experiment. *First Break* **10**(9), 337-344.

Berryhill, J.R. 1979. Wave-equation datuming. *Geophysics* **44**, 1329-1344.

Beydoun, W.B., Delvaux, J., Mendes, M., Noual, G., and Tarantola, A. 1989. Practical aspects of an elastic migration/inversion of crosshole data for reservoir characterization: A Paris basin example. *Geophysics* **54**, 1587-1595.

Beylkin, G. and Burridge, R. 1990. Linearized inverse scattering problem of acoustics and elasticity. *Wave motion* **12**, 15-52.

Cai, W. and Schuster, G.T. 1993. Processing crosswell seismic data for reflection imaging. 55th Meeting, EAEG, Stavanger, Expanded Abstracts, C006.

Carrion, P.M., Sato, H.K. and Buono, A.V.D. 1991. Wavefront sets analysis of limited aperture migration sections. *Geophysics* **56**, 778-784.

Cassel, B.R. 1982. Seismograms in laterally varying media. *Geophysical J. R. astr. Soc.* **69**, 339-354.

Cassel, B.R. 1984. Vertical seismic profiles - an introduction. *First Break* **2**(11), 9-19.

Chen, S.T., Zimmerman, L.J., and Tugnait, J.K. 1990. Subsurface imaging using vertical seismic profiling abd crosshole tomographic methods. *Geophysics* **55**, 1478-1487.

Chen, S.T., Eriksen, E.A., and Miller, M.A. 1990. Experimental studies on downhole seismic sources. *Geophysics* **55**, 1645-1651.

Christie, P.A.F, Hughes, V.J., and Kennett, B.L.N. 1983. Velocity filtering of seismic reflection data. *First Break* **1**(3), 9-24.

Claerbout, J.F. 1970. Coarse grid calculations of waves in inhomogeneous media with application to delineation of complicated seismic structure. *Geophysics* **35**, 407-418.

Claerbout, J.F. 1971. Toward a unified theory of reflector mapping. *Geophysics* **36**, 467-481.

Claerbout, J.F. and Johnson, A.G. 1971. Extrapolation of time dependent waveforms along their path of propagation. *Geophysical Journal of the Royal Astronomical Society* **26**, 285-293.

Cooley, J.W. and Tukey, J.W. 1965. An algorithm for the machine calculation of complex Fourier series. *Maths computations* **19**, 297-301.

Deregowski, S. and Rocca, F. 1981. Geometrical optics and wave theory for constant-offset sections in layered media. *Geophysical Prospecting* **29**, 374-387.

Devey, M.G. 1978. Derivation of the migration integral. Technical Note TN451, BP Company Ltd, Exploration and Production Department.

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

Dillon, P.B. 1988. Vertical seismic profile migration using the Kirchhoff integral. *Geophysics* **53**, 786-789.

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

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

Emeleus, K. 1993. A comparison of receiver types in crosshole seismic investigations. Unpublished MSc dissertation, University of Durham.

Findlay, M.J. 1991. Cross-hole seismic reflection surveying in Coal Measures. Unpublished Ph.D. dissertation, University of Durham.

Findlay, M.J., Goulty, N.R., and Kragh, J.E. 1991. The crosshole seismic reflection method in opencast coal exploration. *First Break* **9**(11), 509-514.

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

Gazdag, J. 1978. Wave equation migration with the phase-shift method. *Geophysics* **43**, 1342-1351.

Geyer, R.L. 1993. Spindletop cross-borehole survey. *The Leading Edge* **12**(1), 26-31.

Goodman, J.W. 1968. An introduction to Fourier optics. McGraw-Hill, San Francisco.

Goulty, N.R. 1993. Controlled-source tomography for mining and engineering applications. **in** *Seismic Tomography*. H.M. Iyer and K. Hirahira (eds), Vol. 1, 797-813. Chapman & Hall, London.

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

Hagedoorn, J.G. 1954. A process of seismic reflection interpretation. *Geophysical Prospecting* **2**, 85-127.

Hale, D. and Claerbout, J.F. 1983. Butterworth dip filters. *Geophysics* **48**, 1033-1038.

Hardage B.A. 1985. Vertical seismic profiling. Part A: Principles. 2nd edition, Pergammon Press.

Hardage, B.A. 1992. Crosswell seismology and reverse VSP. (Seismic Applications Series, Vol. 1). Geophysical Press, London.

Harris, J.M., Nolen-Hoeksema, R., Rector, J.W., Lazaratos, S.K., and Van Shaack, M. 1992. High resolution cross-well imaging of a West Texas carbonate reservoir: Part 1. Data acquisition and project overview. 62nd Meeting SEG, Expanded Abstracts, 35-39.

Hatton, L., Worthington, M.H., and Makin, J. 1988. Seismic Data Processing. Blackwell Scientific Publications Ltd, Oxford.

Hood, P. 1981. Migration. **in** *Developments in geophysical exploration methods*. A.A. Fitch (ed), Vol. 2, 151-230. Applied Science Publishers Ltd., London.

Hu, L.-Z. and McMechan, G.A. 1987. Wave-field transformations of vertical seismic profiles. *Geophysics* **52**, 307-321.

Hubbard, T.P., Sugrue, M.J., Sandham, W.A., and Booth, E.A. 1984. Marine source and receiver deghosting and array inversion in F-K space. 46th EAEG Meeting, Abstracts volume, p26.

Jackson, P.J., Onions, K.R., and Westerman, A.J. 1989. Use of inverted VSP to enhance the exploration value of boreholes. *First Break* **7**(6), 223-246.

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

Justice, J.H., Mathisen, M.E., Vassilou, A.A., Shiao, B.R., Alameddine, B.R., and Guinzy, N.J. 1993. Crosswell seismic tomography in improved oil recovery. *First Break* **11**(6), 229-239.

Kragh, J.E. 1990. Borehole seismic methods for opencast coal exploration. Unpublished PhD dissertation, University of Durham.

Kragh, J.E., Goulty, N.R., and Findlay, M.J. 1991. Hole-to-surface seismic reflection surveys for shallow coal exploration. *First Break* **9**(7), 335-344.

Kragh, J.E., Goulty, N.R., and Brabham, P.J. 1992. Surface and hole-to-surface seismic reflection profiles in shallow Coal Measures. *Quarterly Journal of Engineering Geology* **25**, 217-266.

Krajewski, C., Dresen, L., and Gelbke, C. 1989. Iterative tomographic methods. *Geophysical Prospecting* **37**, 717-751.

Lazaratos, S.K., Rector, J.W., Harris, J.M., and Van Shaack, M. 1992. High resolution cross-well imaging of a West Texas carbonate reservoir: Part 4. Reflection imaging. 62nd Meeting SEG, Expanded Abstracts, 49-53.

Leggett, M. 1992. Crosshole seismic processing of physical model and Coal Measures data. Unpublished PhD dissertation, University of Durham.

Leggett, M., Goulty, N.R., and Kragh, J.E. 1993. Study of traveltime and amplitude time-lapse tomography using physical model data. *Geophysical Prospecting* **41**, 599-619.

Li, Q. and Worthington, M.H. 1990. Crosshole GRT imaging: an experiment with ultrasonic model data. 60th Meeting, SEG, Expanded Abstracts, 156-159.

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

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

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

Miller, D. and Burridge, R. 1992. Multiparameter inversion, dip-moveout, and the generalized radon transform. **in** *Geophysical Inversion*. J.B. Bednar, L.R. Lines, R.H. Stolt, and A.B. Weglein (eds), 46-58. Society for Industrial and Applied Mathematics, Philadelphia.

Newman, P. 1990. Amplitude and phase properties of a digital migration process. *First Break* **8**(11), 397-403.

Özdemir, H. and Saatçilar, R. 1990. Efficient multichannel filtering of seismic data. *Geophysical Prospecting* **38**, 1-22.

Peardon, L.G. and Bacon, C.W.M. 1992. An introduction to FKK techniques. *First Break* **10**(4), 113-123.

Pratt, R.G. and Worthington, M.H. 1990. Acoustic wave equation inverse theory applied to multi-source crosshole tomography, Part I: Acoustic wave-equation method. *Geophysical Prospecting* **38**, 287-310.

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

Pratt, R.G., Li, Q., Dyer, B.C., Goulty, N.R., and Worthington, M.H. 1991. Algorithms for EOR imaging: an experiment with scale model data. *Geoexploration* **28**, 193-220.

Qin, F. and Schuster, G.T. 1993. Crosswell reflection migration by a constrained Kirchhoff integral method. 55th Meeting, EAEG, Stavanger, Expanded Abstracts, C008.

Rector, J.W. and Marion, B.P. 1991. Use of drill bit energy as a downhole seismic source. *Geophysics* **56**, 628-634.

Rector, J.W., Lazaratos, S.K., Harris, J.M., and Van Shaack, M. 1992a. High resolution cross-well imaging of a West Texas carbonate reservoir: Part 3. Wavefield separation: 62nd Meeting SEG, Expanded Abstracts, 45-48.

Rector, J.W., Lazaratos, S.K., Harris, J.M., and Van Shaack, M. 1992b. Extraction of reflections from cross-well wavefields: 62nd Meeting SEG, Expanded Abstracts, 54-57.

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

Rowbotham, P.S. and Goulty, N.R. 1993a. (In press). Imaging capability of crosshole seismic reflection surveys. *Geophysical Prospecting*.

Rowbotham, P.S. and Goulty, N.R. 1993b. (submitted). Wavefield separation by 3D filtering in crosshole seismic reflection processing. *Geophysics*.

Sams, M.S., Worthington, M.H., King, M.S., and Shams Khanshir, M. 1993. A comparison of laboratory and field measurements of P-wave anisotropy. *Geophysical Prospecting* **41**, 189-206.

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

Sharp, R.J., Peacock, J.H., and Goulty, N.R. 1985. Ultrasonic seismic modelling system. 47th EAEG Meeting, Abstracts volume, p37.

Sheriff, R.E and Geldart, L.P. 1983. Exploration Seismology, Vol. 1. Cambridge University Press.

Smith, B.A., Gallagher, J.G., Hoover, G.M., and Hufford, J.M. 1993. Characteristics of crosswell wave propagation at varying well offsets. 55th Meeting, EAEG, Stavanger, Expanded Abstracts, C007.

Stewart, R.R. 1989. 3-D F-K filtering. 59th Meeting SEG, Expanded Abstracts, 1123-1124.

Stewart, R.R. and Marchisio, G. 1991. Cross-well seismic imaging using reflections. 61st Meeting SEG, Expanded Abstracts, 375-378.

Stolt, R.H. 1978. Migration by Fourier transform. *Geophysics* **43**, 23-48.

Sun, R. and McMechan, G.A. 1986. Pre-stack reverse-time migration for eastic waves with application to synthetic offset vertical seismic profiles. *Proc., Inst. Electr. Electron. Eng.* **74**, 457-465.

Tooley, R.D., Spencer, T.W., and Sagoci, H.F. 1965. Reflection and transmission of plane compressional waves. *Geophysics* **30**, 552-570.

Vaage, S. and Ziolkowski, A. 1992. Cross-Hole Seismic Survey in the Groningen Gas Field. 54th EAEG Meeting, Paris, Expanded Abstracts, 578-579.

Van Shaack, M., Harris, J.M., Rector, J.W., and Lazaratos, S.K. 1992. High resolution cross-well imaging of a West Texas carbonate reservoir: Part 2. Wavefield modeling and imaging: 62nd Mtg SEG, Expanded Abstracts, 40-44.

Williamson, P.R., Sams, M.S., and Worthington, M.H.. 1993. Crosshole imaging in anisotropic media. *The Leading Edge* **12**(1), 19-23.

Winbow, G.A. 1991. Seismic sources in open and cased boreholes. *Geophysics* **56**, 1040-1050.

Wong, J., Hurley, P., and West, G.F. 1983. Cross-hole seismology and seismic imaging in crystalline rock. *Geophysics Research Letters* **10**, 686-689.

Wyatt, K.D. and Wyatt, S.B., 1984. Determining subsurface structure using the vertical seismic profiling. **In** *Vertical Seismic Profiling: Advanced Concepts.* M.N. Toksoz and R.R. Stewart (eds), Geophysical Press.

Yilmaz, O. 1987 Seismic Data Processing. Series: Investigations in Geophysics, Volume 2. Society of Exploration Geophysicists, Tulsa.

Zhong, L. and Worthington, M.H. 1992. Crosswell continuity logging using Stoneley waves. *Journal of Seismic Exploration* **1**, 293-308.

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

Appendix A

Deviations of the Groningen boreholes

The deviation reports of the Scheemderzwaag boreholes are in the form of *x,y,z* deviations from the wellhead positions for the depths of interest. A 2-D deviation was assumed for the crosshole survey used in this study as *x* and *y* of the receiver well varied approximately linearly over the depth range 2225m-2600m.

RECEIVER WELL

Wellhead position (m):

x	y	z
1553:60	133.12	0.0

Deviations (m):

x	y	z
1.39	-13.52	1550.0
0.90	-15.62	1575.0
0.06	-19.93	1625.0
-0.35	-22,12	1650.0
-0.76	-24,33	1675.0
-1.15	-26.57	1700.0
-1.56	-28.89	1725.0
-2.02	-31.36	1750.0
-2.53	-33.96	1775.0
-3.07	-36.64	1800.0
-3.65	-39.37	1825.0
-4.47	-42.23	1850.0
-5.70	-45.35	1875.0
-7.29	-48.80	1900.0
-9.26	-52.59	1925.0
-11.40	-56.59	1950.0
-13.55	-60.65	1975.0
-15.80	-64.76	2000.0
-18.04	-68.94	2025.0
-20.14	-73.20	2050.0
-22.21	-77.56	2075.0
-24.38	-81.92	2100.0
-26.63	-86.22	2125.0
-28.91	-90.51	2150.0
-31.35	-94.80	2200.0
-33.72	-99.01	2200.0
-36.06	-103.14	2225.0
-38.59	-107.26	2250.0
-41.40	-111.46	2275.0
-44.49	-115.80	2300.0
-47.82	-120.29	2325.0
-51.29	-124.87	2350.0
-54.78	-129.45	2375.0
-58.27	-133.93	2400.0
-61.79	-138.31	2425.0
-65.29	-142.61	2450.0
-68.76	-146.72	2475.0
-72.20	-150.58	2500.0
-77.68	-154.23	2525.0
-79.17	-157.61	2550.0
-82.71	-160.70	2575.0
-86.42	-163.60	2600.0
-90.31	-166.31	2625.0

SOURCE WELL

Wellhead position (m):

x	y	z
551.55	273.09	0.0

Deviations (m):

x	y	z
22.69	-19.40	1550.0
21.15	-16.40	1575.0
17.40	-10.93	1625.0
15.10	-6.91	1650.0
12.49	-3.63	1675.0
9.60	-0.35	1700.0
6.55	2.94	1725.0
3.47	6.13	1750.0
0.47	9.08	1775.0
-2.39	11.79	1800.0
-5.13	14.37	1825.0
-7.76	16.86	1850.0
-10.38	19.15	1875.0
-13.10	21.17	1900.0
-15.91	22.92	1925.0
-18.75	24.30	1950.0
-21.61	25.66	1975.0
-24.36	26.81	2000.0
-26.89	27.85	2025.0
-29.39	28.87	2050.0
-31.95	29.85	2075.0
-34.51	30.75	2100.0
-37.10	31.57	2125.0
-39.79	32.24	2150.0
-42.58	32.77	2175.0
-45.45	33.22	2200.0
-48.37	33.59	2225.0
-51.33	33.84	2250.0
-54.27	33.96	2275.0
-57.18	34.00	2300.0
-60.07	34.01	2325.0
-62.98	33.96	2350.0

Appendix B

xhr1

This program was originally written by M.J. Findlay. However, it has evolved significantly during the course of this study, and it is therefore appropriate to include the main program and subroutines in this thesis. Subroutines not included here are either in **libxhr.a** or **libtsa.a** in the dgl3psr user area on the University of Durham Geological Sciences Department's SUN system. See also the appendix in Findlay (1991).

Plotting routines have not been included as similar programs can be found in UNIRAS manuals, e.g. SEISPLOT, CONTOUR.

```
      PROGRAM XHR1
C
C  Written by M J Findlay  1987-1990
C  Adapted for the SUNs by P S Rowbotham 1990-1993
C
C  Reads seismic data from tape or disc in variety of formats
C  Dataprocessing software all Menu driven
C  Uses subroutine libraries
C        libtsa.a (Time series analysis -- Robinson, Claerbout et al.)
C        libxhr.a (Dataprocessing, display and file I/O -- Findlay)
C        uniras graphics library
C  Requires defaults file xhrp.dfault1.
C
C  NFK1,NFK2 need to be one greater than NSAMR/2,NCHRR
C respectively.   Also need to change PARAMETER in wvshap2.f
C
c PARAMETER(NSAMR=1024,NCHRR=128,NFK1=513,NFK2=129)
c PARAMETER (NSAMR=1024,NCHRR=32,NFK1=513,NFK2=33)
c PARAMETER (NSAMR=512,NCHRR=32,NFK1=257,NFK2=33)
c PARAMETER (NSAMR=1024,NCHRR=32,NFK1=513,NFK2=33)
c PARAMETER (NSAMR=128,NCHRR=4, NFK1=65, NFK2=5)
c PARAMETER (NSAMR=1024,NCHRR=16,NFK1=513,NFK2=17)
c PARAMETER (NSAMR=1024,NCHRR=64,NFK1=513,NFK2=65)
c PARAMETER (NSAMR=512,NCHRR=16, NFK1=257,NFK2=17)
c PARAMETER (NSAMR=256,NCHRR=16, NFK1=129, NFK2=17)
cPARAMETER(NSAMR=2048,NCHRR=128,NFK1=1025,NFK2=129)
c PARAMETER (NSAMR=2048,NCHRR=32,NFK1=1025,NFK2=33)
c PARAMETER (NSAMR=2048,NCHRR=16,NFK1=1025,NFK2=17)
c PARAMETER (NSAMR=512,NCHRR=64, NFK1=257,NFK2=65)
      PARAMETER (NSAMR=2048,NCHRR=64,NFK1=1025,
     1 NFK2=65)
      CHARACTER IPFORM*6, OPFORM*6, IPDISC1*25, OPDISC*25
      CHARACTER IHFORM*4, INPOPT*3, TITOFF*3, CHNSPL*26
      CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, SORLOC
      CHARACTER*20 FILREC, A(9), COMSHT, RECTYP, RECLOC
      CHARACTER*25 PROC(20), IPDISC2, AMSPEC*3,VARARE*3
      CHARACTER TITLE*30, XLB*30, YLB*30, POLART*7
      COMPLEX CPDUM(NSAMR,NCHRR), CWDUM(NCHRR)
      COMPLEX CDUM(NSAMR),CDUM1(NSAMR)
      INTEGER NFILE(2), NCR(2), NFIRST(NCHRR)
      INTEGER IDUM1(NCHRR), IDPROC(5,NCHRR)
      INTEGER IDUM(NCHRR), NFTEMP1(NCHRR)
      INTEGER NFTEMP2(NCHRR), NDUM(NCHRR)
      REAL R4DAT(NSAMR,NCHRR), R2DAT(NSAMR,NCHRR)
      REAL RTEMP1(NSAMR,NCHRR),RTEMP2(NSAMR,NCHRR)
      REAL FILDUM(NFK1,NFK2),AMPDUM(NFK2,NFK1)
      REAL SORPOS, RECDEP(NCHRR), DT, DBGAIN(NCHRR)
      REAL GCMSCL(NCHRR),RDUM2(NSAMR,2)
      REAL RDUMA(NSAMR),RDUMB(NSAMR),RDUMC(NSAMR)
      REAL RDUMD(NSAMR), RDUME(NSAMR), RDUMF(NSAMR)
      REAL RDUM(NSAMR),RIWAVE(NSAMR)
      EQUIVALENCE (LOCATE,A(1)), (DATE,A(2)), (DEVICE,A(3)),
     1          (SORTYP,A(4)), (SORLOC,A(5)), (RECTYP,A(6)),
     2          (RECLOC,A(7)), (COMSHT,A(8)), (FILREC,A(9))
C Set up initial default values - READ IN FROM DEFAULT FILE
      OPEN(UNIT=1,FILE='xhrp.dfault1')
C
      READ (1,10) LOCATE, DATE, DEVICE, SORTYP, SORLOC,
     1RECTYP, RECLOC,COMSHT, FILREC, INPOPT, IPDISC1,
     1IPDISC2, OPDISC, IPFORM, OPFORM, IHFORM
10    FORMAT (A20, 8(/A20)/A3/A25/A25/A25/A6/A6/A4)
      READ (1,30) NSAM, NCHR, NFILES, NFILE(1), NFILE(2),
     1NCR(1), NCR(2), SORPOS, TOPREC, RECSEP, DT, DZ, SCAL
30    FORMAT (I4/I3/I1/I3/I3/I2/I2/F6.1/F6.1/F6.2/F6.1/F6.2/F6.2)
20    READ (1,40) TITOFF, CHNSPL, POLART, VARARE, NPLFST,
     1 NPLLST, IOPT,
     1NCHDEF,TITLE, XLB, YLB
40    FORMAT (A3/A26/A7/A3/I4/I4/I2/I2/A30/A30/A30)
      CLOSE (1)
      IF (NSAM.LT.1024) THEN
        LEN = 4104
      ELSE
        LEN = (NSAM + 2) * 4
      ENDIF
      RECDEP(1) = TOPREC
      DO 50 I = 2, NCHR
        IF (I .EQ. NCR(2)) THEN
          RECDEP(I) = RECDEP(I - 1)
        ELSE
          RECDEP(I) = RECDEP(I - 1) + RECSEP
        END IF
50    CONTINUE
      NSHOT = NFILE(1)
      DO 60 I = 1, NCHR
        IDPROC(1,I) = 0
        NFIRST(I) = 0
        DBGAIN(I) = 0.0
        GCMSCL(I) = 1.0
60    CONTINUE
      NPROCS = 1
      IDPROC(1,NPROCS) = 1
      NCHSV = 0
      IHEAD = 0
      IFPROC = 0
      IFNOHD = 0
      IBAT = 0
      AMSPEC = 'NO'
C
C  DT in microsecs, SAMFRQ in kHz
C
70    IF (IBAT.NE.1) THEN
      PRINT*,'DT=',DT
      WRITE (6,*)
      WRITE (6,*)'           MASTER MENU      (PRESS 0 TO EXIT)'
      WRITE (6,*)'           ~~~~~~ ~~~~                      '
      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,*)
      WRITE (6,*)'   4  FFT traces and plot spectra          '
      WRITE (6,*)'   5  Apply filters to traces              '
      WRITE (6,*)'   6  F-K transform traces                 '
      WRITE (6,*)
      WRITE (6,*)'   7  First breaks, ramps, AGC, tapers etc '
      WRITE (6,*)'   17  Interactively interpolate saturations'
      WRITE (6,*)'   8  Apply wavelet shaping filter         '
      WRITE (6,*)
      WRITE (6,*)'   9  Adjust I/O parameters                '
      WRITE (6,*)'   10  Adjust display parameters           '
      WRITE (6,*)'   11  Adjust processing record            '
      WRITE (6,*)
      WRITE (6,*)'   12  Output first break times            '
      WRITE (6,*)'   13  Save/move traces to temp array      '
      WRITE (6,*)'   14  Secondary save/move traces          '
      WRITE (6,*)'   15  Save traces from temp array         '
      WRITE (6,*)'   (move traces back after wvsh using 14)  '
      WRITE (6,71) IBAT
71    FORMAT('   16  Batch mode (text off = 1)      :',I1)
      WRITE (6,*)
      WRITE (6,*) 'NSHOT=', NSHOT
      ENDIF
      READ (5,*) IANSWR
      IF (NPROCS .GT. 20 .OR. NPROCS .LT. 0) NPROCS = 0
      IF (IANSWR.EQ.1) THEN
       IF (NFILES.EQ.1.AND.IFILE2.NE.1) THEN
         NREAD = 1
         CALL RDFILE(NSAM, NCHR, NSAMR, NCHRR, R4DAT,
     1  NREAD, NFIRST, NRECS, RECDEP, DBGAIN, GCMSCL,
     2  NFILE, NCR,NSHOT, DT, LEN, SORPOS, NPROCS, IDPROC,
     3  IPDISC1, IPDISC2,RDUMA, IHFORM,INPOPT,IPFORM,
     4 IFPROC,IFNOHD,A,PROC,ICSCRG,ISNRN,IBAT)
       ELSE
         IF (IFILE2.EQ.1) THEN
```

```
      NREAD = 2
      CALL RDFILE(NSAM, NCHR, NSAMR, NCHRR, R2DAT,
    1 NREAD, NFIRST, NRECS, RECDEP, DBGAIN, GCMSCL,
    2 NFILE, NCR,NSHOT, DT, LEN, SORPOS, NPROCS, IDPROC,
    3 IPDISC1, IPDISC2,RDUMA, IHFORM,INPOPT,IPFORM,
    4 IFPROC,IFNOHD,A,PROC,ICSCRG,ISNRN,IBAT)
      NFILES = 2
     ELSE
      NREAD = 2
      CALL RDFILE(NSAM, NCHR, NSAMR, NCHRR, R2DAT,
    1 NREAD, NFIRST, NRECS, RECDEP, DBGAIN, GCMSCL,
    2 NFILE, NCR,NSHOT, DT, LEN, SORPOS, NPROCS, IDPROC,
    3 IPDISC1, IPDISC2,RDUMA, IHFORM,INPOPT,IPFORM,
    4 IFPROC,IFNOHD,A,PROC,ICSCRG,ISNRN,IBAT)
      NREAD = 1
      CALL RDFILE(NSAM, NCHR, NSAMR, NCHRR, R4DAT,
    1 NREAD, NFIRST, NRECS, RECDEP, DBGAIN, GCMSCL,
    2 NFILE, NCR,NSHOT, DT, LEN, SORPOS, NPROCS, IDPROC,
    3 IPDISC1, IPDISC2,RDUMA, IHFORM,INPOPT,IPFORM,
    4 IFPROC, IFNOHD,A,PROC,ICSCRG,ISNRN,IBAT)
     ENDIF
     DO 80 I = 1,NSAMR
      DO 90 J = 1,NCHRR
       RTEMP1(I,J) = R4DAT(I,J)
90     CONTINUE
80    CONTINUE
    ENDIF
   ENDIF
   IF (IANSWR .EQ. 1) AMSPEC = 'NO'
   IF (IANSWR .EQ. 2) CALL WRFILA(NSAM, NCHR, NSAMR,
    1 NCHRR, R4DAT, NFIRST, NRECS, RECDEP, DBGAIN,
    2 GCMSCL, NCR, NSHOT, DT, LEN,  SORPOS, NPROCS,
    3 IDPROC, OPDISC, IHFORM, OPFORM, LOCATE, DATE,
    3 DEVICE, SORTYP, SORLOC, RECTYP, RECLOC, COMSHT,
    5 FILREC, PROC, IDUM, DBGAIN, IHEAD, ICSCRG, IBAT)
   IF (IANSWR .EQ. 3)
    1 WRITE (6,*)'NSAMS,NCHANS =', NSAM, NCHR
   IF (IANSWR .EQ. 3) CALL MENPL2(NSAM, NCHR, NSAMR,
    1 NCHRR, R4DAT, NFIRST, TITOFF, CHNSPL, POLART,
    2 VARARE, NPLFST, NPLLST, IOPT, SORPOS, RECDEP, NCR,
    3 DT, LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP,
    4 RECLOC, COMSHT, FILREC, NSHOT, NPROCS, IDPROC,
    5 TITLE, XLB, YLB, SCAL, AMSPEC, PROC)
   IF (IANSWR .EQ. 4) CALL MENUD(NSAM, NCHR, NSAMR,
    1 NCHRR, R4DAT, DT, NSHOT, AMSPEC, CDUM, CDUM1,
    2 RDUM2, RDUM2, RDUMA, RDUMB, RDUMC, RDUMD,
    3 RDUME, RDUMF)
   IF (IANSWR .EQ. 5) CALL MENUE(NSAM, NCHR, NSAMR,
    1 NCHRR, R4DAT, NFIRST, DT, NPROCS, IDPROC, CDUM,
    2 RDUMA, RDUMB, RDUMC, RDUMD, RTEMP1,IBAT)
   IF (IANSWR .EQ. 6) CALL MENUF(NSAM, NCHR, NSAMR,
    1 NCHRR, NFK1,NFK2, R4DAT, DZ, DT, NCR, CPDUM,
    2 CWDUM,FILDUM,AMPDUM,IBAT)
   IF (IANSWR .EQ. 7) CALL GNCOMP(NSAM, NCHR, NSAMR,
    1 NCHRR, R4DAT, R2DAT, NFIRST, DBGAIN, NSHOT,
    2 GCMSCL, RECDEP, NCR, IDPROC, NPROCS, SORPOS, DT,
    3 RDUMA, RDUMB, RDUMC, RIWAVE, IDUM,
    4   NDUM, IDUM1, ICSCRG, IBAT)
   IF (NFILES.EQ.1) THEN
    IF (IANSWR .EQ. 8) CALL WVSHAP(NSAM, NCHR, NSAMR,
   1NCHRR, R4DAT, R2DAT, NFILES, DT, NFIRST, NSHOT, IBAT)
   ELSE
    IF (IANSWR .EQ. 8) CALL WVSHAP(NSAM, NCHR, NSAMR,
    1 NCHRR, RTEMP1, R2DAT, NFILES, DT, NFIRST, NSHOT,
    2 IBAT)
   ENDIF
   IF (IANSWR .EQ. 9) CALL MENUIO(NSAM, NCHR, NRECS,
    1 NSHOT, LEN, NFILES, NFILE, IPDISC1, IPDISC2, IHFORM,
    2 INPOPT, OPDISC, IPFORM,OPFORM,IHEAD,IFPROC,
    3 IFNOHD,IFILE2,ICSCRG,ISNRN,IBAT)
   IF (IANSWR .EQ. 9) THEN
    IF (NSAM.LT.1024) THEN
     LEN = 4104
    ELSE
     LEN = (NSAM + 2) * 4
    ENDIF
   ENDIF
   IF (IANSWR .EQ. 10) CALL MENUDS(NCHR,NCHRR,SORPOS,
    1 RECDEP, NCR,DT, LOCATE, DATE, DEVICE, SORTYP,
    2 SORLOC, RECTYP, RECLOC, COMSHT, FILREC, TITLE,
    3 XLB, YLB)
   IF (IANSWR .EQ. 11) CALL PROCES(NPROCS, PROC)
   IF (IANSWR .EQ. 12) CALL OPFIRS(NCHR, NCHRR, NFIRST,
    1   RECDEP, LEN, IPDISC1)
   IF (IANSWR .EQ. 13) CALL SAVETR(NSAM,NCHR,NSAMR,
    1 NCHRR,R4DAT,NFIRST,RTEMP1,NFTEMP1, NCHSV1,
    2 NCHDEF,IBAT)
   IF (IANSWR .EQ. 14) CALL SAVETR(NSAM,NCHR,NSAMR,
    1 NCHRR,R4DAT,NFIRST,RTEMP2,NFTEMP2, NCHSV2,
    2 NCHDEF,IBAT)
   IF (IANSWR .EQ. 15) CALL SAVETR(NSAM,NCHR,NSAMR,
    1 NCHRR,RTEMP1, NFIRST,RTEMP2,NFTEMP2,NCHSV2,
    2 NCHDEF,IBAT)
   IF (IANSWR .EQ. 16) THEN
    IF (IBAT.EQ.0) THEN
     IBAT = 1
    ELSE
     IBAT = 0
    ENDIF
   ENDIF
   IF (IANSWR .EQ. 0) CALL EXIT(0)
c   DZ = ABS(RECDEP(2) - RECDEP(1))
   IF (DZ .EQ. 0.0) DZ = 1.0
   NN=NSAMR*2
   NFK=NFK1*NFK2
   CALL ZERO(NSAMR, RDUM)
   CALL ZERO(NSAMR, RDUMA)
   CALL ZERO(NSAMR, RDUMB)
   CALL ZERO(NSAMR, RDUMC)
   CALL ZERO(NSAMR, RDUMD)
   CALL ZERO(NSAMR, RDUME)
   CALL ZERO(NSAMR, RDUMF)
   CALL ZERO(NCHRR, IDUM)
   CALL ZERO(NCHRR, NDUM)
   CALL ZERO(NFK, AMPDUM)
   CALL ZERO(NFK, FILDUM)
   CALL ZERO(NN, CDUM)
   GO TO 70
   END

c--------------------------------------------------------------------
C Adapted by Peter Rowbotham 1992 from alignd.f (Ed Kragh 1991)
   SUBROUTINE ALIGNP(NSAM,NCHR,NSAMR,NCHRR,R4DAT,
  1   NFIRST,TEMP,NFB)
C   Shifts data by first break pick ... Aligns data at given NFB
C   IS = +1 then +ve shift
C   IS = -1 then -ve shift .... to shift data back1
   REAL    R4DAT(NSAMR,NCHRR),TEMP(NSAMR)
   INTEGER   NFIRST(NCHRR)
c
   DO 100 I = 1,NCHR
    PRINT*,' I = ',I
    DO 110 J = 1,NSAM
      TEMP(J) = R4DAT(J,I)
      R4DAT(J,I) = 0.0
110   CONTINUE
    NFBDIF = NFIRST(I) - NFB
    NLIMIT = NSAM - NFBDIF
    IF (NFBDIF .EQ. ABS(NFBDIF)) THEN
     I1 = 1
    ELSE
     I1 = -1
    ENDIF
    DO 130 J = 1, NSAM
```

```
      IF (I1 .EQ. 1) THEN
       IF (J .GE. NLIMIT) THEN
        R4DAT(J,I) = 0.0
       ELSE
        R4DAT(J,I) = TEMP(J+NFBDIF)
       ENDIF
      ELSE
       IF (J .LE. ABS(NFBDIF)) THEN
        R4DAT(J,I) = 0.0
       ELSE
        R4DAT(J,I) = TEMP(J+NFBDIF)
       ENDIF
      ENDIF
130   CONTINUE
100 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 carry out simple 2D Fourier transform (Claerbout)
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 FFT(N1, CP(1,I), SIGNA)
  10 CONTINUE
    DO 40 J = 1, N1
     DO 20 K = 1, N2
      CW(K) = CP(J,K)
  20  CONTINUE
     CALL FFT(N2, CW, SIGNB)
     DO 30 L = 1, N2
      CP(J,L) = CW(L)
  30  CONTINUE
  40 CONTINUE
    RETURN
    END
c-------------------------------------------------------------------
C.Subroutine to carry out simple 3D Fourier transform
C
    SUBROUTINE FT3D(N1, N2, N3, CP, SIGNA, SIGNB,
   1   SIGNC, CW1, CW2)
    INTEGER N1, N2, N3
    COMPLEX CP(N1,N2,N3), CW1(N2,N3), CW2(N3)
    REAL SIGNA, SIGNB, SIGNC
    DO 10 I = 1, N3
     CALL FFT(N1, CP(1,1,I), SIGNA)
10 CONTINUE
    DO 20 J = 1, N1
     DO 30 K = 1, N2
      DO 40 L = 1, N3
       CW1(K,L) = CP(J,K,L)
40     CONTINUE
30    CONTINUE
     CALL FFT(N2, CW1, SIGNB)
     DO 50 K = 1, N2
      DO 60 L = 1, N3
        CW2(L) = CW1(K,L)
60     CONTINUE
      CALL FFT(N3, CW2, SIGNC)
      DO 70 L = 1, N3
       CW1(K,L) = CW2(L)
70     CONTINUE
50    CONTINUE
     DO 80 K = 1, N2
      CP(J,K,L) = CW1(K,L)
80    CONTINUE
20 CONTINUE
    RETURN
    END

c-------------------------------------------------------------------
C. Subroutine to apply gain compensation across array of traces
C
    SUBROUTINE GNCOMP(NSAM, NCHR, NSAMR, NCHRR,
   1 R4DAT, R2DAT, NFIRST, DBGAIN,NSHOT,GCMSCL,
   2 RECDEP,NCR,IDPROC, NPROCS, SORPOS, DT, TEMP,
   3 XTEMP2, XTEMP3, A, KILLID, NTEM, IDUM1, ICSCRG,
   4 IBAT)
    DIMENSION R4DAT(NSAMR,NCHRR), NFIRST(NCHRR)
    DIMENSION DBGAIN(NCHRR),R2DAT(NSAMR,NCHRR)
    DIMENSION GCMSCL(NCHRR), KILLID(NCHRR),
    DIMENSION RECDEP(NCHRR), IDPROC(5,NCHRR), NCR(2)
    DIMENSION TEMP(NSAMR),XTEMP3(NSAMR)
    DIMENSION A(NSAMR),NTEM(NCHRR),XTEMP2(NSAMR)
    DIMENSION IDUM1(NCHRR)
    CHARACTER ANS*1, FNAME*20, ANSIN*1
    INTEGER NREPLY, NSHOT, NEW
    REAL RSCAL(2),XSO(51),ZSO(51),XRE(51),ZRE(51)
c   COMPLEX CX(4096)
    IF (IBAT.NE.1) WRITE (6,*)'NSAM,NCHR,NSHOT = ',
   1  NSAM,NCHR,NSHOT
10 IF (IBAT.NE.1) THEN
    PRINT *,'                                          '
    PRINT *,'          GAIN COMPENSATION MENU          '
    PRINT *,'          ---- ------------ ----          '
    WRITE (6,20)
20 FORMAT (/' 1   Normalise each trace wrt rms energy     ')
    WRITE (6,30)
30 FORMAT (' 2   Normalise each trace wrt peak amplitude ')
    WRITE (6,40)
40 FORMAT (/' 3   Kill / Mute (taper) traces              ')
    WRITE (6,50)
50 FORMAT (' 4   Apply AGC or T-squared ramp to traces   ')
    WRITE (6,60)
60 FORMAT (' 5   Apply tapers to data spatially          ')
    WRITE (6,70)
70 FORMAT (/' 6   Remove common trace                     ')
    WRITE (6,80)
80 FORMAT (' 7   Remove spike / D.C. from data           ')
    WRITE (6,90)
90 FORMAT (' 8   Correct trace misalignment              ')
    WRITE (6,100)
100 FORMAT (' 9   Apply / Enter channel gains             ')
    WRITE (6,110)
110 FORMAT (/' 10  Enter or alter first break samples      ')
    WRITE (6,120)
120 FORMAT (' 11  Apply mute to channels   (Cos taper)    ')
    WRITE (6,121)
121 FORMAT (/' 12  Shift traces using Eds ALIGND subroutine')
    WRITE (6,122)
122 FORMAT (' 13  Sum traces to first trace               ')
    WRITE (6,123)
123 FORMAT (' 14  Shift first trace back to true position ')
    WRITE (6,124)
124 FORMAT (/' 15  Create traces (entered or sine)         ')
    WRITE (6,125)
125 FORMAT (' 16  Output first breaks in runfile format   ')
    WRITE (6,126)
126 FORMAT (/' 17  Energy in a given window                ')
    WRITE (6,127)
127 FORMAT (' 18  Norm same traces in 2 files wrt rms nrg ')
    WRITE (6,128)
128 FORMAT (' 19  Norm same traces in 2 files wrt peak amp')
    WRITE (6,129)
129 FORMAT (' 20  Stack 2 files                           ')
```

```
      WRITE (6,131)
131 FORMAT (' 21   Input wavelet for Ed's pro16 ACF program ')
      WRITE (6,132)
132 FORMAT (' 22   Ed's pro16 ACF program (altered)        ')
      WRITE (6,133)
133 FORMAT (' 23   Max amp for each trace                ')
      WRITE (6,134)
134 FORMAT (' 24   Enter scaling factor                  ')
      WRITE (6,135)
135 FORMAT (' 25   Normalise by S-R separation           ')
      PRINT*
      ENDIF
      READ (5,*) IOPT
      IF (IOPT .EQ. 1) THEN
       NPROCS = NPROCS + 1
       IDPROC(1,NPROCS) = 2
       IDPROC(2,NPROCS) = 0
       DO 130 J = 1, NCHR
        CALL RMSERR(NSAM, R4DAT(1,J), TEMP(J))
130  CONTINUE
       CALL MAXSN(NCHR, TEMP, RMSMAX, MAXTRC)
       DO 150 J = 1, NCHR
        IF (TEMP(J) .EQ. 0.0) GO TO 150
        SCALE = RMSMAX / TEMP(J)
        IF (IBAT.NE.1) PRINT *, 'SCALE #', J, ' = ', SCALE
        GCMSCL(J) = GCMSCL(J) * SCALE
        DO 140 I = 1, NSAM
         R4DAT(I,J) = R4DAT(I,J) * SCALE
140    CONTINUE
150  .CONTINUE
       LENGCM = (NCHRR+2) * 4
       IGCM = NSHOT
       IF (IBAT.NE.1) PRINT *, GCMSCL
      ELSE IF (IOPT .EQ. 2) THEN
       NPROCS = NPROCS + 1
       IDPROC(1,NPROCS) = 2
       IDPROC(2,NPROCS) = 1
       DO 160 J = 1, NCHR
        CALL MAXAMP(NSAM, R4DAT(1,J), TEMP(J), INDAMP)
160  CONTINUE
       CALL MAXSN(NCHR, TEMP, BIGAMP, II)
       DO 180 J = 1, NCHR
        IF (TEMP(J) .EQ. 0) GO TO 180
        SCALE = BIGAMP / TEMP(J)
        GCMSCL(J) = GCMSCL(J) * SCALE
        DO 170 I = 1, NSAM
         R4DAT(I,J) = R4DAT(I,J) * SCALE
170    CONTINUE
180  CONTINUE
       IF (IBAT.NE.1) PRINT *, GCMSCL
      ELSE IF (IOPT .EQ. 3) THEN
       IF (IBAT.NE.1) THEN
       PRINT *, '1 KILLS channels; 0 TAPERS ends :2 TAPERS past Sfbs'
       PRINT *, '3:TAPERS past Pfbs'
       PRINT *, '(will act on both files if 2 have been read in)'
       ENDIF
       READ (5,*) IKM
       IF (IKM .EQ. 1) THEN
        IF (IBAT.NE.1) PRINT *, 'Enter No. of channels to be killed:'
        READ (5,*) NKILLS
        DO 190 I = 1, NKILLS
         IF (IBAT.NE.1) PRINT *, ' Kill #', I, ' : '
         READ (5,*) KILLID(I)
190    CONTINUE
        DO 200 I = 1, NKILLS
         CALL ZERO(NSAM, R4DAT(1,KILLID(I)))
         CALL ZERO(NSAM, R2DAT(1,KILLID(I)))
200    CONTINUE
       ELSE
        IF (IBAT.NE.1) THEN
        WRITE (6,*)'START AND END OF TAPER IN SAMPLES :'
        WRITE (6,*)'(Relative to Swv FBs if 2 chosen above)'
        WRITE (6,*)'(TAPERS TO ZERO AT TAPER END)'
        ENDIF
        READ (5,*) NTPST, NTPEND
        IF (IBAT.NE.1)
     1   WRITE(6,*)'FIRST,LAST TRACE TO APPLY (0,0 ALL):'
        READ (5,*) NTR0, NTR1
        IF (NTR0 .EQ. 0 .AND. NTR1 .EQ. 0) THEN
         NTR0 = 1
         NTR1 = NCHR
        ENDIF
        IF (IKM.EQ.2) THEN
         OPEN(UNIT=8,FILE='fortt.3')
         IF (ICSCRG.EQ.1) THEN
          NEW = NSHOT - 1
         ELSE
          NEW = NCHR*(NSHOT-1)
         ENDIF
         DO 201 I=1,NEW
          READ(8,*)
201     CONTINUE
         DO 202 I = 1,NCHR
          READ (8,*) NTEM1,NTEM2,NTEM(I)
          IF (NTEM2.NE.NFIRST(I)) THEN
           PRINT*,'uh oh, non matching fbs'
          ENDIF
          IF (ICSCRG.EQ.1) THEN
           DO 203 J=1,NCHR-1
            READ(8,*)
203        CONTINUE
          ENDIF
202     CONTINUE
         CLOSE(UNIT=8)
        ENDIF
        IF (IKM.EQ.3) THEN
         DO 204 I = 1,NCHR
          NTEM(I) = NFIRST(I)
204     CONTINUE
        ENDIF
        LENTAP = ABS(NTPEND - NTPST) + 1
        IF (NTPST .LT. NTPEND) THEN
         DO 230 J = NTR0, NTR1
          IF (IKM .EQ. 2.OR.IKM.EQ.3) THEN
           NTP1 = NTPST + NTEM(J)
           NTP2 = NTPEND + NTEM(J)
          ELSE
           NTP1 = NTPST
           NTP2 = NTPEND
          ENDIF
          DO 210 I = 1, LENTAP
           FAC = REAL(LENTAP - I) / REAL(LENTAP)
           R4DAT(I + NTP1,J) = R4DAT(I + NTP1,J) * FAC
           R2DAT(I + NTP1,J) = R2DAT(I + NTP1,J) * FAC
210      CONTINUE
          DO 220 I = NTP2 + 2, NSAM
           R4DAT(I,J) = 0.0
           R2DAT(I,J) = 0.0
220      CONTINUE
230     CONTINUE
        ELSE
         DO 260 J = NTR0, NTR1
          DO 240 I = 1, LENTAP
           FAC = REAL(LENTAP - I) / REAL(LENTAP)
           R4DAT(NTPST - I,J) = R4DAT(NTPST - I,J) * FAC
           R2DAT(NTPST - I,J) = R2DAT(NTPST - I,J) * FAC
240      CONTINUE
          DO 250 I = 1, NTPEND - 1
           R4DAT(I,J) = 0.0
           R2DAT(I,J) = 0.0
250      CONTINUE
260     CONTINUE
        END IF
       END IF
```

```
      CALL ZERO(NCHRR,NTEM)
     ELSE IF (IOPT .EQ. 4) THEN
      NPROCS = NPROCS + 1
      IDPROC(1,NPROCS) = 6
      IF (IBAT.NE.1) WRITE (6,*)'Ramp or AGC or reDefine data :'
      READ (5,490) ANS
      IF (ANS .EQ. 'R' .OR. ANS .EQ. 'r') THEN
       DO 280 J = 1, NCHR
        DO 270 I = 1, NSAM
         R4DAT(I,J) = R4DAT(I,J) * (I - 1) ** 2 / 10000
270     CONTINUE
280    CONTINUE
      IF (IBAT.NE.1) WRITE (6,*)'Taper (Y/N) '
      READ (5,490) ANS
      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
       IF (IBAT.NE.1) WRITE (6,*)'Enter length of taper in sams :'
       READ (5,*) LENTAP
       NTAP0 = NSAM - LENTAP
       DO 300 J = 1, NCHR
        DO 290 I = NTAP0, NSAM
         TAP = 1.0 - REAL(I - NTAP0) / REAL(LENTAP)
         IF (I .GT. NTAP0 + LENTAP) TAP = 0.0
         R4DAT(I,J) = R4DAT(I,J) * TAP
290      CONTINUE
300     CONTINUE
      END IF
     ELSE IF (ANS .EQ. 'D' .OR. ANS .EQ. 'd') THEN
      IF (IBAT.NE.1) THEN
      WRITE (6,*)'ENTER SPIKE (1),FLAT(2),T-RAMP(3),
    1 T1/2-RAMP(4)'
      WRITE (6,*)'T-.5 RAMP(5)'
      ENDIF
      READ (5,*) IOP
      IF (IOP .EQ. 1) THEN
       DO 310 J = 1, NCHR
        CALL ZERO(NSAM, R4DAT(1,J))
        R4DAT(NSAM/2 + 1,J) = 1.0
310     CONTINUE
      ELSE IF (IOP .EQ. 2) THEN
       DO 330 J = 1, NCHR
        DO 320 I = 1, NSAM
         R4DAT(I,J) = 1.0
320      CONTINUE
330     CONTINUE
      ELSE IF (IOP .EQ. 3) THEN
       DO 350 J = 1, NCHR
        DO 340 I = 1, NSAM
         R4DAT(I,J) = REAL(I - 1)
340      CONTINUE
350     CONTINUE
      ELSE IF (IOP .EQ. 4) THEN
       DO 370 J = 1, NCHR
        DO 360 I = 1, NSAM
         R4DAT(I,J) = SQRT(REAL(I - 1))
360      CONTINUE
370     CONTINUE
      ELSE IF (IOP .EQ. 5) THEN
       DO 390 J = 1, NCHR
        DO 380 I = 1, NSAM
         R4DAT(I,J) = 1. / SQRT(REAL(I))
380      CONTINUE
390     CONTINUE
      END IF
     ELSE
      IF (IBAT.NE.1) WRITE (6,*)'Enter length of AGC win in sams:'
      READ (5,*) LENAGC
      IDPROC(2,NPROCS) = LENAGC
      DO 400 J = 1, NCHR
       CALL AGC(NSAM, 1, R4DAT(1,J), LENAGC)
400   CONTINUE
     END IF
    ELSE IF (IOPT .EQ. 5) THEN
     IF (IBAT.NE.1) WRITE (6,*)'No of traces to apply taper over' //
    1   '(affects L-1 traces):'
     READ (5,*) LTAPER
     IF (IBAT.NE.1) WRITE (6,*)'Enter first and last traces with ' //
    1   'significant data:'
     READ (5,*) NTRAC0, NTRAC1
     PI = 3.1415926535
     IF (IBAT.NE.1) WRITE (6,*)'Apply / Remove Taper (A/R):'
     READ (5,490) ANS
     DO 420 J = 1, LTAPER
      THETA = PI * REAL(J) / REAL(LTAPER)
      FAC = 0.5 * (1.0 - COS(THETA))
      IF (ANS .EQ. 'R' .OR. ANS .EQ. 'r') FAC = 1.0 / FAC
      NTR0 = NTRAC0 - 1 + J
      NTR1 = NTRAC1 + 1 - J
      IF (NTR0 .LE. 0 .OR. NTR1 .GT. NCHR) GO TO 420
      DO 410 I = 1, NSAM
       R4DAT(I,NTR0) = R4DAT(I,NTR0) * FAC
       R4DAT(I,NTR1) = R4DAT(I,NTR1) * FAC
410   CONTINUE
420  CONTINUE
    ELSE IF (IOPT .EQ. 6) THEN
     IF (IBAT.NE.1) WRITE (6,*)'Enter no. of channel to be removed :'
     READ (5,*) NCHNL0
     DO 440 J = NCHNL0, NCHR - 1
      NFIRST(J) = NFIRST(J + 1)
      RECDEP(J) = RECDEP(J + 1)
      DO 430 I = 1, NSAM
       R4DAT(I,J) = R4DAT(I,J + 1)
430   CONTINUE
440  CONTINUE
     CALL ZERO(NSAM, R4DAT(1,NCHR))
     NFIRST(NCHR) = 0
     RECDEP(NCHR)=RECDEP(2)-RECDEP(1)+RECDEP(NCHR-1)
    ELSE IF (IOPT .EQ. 7) THEN
     IF (IBAT.NE.1) 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')
    1      THEN
      IF (IBAT.NE.1) WRITE (6,*)'ENTER TRACE NUMBER  :'
      READ (5,*) NTRSPK
      IF (IBAT.NE.1) 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*(R4DAT(I-1,NTRSPK)+R4DAT(I+1,NTRSPK))
       END IF
       R4DAT(I,NTRSPK) = ALPHA
450   CONTINUE
     ELSE
      DO 480 J = 1, NCHR
       DC = 0.0
       DO 460 I = 1, NSAM
        DC = DC + R4DAT(I,J)
460     CONTINUE
       DC = DC / REAL(NSAM)
       IF (IBAT.NE.1) PRINT *, ' TRACE ', J, '  DC = ', DC
       DO 470 I = 1, NSAM
        R4DAT(I,J) = R4DAT(I,J) - DC
470     CONTINUE
480    CONTINUE
     END IF
    ELSE IF (IOPT .EQ. 8) THEN
     IF (IBAT.NE.1) WRITE (6,*)'CALC COM TRACE
    1  MISALIGNMENT (1=Y)?'
     READ (5,*) ICORR
     IF (ICORR .EQ. 1) THEN
      IF (IBAT.NE.1) WRITE (6,*)'ENTER CHANNELS #1, #2  :'
      READ (5,*) I1, I2
      IF (IBAT.NE.1) WRITE (6,*)'ENTER MAX LAG :'
```

```
      READ (5,*) LG
      CALL CROSS(NSAM, R4DAT(1,I1), NSAM, R4DAT(1,I2),
     1   LG, TEMP)
      CALL MAXSN(LG, TEMP, GMAX, II)
      CALL CROSS(NSAM, R4DAT(1,I2), NSAM, R4DAT(1,I1),
     1  LG, TEMP)
      CALL MAXSN(LG, TEMP, GMAX1, II1)
      IF (IBAT.NE.1) THEN
        WRITE (6,*) GMAX, II, GMAX1, II1
        IF (GMAX1 .GT. GMAX) THEN
          WRITE (6,*)'TRACE 2 LEADS TRACE1 BY :', II1 - 1
        ELSE
          WRITE (6,*)'TRACE 2 LAGS TRACE1 BY:', II - 1
        END IF
        WRITE (6,*)'LAG implies INCREASE times on trace2.'
        WRITE (6,*)'ENTER 0 TO EXIT 1 TO CONTINUE:'
       ENDIF
       READ (5,*) IGO
       IF (IGO .NE. 1) GO TO 570
      END IF
      IF (IBAT.NE.1)
     1 WRITE (6,*)'Shift single or range of traces(S/R)?'
      READ (5,490) ANS
 490  FORMAT (A1)
      IF (ANS .EQ. 'S' .OR. ANS .EQ. 's') THEN
       IF (IBAT.NE.1) WRITE (6,*)'Enter no. of trace :'
       READ (5,*) NTRA
       IF (IBAT.NE.1) WRITE (6,*)'+ shift reduces travel time!'
       IF (IBAT.NE.1) WRITE (6,*)'Enter shift (rotatl) to apply :'
       READ (5,*) NCHANG
       IF (IBAT.NE.1) WRITE (6,*)'ENTER FIRST SAMPLE TO
SHIFT :'
       READ (5,*) ITOSHF
       DO 500 I = ITOSHF, NSAM - NCHANG
        IF (I .GT. NSAM) GO TO 500
        IF ((I + NCHANG) .LT. ITOSHF) THEN
         TEMP(I) = 0.0
         GO TO 500
        ELSE
         TEMP(I) = R4DAT(I + NCHANG,NTRA)
        END IF
 500   CONTINUE
       DO 510 I = NSAM - NCHANG + 1, NSAM
        IF (I .GT. NSAM) GO TO 510
        TEMP(I) = R4DAT(I - NSAM + NCHANG,NTRA)
 510   CONTINUE
       DO 520 I = 1, NSAM
        R4DAT(I,NTRA) = TEMP(I)
 520   CONTINUE
       NFIRST(NTRA) = NFIRST(NTRA) - NCHANG
       IF (IBAT.NE.1) WRITE (6,*)'Taper the trace (Y/N) ?'
       READ (5,490) ANS
       IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
        IF (IBAT.NE.1) WRITE (6,*)'Enter start of tap in sams :'
        READ (5,*) NTAP0
        IF (IBAT.NE.1) WRITE (6,*)'Enter length of tap in sams :'
        READ (5,*) LENTAP
        DO 530 I = NTAP0, NSAM
         TAP = 1.0 - REAL(I - NTAP0) / REAL(LENTAP)
         IF (I .GT. NTAP0 + LENTAP) TAP = 0.0
         R4DAT(I,NTRA) = R4DAT(I,NTRA) * TAP
 530    CONTINUE
       END IF
      ELSE
       IF (IBAT.NE.1) WRITE (6,*)'1ST/LAST TRACES TO SHIFT : '
       READ (5,*) NTOGO0, NTOGO1
       IF (IBAT.NE.1) THEN
        WRITE (6,*)'+ shift reduces travel time of traces!'
        WRITE (6,*)'ENTER SHIFT TO APPLY IN SAMPLES :'
       ENDIF
       READ (5,*) NSHIFT
       DO 570 J = NTOGO0, NTOGO1
        NFIRST(J) = NFIRST(J) - NSHIFT
        AVER0 = 0.0
        DO 540 I = 1, 50
         AVER0 = AVER0 + R4DAT(I,J)
 540    CONTINUE
        AVER0 = AVER0 / 50.0
        AVERAG = 0.0
        DO 550 I = NSAM / 2 + 1, NSAM
         AVERAG = AVERAG + R4DAT(I,J)
 550    CONTINUE
        AVERAG = 2.0 * AVERAG / REAL(NSAM)
        ISHIFT = NSHIFT
        ISIG = ISIGN(1,ISHIFT)
        I0 = 1
        IL = NSAM
        IF (ISIG .LT. 0) THEN
         I0 = NSAM
         IL = 1
        END IF
        DO 560 I = I0, IL, ISIG
         IF (I + ISHIFT .LE. 0) THEN
          R4DAT(I,J) = AVER0
         ELSE IF (I + ISHIFT .LE. NSAM) THEN
          R4DAT(I,J) = R4DAT(I + ISHIFT,J)
         ELSE
          R4DAT(I,J) = AVERAG
         END IF
 560    CONTINUE
 570   CONTINUE
      END IF
     ELSE IF (IOPT .EQ. 9) THEN
      IF (IBAT.NE.1) WRITE (6,*)'Apply current/New gains (A/N)?'
      READ (5,490) ANS
      IF (ANS .EQ. 'A' .OR. ANS .EQ. 'a') THEN
       NCR(1) = 12
       NCR(2) = 14
       IF (NCR(1) .EQ. 12 .AND. NCR(2) .GE. 13) THEN
        CALL ZERO(NCHR, DBGAIN)
       END IF
       DO 590 J = 1, NCHR
        RMULT = 10 ** (-DBGAIN(J)/20.0)
        DBGAIN(J) = 0.0
        DO 580 I = 1, NSAM
         R4DAT(I,J) = R4DAT(I,J) * RMULT
 580    CONTINUE
 590   CONTINUE
       NPROCS = NPROCS + 1
       IDPROC(1,NPROCS) = 1
       IDPROC(2,NPROCS) = 0
       IF (IBAT.NE.1) WRITE (6,*)'Equate rms amps of com rec chans ?'
       READ (5,490) ANS
       IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
        CALL RMSERR(NSAM,R4DAT(1,NCR(1)),SCALE1,INDAMP)
        CALL RMSERR(NSAM,R4DAT(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, NSAM
            R4DAT(I,J) = R4DAT(I,J) * SCALE
 600       CONTINUE
 610      CONTINUE
         ELSE
C ASSUME NCR(1) IS HIGHER COMMON TRACE
C        AND   COMMON SHOT TRACES ARE ALTERNATE
          IDTRAC = 2 - MOD(NCR(2),2)
          IF (IBAT.NE.1) PRINT*,'IDTRAC=',IDTRAC,'SHOULD BE 1'
          DO 630 J = IDTRAC, NCHR, 2
           DO 620 I = 1, NSAM
            R4DAT(I,J) = R4DAT(I,J) * SCALE
 620       CONTINUE
```

```
630      CONTINUE
          IDPROC(2,NPROCS) = 1
         END IF
        END IF
       ELSE
        IF (IBAT.NE.1) WRITE (6,*)'Same gain for each trace 1=y'
        READ(5,*) IYNGA
        IF (IYNGA.EQ.1) THEN
         IF (IBAT.NE.1) WRITE (6,*)'Enter SCALING FACTOR - not
gain'
         IF (IBAT.NE.1) WRITE (6,*)'Scaling applied now '
         READ (5,*) DBGALL
         DO 641 J = 1, NCHR
          DO 642 I = 1, NSAM
           R4DAT(I,J) = R4DAT(I,J) * DBGALL
642       CONTINUE
641      CONTINUE
        ELSE
         DO 640 I = 1, NCHR
         IF (IBAT.NE.1) WRITE (6,*)'Enter gain in dB for chan #', I
          READ (5,*) DBGAIN(I)
640      CONTINUE
        END IF
       END IF
      ELSE IF (IOPT .EQ. 10) THEN
       IF (IBAT.NE.1)  WRITE (6,*)'AUTO PICK:
     1 1=Y 0=N 2/4=tomo/swave file 3=shift?'
       READ (5,*) IREPLY
       IF (IREPLY .EQ. 1) THEN
        CALL ZERO(NCHRR,NFIRST)
649     IF (IBAT.NE.1) WRITE (6,*)'ENTER 1ST SAM
     1   TO BEGIN ESTIMATE:'
        IF (IBAT.NE.1) WRITE (6,*)'(999=find max amp):'
        READ (5,*) N0
        IF (N0.EQ.999) THEN
         IMAXFB = 1
         GOTO 649
        ENDIF
        IF (IMAXFB.EQ.1) THEN
         DO 648 J = 1, NCHR
          RMAXFB = 1E-9
          DO 648 I = N0, NSAM
           IF (R4DAT(I,J).GT.RMAXFB) THEN
            RMAXFB = R4DAT(I,J)
            NFIRST(J) = I
           ENDIF
648      CONTINUE
        ELSE
         IF (IBAT.NE.1) WRITE (6,*)'ENTER signif no. of S.D.s:'
         READ (5,*) SSDS
         DO 680 J = 1, NCHR
          DO 670 I = N0, NSAM
           RMEAN = 0.0
           DO 650 III = 1, I - 1
            RMEAN = RMEAN + R4DAT(III,J)
650        CONTINUE
           RMEAN = RMEAN / REAL(I - 1)
           RMERR = 0.0
           DO 660 III = 1, I - 1
            DIF = R4DAT(III,J) - RMEAN
            RMERR = RMERR + DIF ** 2
660        CONTINUE
           RMERR = RMERR / REAL(I - 1)
           RMERR = SQRT(RMERR)
           DIF0 = ABS(R4DAT(I,J) - R4DAT(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
670       CONTINUE
680      CONTINUE
        ENDIF
        IMAXFB = 0
       ELSE IF (IREPLY .EQ. 0) THEN
        NREPLY = 0
        DO 690 I = 1, NCHR
         IF (NREPLY.NE.99) THEN
          IF (IBAT.NE.1) THEN
           WRITE (6,*)'1st break SAMPLE for channel #', I ,
     1'is', NFIRST(I)
           WRITE (6,*)'Change it ? Y=1, N=0, change all = 99'
          ENDIF
          READ (5,*) NREPLY
         ENDIF
         IF (NREPLY.EQ.1.OR.NREPLY.EQ.99) THEN
          IF (IBAT.NE.1)
     1   WRITE (6,*)'Enter 1st break SAMPLE for channel #', I
          READ (5,*) NFIRST(I)
         ENDIF
690     CONTINUE
       ELSE IF (IREPLY .EQ. 2) THEN
        N1SHOT = NSHOT
        IF (IBAT.NE.1) WRITE (6,*)'down(1) or up(-1)'
        READ(5,*) IDUP
        IF (IBAT.NE.1) WRITE (6,*)'File name'
        READ(5,689) FNAME
689     FORMAT(A20)
        CALL ZERO(NCHRR,NFIRST)
        NEW = 0
        OPEN(UNIT=8,FILE=FNAME)
        DO 691 I = 1, 5
         READ (8,*)
691     CONTINUE
        PRINT*,'SHIFT BY 9.521 (1=Y)'
        READ(5,*) ISHYN
        IF(ISHYN.NE.1) ISHYN = 0
        IF (IDUP.EQ.1) THEN
         NEW = NCHR*(N1SHOT-1)
        ELSE
         NEW = NCHR*(NCHR-N1SHOT)
        ENDIF
        WRITE (6,*) NEW,N1SHOT,NCHR
        DO 693 I=1,NEW
         READ(8,*)
693     CONTINUE
C-- READ IN FIRST BREAK DATA ------
        IF (IDUP.EQ.1) THEN
         DO 694 I = 1,NCHR,1
          READ (8,695) NFIRST(I)
          NFIRST(I)=NFIRST(I)-ISHYN*INT(9.521)
694      CONTINUE
        ELSE
         DO 688 I = NCHR,1,-1
          READ (8,695) NFIRST(I)
          NFIRST(I)=NFIRST(I)-ISHYN*9
688      CONTINUE
        ENDIF
695     FORMAT(59X,I3)
        CLOSE(UNIT=8)
       ELSE IF (IREPLY .EQ. 3) THEN
        IF (IBAT.NE.1)
     1   WRITE (6,*)'No of samples to shift all channels (+/-)?'
        READ(5,*) IMOVE
        DO 696 I = 1, NCHR
         NFIRST(I) = NFIRST(I)+IMOVE
696     CONTINUE
       ELSE IF (IREPLY .EQ. 4) THEN
        IF (IBAT.NE.1) WRITE (6,*)'Pre/Post-flood (t/f)?'
        READ(5,*) ANS
        NEW = 0
        IF (ANS.EQ.'t'.OR.ANS.EQ.'T') OPEN(UNIT=8,FILE='fortt.3')
        IF (ANS.EQ.'f'.OR.ANS.EQ.'F') OPEN(UNIT=8,FILE='fortf.3')
```

```
      IF (ICSCRG.EQ.1) THEN
       NEW = NSHOT - 1
      ELSE
       NEW = NCHR*(NSHOT-1)
      ENDIF
      WRITE (6,*) NEW,NSHOT,NCHR
      DO 697 I=1,NEW
       READ(8,*)
697   CONTINUE
C-- READ IN FIRST BREAK DATA ------
       DO 698 I = 1,NCHR
        READ (8,*) NTEM1,NTEM2,NTEM3
        IF (NTEM2.NE.NFIRST(I)) THEN
         PRINT*,'uh oh, non matching fbs'
        ENDIF
        NFIRST(I) = NTEM3
        IF (ICSCRG.EQ.1) THEN
         DO 699 J=1,NCHR-1
          READ(8,*)
699      CONTINUE
        ENDIF
698    CONTINUE
       CLOSE(UNIT=8)
      END IF
     ELSE IF (IOPT .EQ. 11) THEN
700   IF (IBAT.NE.1) THEN
       WRITE (6,*) (NFIRST(I),I=1,NCHR)
       WRITE (6,*)'APPLY SAME MUTE TO ALL TRACES (1=Y) :'
      ENDIF
      READ (5,*) IOANS
      IF (IOANS .EQ. 1) THEN
       IF (IBAT.NE.1) WRITE (6,*)'Enter length of cos tap in sams :'
       READ (5,*) NTAP10
       IF (NTAP10 .EQ. 0) GO TO 730
       IF (IBAT.NE.1) WRITE (6,*)'Enter start of cos tap (0=FB):'
       READ (5,*) NTAP00
       DO 730 J = 1, NCHR
        NTAP0 = NFIRST(J) + NTAP00
        NTAP1 = NTAP10 + NTAP0
        IF (NTAP1 - 1 .GT. NSAM .OR. NTAP0 .LT. 1) THEN
         IF (IBAT.NE.1) THEN
          WRITE (6,*)'CHANNEL ', J
          WRITE (6,*)'ERROR NTAP0,NTAP1 = ',NTAP0,NTAP1
         ENDIF
         GO TO 700
        END IF
        DO 710 I = NTAP0 + 1, NTAP1 - 1
         FAC = REAL(I - NTAP0) / REAL(NTAP1 - NTAP0)
         FAC = FAC * 3.1415926535
         FAC = 0.5 * (1.0 - COS(FAC))
         R4DAT(I,J) = R4DAT(I,J) * FAC
710     CONTINUE
        DO 720 I = 1, NTAP0
         R4DAT(I,J) = 0.0
720     CONTINUE
730    CONTINUE
      ELSE
740    IF (IBAT.NE.1)
     1 WRITE (6,*)'Enter length of cosine taper (0=NO mute):'
       READ (5,*) NTAP10
       IF (NTAP10 .EQ. 0) GO TO 781
       IF (IBAT.NE.1)
     1 WRITE (6,*)'Enter start of cosine taper (0=1st break):'
       READ (5,*) NTAP00
       IF (IBAT.NE.1)
     1 WRITE (6,*)'1ST AND LAST TRACES TO APPLY MUTE'
       READ (5,*) NGO0, NGO1
       DO 780 J = NGO0, NGO1
        NTAP0 = NFIRST(J) + NTAP00
        NTAP1 = NTAP10 + NTAP0
        IF (NTAP1 - 1 .GT. NSAM .OR. NTAP0 .LT. 1) THEN
         IF(IBAT.NE.1)WRITE(6,*)'ERROR NTAP0,NTAP1=',
     1   NTAP0,NTAP1
         GO TO 780
750     END IF
        DO 760 I = NTAP0 + 1, NTAP1 - 1
         FAC = REAL(I - NTAP0) / REAL(NTAP1 - NTAP0)
         FAC = FAC * 3.14159265
         FAC = 0.5 * (1.0 - COS(FAC))
         R4DAT(I,J) = R4DAT(I,J) * FAC
760     CONTINUE
        DO 770 I = 1, NTAP0
         R4DAT(I,J) = 0.0
770     CONTINUE
780    CONTINUE
781   END IF
     ELSE IF (IOPT .EQ. 12) THEN
      CALL ZERO(NSAMR,TEMP)
C     Shifts data by first break pick ... Aligns data at sample 50
      IF (IBAT.NE.1) PRINT*, 'What shift? (+/-1 +ve/-ve shift)'
      READ(5,*) IS
      CALL ALIGND(NSAM,NCHR,NSAMR,NCHRR,R4DAT,
     1 NFIRST,TEMP,IS)
      CALL ZERO(NSAMR,TEMP)
     ELSE IF (IOPT .EQ. 13) THEN
      IF (IBAT.NE.1) PRINT*,'WHICH CHANNELS(FIRST,LAST)'
      READ(5,*) ICH1,ICH2
      ICH=ICH2-ICH1+1
      CALL ZERO(NSAMR,TEMP)
      DO 790 J = ICH1, ICH2
       DO 800 I = 1, NSAM
        TEMP(I)=TEMP(I)+R4DAT(I,J)
800    CONTINUE
790   CONTINUE
      DO 810 I = 1, NSAM
       R4DAT(I,1)=TEMP(I)/ICH
810   CONTINUE
      CALL ZERO(NSAMR,TEMP)
     ELSE IF (IOPT .EQ. 14) THEN
      IF (IBAT.NE.1) WRITE (6,*) 'Which trace FB for realignment :'
      IF (IBAT.NE.1) WRITE (6,*) '(99/98 for any/several trace)'
      READ (5,*) NFB
      IS = -1
      IF (NFB.EQ.99.OR.NFB.EQ.98) THEN
       IF (NFB.EQ.98) THEN
        IF (IBAT.NE.1) WRITE (6,*) 'Range of traces to realign :'
        READ (5,*) NALIGN1,NALIGN2
       ELSE
        IF (IBAT.NE.1) WRITE (6,*) 'Which trace to realign :'
        READ (5,*) NALIGN1
        NALIGN2 = NALIGN1
       ENDIF
       DO 813 J=NALIGN1,NALIGN2
        DO 811 I = 1,NSAMR
         TEMP(I) = R4DAT(I,J)
811     CONTINUE
        IDUM1(1) = NFIRST(J)
        CALL ALIGND(NSAM,1,NSAMR,NCHRR,TEMP,
     1  IDUM1,A,IS)
        DO 812 I = 1,NSAMR
         R4DAT(I,J) = TEMP(I)
812     CONTINUE
813    CONTINUE
      ELSE
       NFIRST(1) = NFIRST(NFB)
       CALL ALIGND(NSAM,1,NSAMR,NCHRR,R4DAT,NFIRST,
     1 TEMP,IS)
      ENDIF
C-------------------------------------------------------------
C  NICKED FROM ED'S PROC PROGRAM
C-------------------------------------------------------------
     ELSE IF (IOPT.EQ.15) THEN
      IF (IBAT.NE.1) PRINT*,'sine wave (S) or enter values (V)?'
      READ(5,*) ANSIN
```

```
C    Enter data sample values...
154    IF (IBAT.NE.1) PRINT*,'Enter trace to create'
      READ*,IT
      IF (IBAT.NE.1) PRINT*,'Enter first and last samples to input'
      IF (IBAT.NE.1) PRINT*,'Other samples are left unchanged'
      IF (IBAT.NE.1) PRINT*,'For sin put a spike at the sample value'
      READ*,IFS,ILS
      IF (IBAT.NE.1) PRINT*,'Trace ',IT
      DO 322 IS = IFS,ILS
       IF (IBAT.NE.1) PRINT*,'Enter sample value ',IS,' trace ',IT
       READ*,DATAP
       R4DAT(IS,IT) = DATAP
322    CONTINUE
      IF (IBAT.NE.1) PRINT*,' '
      IF (IBAT.NE.1)  PRINT*,'Another trace ?  1=y'
      READ*,IMORE
      IF(IMORE.EQ.1) GO TO 154
c
      IF (ANSIN.EQ.'S'.OR.ANSIN.EQ.'s') THEN
       CALL ZERO(NSAMR,TEMP)
       PI = ACOS(-1.)
       DTS = DT/1000000.
       IF (IBAT.NE.1) PRINT*,' Enter frequency of sin wave '
       READ*, RFR
       T = 1./RFR
       NS = INT(T/DTS)
       PRINT*,' dts  = ',DTS
       PRINT*,' RFR  = ',RFR
       PRINT*,' T    = ',T
       PRINT*,' ns   = ',NS
       DO 2932 I = 1,NS+1
        TEMP(I) = SIN(2.*PI*RFR*FLOAT(I-1)*DTS)
2932    CONTINUE
       DO 2933 J = 1,NCHR
        DO 2934 I = 1,NSAM
            XTEMP2(I) = R4DAT(I,J)
2934     CONTINUE
        CALL FOLD(NSAM-NS,XTEMP2,NS+1,TEMP,NSAM,
   1    XTEMP3)
        DO 2935 I = 1,NSAM
            R4DAT(I,J) = XTEMP3(I)
2935     CONTINUE
2933    CONTINUE
       ENDIF
       GO TO 10
C----------------------------------------------------------
      ELSE IF (IOPT .EQ. 16) THEN
       IF (IBAT.NE.1) WRITE (6,*)'For tomog-lows(1),-tank(2)
     1,runfile(3) :'
       READ (5,*) ITOM
       IF (IBAT.NE.1) WRITE (6,*)'Enter name of file to write to :'
       READ (5,822) FNAME
       IF(IBAT.NE.1)WRITE(6,*)'Static correction sams (-means longer):'
       READ (5,*) RCOREC
822    FORMAT (A10)
       OPEN (4,FILE=FNAME,STATUS='UNKNOWN')
       IF (ITOM.EQ.3) THEN
        DO 820 I = 1, NCHR
         WRITE(4,*) '1'
         WRITE(4,821) NFIRST(I)
821       FORMAT(I3)
820     CONTINUE
       ELSE
        IF (ITOM.EQ.1) THEN
         X1 = 0.0
         X2 = 37.1
         Z1 = SORPOS + 0.28
        ELSE
         OPEN(3,FILE='tankposn',STATUS='OLD')
         DO 819 JJJ = 1,51
          READ(3,*) XSO(JJJ),ZSO(JJJ)
819       CONTINUE
         DO 818 JJJ = 1,51
          READ(3,*) XRE(JJJ),ZRE(JJJ)
818       CONTINUE
        ENDIF
        DO 815 KKK = 1,NCHR*(NSHOT-1)
         READ(4,*)
815     CONTINUE
        DO 817 I = 1, NCHR
         IF(ITOM.EQ.2) THEN
          IF(ICSCRG.EQ.1) THEN
           X1 = XSO(NSHOT)
           X2 = XRE(I)
           Z1 = ZSO(NSHOT)
           RECDEP(I) = ZRE(I)
          ELSE
           X1 = XRE(NSHOT)
           X2 = XSO(I)
           Z1 = ZRE(NSHOT)
           RECDEP(I) = ZSO(I)
          ENDIF
         ENDIF
         TT = DT/1E6*(NFIRST(I)-RCOREC-1)
         NTEM(I) = NFIRST(I)- INT(RCOREC)
         WRITE(4,816) X1,Z1,X2,RECDEP(I),TT,NTEM(I)
816      FORMAT(F8.4,2X,F8.4,2X,F8.4,2X,F8.4,2X,F13.9,2X,I7)
817     CONTINUE
        IF(ITOM.EQ.2) CLOSE(3)
       ENDIF
       CLOSE (4)
       CALL ZERO(NCHRR,NTEM)
      ELSE IF (IOPT .EQ. 17) THEN
       IF (IBAT.NE.1) WRITE(6,*) 'Enter trace no:'
       READ(5,*) NTRWIN
       IF (IBAT.NE.1) THEN
        WRITE(6,*) 'Enter first and last sample for window:'
        WRITE(6,*) ' (999,100=FB,FB+100):'
       ENDIF
       READ(5,*) NF, NL
       IF(NF .EQ. 999) THEN
        NFWIN = NFIRST(NTRWIN)
        NLWIN = NFIRST(NTRWIN) + NL
       ELSE
        NFWIN = NF
        NLWIN = NL
       ENDIF
       NWIN = NLWIN - NFWIN + 1
       DO 823 I = NFWIN, NLWIN
        TEMP(I-NFWIN+1) = R4DAT(I,NTRWIN)
823    CONTINUE
       CALL RMSERR(NWIN, TEMP, ENTRAC)
       IF (IBAT.NE.1) PRINT *, ENTRAC
      ELSE IF (IOPT .EQ. 18) THEN
       IF (IBAT.NE.1) THEN
       WRITE(6,*) 'First,last sample for window (0,0=whole trace):'
       WRITE(6,*) ' (999,100=FB,FB+100):'
       ENDIF
       READ(5,*) NF, NL
       DO 824 J = 1, NCHR
        IF(NF .EQ. 0 .AND. NL .EQ. O) THEN
         NFWIN = 1
         NLWIN = NSAM
        ELSE IF(NF .EQ. 999) THEN
         NFWIN = NFIRST(J)
         NLWIN = NFIRST(J) + NL
        ELSE
         NFWIN = NF
         NLWIN = NL
        ENDIF
        NWIN = NLWIN - NFWIN + 1
        CALL ZERO (2,RSCAL)
        DO 825 I = NFWIN, NLWIN
         TEMP(I-NFWIN+1) = R4DAT(I,J)
```

```
825   CONTINUE
      CALL RMSERR(NWIN, TEMP, RSCAL(1))
      DO 826 I = NFWIN, NLWIN
       TEMP(I-NFWIN+1) = R2DAT(I,J)
826   CONTINUE
      CALL RMSERR(NWIN, TEMP, RSCAL(2))
      CALL MAXSN(2, RSCAL, RMSMAX, MAXTRC)
      IF (RSCAL(1) .NE. 0.0) THEN
       SCAL1 = RMSMAX / RSCAL(1)
      ELSE
       SCAL1 = 0.0
      ENDIF
      IF (RSCAL(2) .NE. 0.0) THEN
       SCAL2 = RMSMAX / RSCAL(2)
      ELSE
       SCAL2 = 0.0
      ENDIF
      IF (IBAT.NE.1) PRINT *, 'SCALE 1ST ID', J, ' = ', SCAL1
      IF (IBAT.NE.1) PRINT *, 'SCALE 2ND ID', J, ' = ', SCAL2
      DO 827 I = 1, NSAM
       R4DAT(I,J) = R4DAT(I,J) * SCAL1
       R2DAT(I,J) = R2DAT(I,J) * SCAL2
827   CONTINUE
824   CONTINUE
     ELSE IF (IOPT .EQ. 19) THEN
      DO 828 J = 1, NCHR
       CALL MAXAMP(NSAM, R4DAT(1,J), RSCAL(1), INDAMP)
       CALL MAXAMP(NSAM, R2DAT(1,J), RSCAL(2), INDAMP)
       CALL MAXSN(2, RSCAL, BIGAMP, II)
       IF (RSCAL(1) .NE. 0.0) THEN
        SCAL1 = BIGAMP / RSCAL(1)
       ELSE
        SCAL1 = 0.0
       ENDIF
       IF (RSCAL(2) .NE. 0.0) THEN
        SCAL2 = BIGAMP / RSCAL(2)
       ELSE
        SCAL2 = 0.0
       ENDIF
       IF (IBAT.NE.1) PRINT *, 'SCALE 1ST ID', J, ' = ', SCAL1
       IF (IBAT.NE.1) PRINT *, 'SCALE 2ND ID', J, ' = ', SCAL2
       DO 829 I = 1, NSAM
        R4DAT(I,J) = R4DAT(I,J) * SCAL1
        R2DAT(I,J) = R4DAT(I,J) * SCAL2
829   CONTINUE
828   CONTINUE
     ELSE IF (IOPT .EQ. 20) THEN
      IF (IBAT.NE.1) WRITE (6,*)'Stack (1 or -1 == REV POL ) :'
      READ (5,*) ISTACK
      DO 830 J = 1, NCHR
       DO 831 I = 1, NSAM
        IF (ISTACK .GE. 0) THEN
         R4DAT(I,J) = R4DAT(I,J) + R2DAT(I,J)
        ELSE
         R4DAT(I,J) = R4DAT(I,J) - R2DAT(I,J)
        END IF
831    CONTINUE
830   CONTINUE
     ELSE IF (IOPT .EQ. 21) THEN
      IF (IBAT.NE.1) WRITE (6,*)' Enter file number 1 or 2:'
      READ (5,*) I1OR2
      IF (IBAT.NE.1) WRITE (6,*)' Enter trace number :'
      READ (5,*) NTRACE
      IF (IBAT.NE.1) THEN
       WRITE (6,*)' ENTER window on trace in samples :'
       WRITE (6,*)' 0,0 = FB->NSAMR 0,-1 = FB->FB+NSAMR/2:'
      ENDIF
      READ (5,*) NWT1, NWT2
      IF (NWT1.EQ.0.AND.NWT2.EQ.0) THEN
       NWT1 = NFIRST(NTRACE)
       NWT2 = NSAMR
      ENDIF
      IF (NWT1.EQ.0.AND.NWT2.EQ.-1) THEN
       NWT1 = NFIRST(NTRACE)
       NWT2 = NWT1+NSAMR/2
      ENDIF
      NWTDIF = NWT2 - NWT1 + 1
      PRINT*,NTRACE,NWT1,NWT2,I1OR2
      CALL ZERO(NS2, A)
      IF (I1OR2.EQ.1) THEN
       DO 840 I = 1, NWTDIF
        A(I) = R4DAT(I + NWT1 - 1,NTRACE)
840    CONTINUE
      ELSE
       DO 841 I = 1, NWTDIF
        A(I) = R2DAT(I + NWT1 - 1,NTRACE)
841    CONTINUE
      ENDIF
      CALL NORMAN(NSAMR, A)
     ELSE IF (IOPT .EQ. 22) THEN
      CALL PRO16A (NSAM,NCHR,NSAMR,NCHRR,NWTDIF,
     1 R4DAT,A,NFIRST,IBAT)
     ELSE IF (IOPT .EQ. 23) THEN
      DO 842 J = 1, NCHR
       CALL MAXAMP(NSAM, R4DAT(1,J), TAMP, INDAMP)
       PRINT*,'1'
       PRINT*,INDAMP
842   CONTINUE
     ELSE IF (IOPT .EQ. 24) THEN
      PRINT*,'same scaling factor for whole gather'
      PRINT*,'What is scaling factor?'
      READ(5,*) SCAFAC
      DO 843 J = 1, NCHR
       DO 844 I = 1, NSAM
        R4DAT(I,J) = R4DAT(I,J) * SCAFAC
844    CONTINUE
843   CONTINUE
     ELSE IF (IOPT .EQ. 25) THEN
      IF(IBAT.NE.1) PRINT*,'Source Depth is', SORPOS,'m'
      IF(IBAT.NE.1) PRINT*,'Rec Depths are', (RECDEP(I),I=1,NCHR)
      IF(IBAT.NE.1) PRINT*,'What is the borehole separation (m)?'
      IF(IBAT.NE.1) PRINT*,'0 = RETURN'
      READ(5,*) BORSEP
      IF (BORSEP.EQ.0.) GO TO 10
      DO 845 J = 1, NCHR
       SRSEP = SQRT(BORSEP**2+(SORPOS-RECDEP(J))**2)
       IF(IBAT.NE.1) PRINT*,SRSEP/BORSEP
       DO 846 I = 1, NSAM
        R4DAT(I,J) = R4DAT(I,J) * SRSEP/BORSEP
846    CONTINUE
845   CONTINUE
     ELSE IF (IOPT .EQ. 0) THEN
      RETURN
     END IF
     GO TO 10
     END

c------------------------------------------------------------------------
      integer function lnb(string)
c-t determines last non blank character in a string
c-a dave stevenson 1986
c-l fortran77
c-d+
c  scans the string starting at the last column and checks each
c  character in turn to see if it is a space, if not it returns
c  the character position number, if all string is blank then
c  zero is returned.
c  routines called:-
c    len
c-d-
      character*(*) string
      l=len(string)
      lnb=0
      do 100 i=l,1,-1
```

```
      if(string(i:i).ne.' ') then
        lnb=i
        goto 10
      else
      endif
100 continue
10  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 PIKSRT(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
      RETURN
      END

c---------------------------------------------------------------------
C  Subroutine to median filter seismic data
      SUBROUTINE MEDFIL(NSAM, NCHR, NSAMR, NCHRR,
     1 R4DAT, NFIRST, LENFIL, DGDAT)
c X array size > NCHR + twice the med fil overlap
C     eg 51 +3 +3 for LENFIL=7
C  Shift the traces :   FIRST BREAKS AT NALIGN SAMPLES
      DIMENSION R4DAT(NSAMR,NCHRR), NFIRST(NCHRR)
      REAL X(60), Y(60), DGDAT(NSAMR,NCHRR)
      CHARACTER*1 ANS
 10 FORMAT (A1)
      PRINT*,NSAM, NCHR, NSAMR, NCHRR, LENFIL
      LENDU = NSAMR * NCHRR
c     CALL ZERO(LENDU, UGDAT(1,1))
      CALL ZERO(LENDU, DGDAT(1,1))
      DO 50 J = 1, NCHR
        DO 40 I = 1, NSAMR
          DGDAT(I,J) = R4DAT(I,J)
 40   CONTINUE
 50 CONTINUE
      IF (NFIRST(1) .EQ. 0) THEN
        PRINT *, 'INPUT FIRST BREAK SAMPLES :'
        DO 51 J = 1, NCHR
          READ (5,*) NFIRST(J)
 51   CONTINUE
      END IF
      PRINT *, 'ALIGNING TRACES...'
      NALIGN = 50
      DO 80 J = 1, NCHR
        NDT = NFIRST(J) - NALIGN
        DO 60 I = 1, NSAMR - NDT
          DGDAT(I,J) = DGDAT(I + NDT,J)
 60     CONTINUE
        DO 70 I = NSAMR - NDT + 1, NSAMR
          DGDAT(I,J) = 0.0
 70     CONTINUE
 80 CONTINUE
      LN = NALIGN - 5
C  MUTE  before first arrivals
      DO 90 J = 1, NCHR
        CALL ZERO(LN, DGDAT(1,J))
 90 CONTINUE
      PRINT *, 'TRACES ALIGNED '
C  Apply median filter across traces
100 PRINT *, 'APPLYING MEDIAN FILTER ....'
      DO 140 I = LN + 1, NSAMR
        DO 110 J = 1 - LENFIL / 2, NCHR + LENFIL / 2
C  Pad with zeros
          IF (J .LT. 1 .OR. J .GT. NCHR) THEN
            X(J + LENFIL/2) = 0.0
          ELSE
            X(J + LENFIL/2) = DGDAT(I,J)
          END IF
 110    CONTINUE
        DO 130 J = 1, NCHR
C this is needed cos the mdian1 subrout sorts the array
          DO 120 JJ = 1, NCHR+LENFIL
            Y(JJ) = X(JJ)
 120      CONTINUE
          CALL MDIAN1(Y(J), LENFIL, DGDAT(I,J))
 130    CONTINUE
 140 CONTINUE
      PRINT *, 'TRACES STILL ALIGNED '
      PRINT *, 'RE-APPLY MEDIAN FILTER (Y/N)?'
      READ (5,10) ANS
      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') GO TO 100
C  Shift back the traces
      PRINT *, 'SHIFTING BACK TRACES ...'
      DO 170 J = 1, NCHR
        NDT = NFIRST(J) - NALIGN
        DO 150 I = NSAMR, NALIGN, -1
          DGDAT(I,J) = DGDAT(I - NDT,J)
 150    CONTINUE
        DO 160 I = 1, NALIGN - 1
          DGDAT(I,J) = 0.0
 160    CONTINUE
 170 CONTINUE
      PRINT *, 'TRACES SHIFTED BACK '
      PRINT*,'COPY WAVEFIELD TO R4DAT (Up(ref)/Down(dir)/No)?'
      READ (5,10) ANS
      IF (ANS .EQ. 'U' .OR. ANS .EQ. 'u') THEN
        DO 210 J = 1, NCHR
          DO 200 I = 1, NSAMR
            R4DAT(I,J) = R4DAT(I,J) - DGDAT(I,J)
 200      CONTINUE
 210    CONTINUE
      ELSE IF (ANS .EQ. 'D' .OR. ANS .EQ. 'd') THEN
        DO 230 J = 1, NCHR
          DO 220 I = 1, NSAMR
            R4DAT(I,J) = DGDAT(I,J)
 220      CONTINUE
 230    CONTINUE
      END IF
      RETURN
      END

c---------------------------------------------------------------------
      SUBROUTINE MENPL2 (NSAM, NCHR, NSAMR, NCHRR,
     1 R4DAT, NFIRST, TITOFF, CHNSPL, POLART, VARARE,
     2 NPLFST, NPLLST, IOPT, SORPOS, RECDEP, NCR, DT,
     3 LOCATE, DATE, DEVICE, SORTYP, SORLOC, RECTYP,
     4 RECLOC, COMSHT, FILREC, NSHOT, NPROCS,
     5 IDPROC, TITLE, XLB, YLB, SCAL, AMSPEC, PROC)
C  plan to pass three dummy arrays for  B, RDAT all NSAMS,NCHR
C  Other arrays set to 128 i.e. max no of channels likely
C  Subroutine for graphics output of array R4DAT(NSAMR,NCHRR)
C  NB This is MENPL3_U in Mike Findlay's MTS archive
      PARAMETER(NS1=2048,NS2=128)
      CHARACTER*20 LOCATE,DATE,DEVICE,SORTYP,SORLOC
      CHARACTER*20 RECTYP,RECLOC,COMSHT,FILREC
      CHARACTER CHNSPL*26,POLART*7,COMENT*40,ANS*1
      CHARACTER*25 PROC(20),AQW*1
      CHARACTER*30 TITLE,XLB,YLB,FNAME
      CHARACTER*3 VARARE,TITOFF,FLAGBR,COLOUR,RMSYN
      CHARACTER*3 AGCYN, RAMYN,AMSPEC,SPYES,TRAMP
      REAL RECDEP(NCHRR),TEMP(NS2),RECD1(NS2)
      REAL B(NS1,NS2),RDAT(NS1,NS2),R4DAT(NSAMR,NCHRR)
      INTEGER NFIRST(NCHRR),NCR(2),IDPROC(5,NCHRR)
```

```
      INTEGER NCHPLT(NS2),NF1(NS2)
      AGCYN = 'NO'
      RMSYN = 'NO'
      FLAGBR = 'OFF'
      COLOUR = 'ON'
      COMENT = ' '
      SPYES = 'NO'
      TRAMP = 'NO'
  991 FORMAT(A1)
C
C  SET UP DUMMY PLOT ARRAY RDAT(NSAMR,NCHRR)
C
      DO 111 J = 1, NCHR
        NF1(J)=NFIRST(J)
        RECD1(J)=RECDEP(J)
  111 CONTINUE
      IF (CHNSPL .EQ. 'ALL') THEN
       NCHPL = NCHR
       DO 370 III = 1, NCHR
        NCHPLT(III) = III
  370  CONTINUE
      ELSE IF (CHNSPL .EQ. 'ODD') THEN
       NCHPL = 0
       DO 371 III = 1, NCHR,2
        NCHPL = NCHPL + 1
        NCHPLT(NCHPL) = III
        NF1(NCHPL) = NFIRST(III)
        RECD1(NCHPL) = RECDEP(III)
  371  CONTINUE
      ELSE IF (CHNSPL .EQ. 'EVEN') THEN
       NCHPL = 0
       DO 372 III = 2, NCHR, 2
        NCHPL = NCHPL + 1
        NCHPLT(NCHPL) = III
        NF1(NCHPL) = NFIRST(III)
        RECD1(NCHPL) = RECDEP(III)
  372  CONTINUE
      ELSE IF (CHNSPL .EQ. 'SOME') THEN
       DO 373  III = 1, NCHPL
        WRITE(6,*) 'Enter i.d. of channel #',III
        READ(5,*) NCHPLT(III)
        NF1(III) = NFIRST(NCHPLT(III))
        RECD1(III) = RECDEP(NCHPLT(III))
  373  CONTINUE
      ELSE IF (CHNSPL .EQ. 'SEQUENTIAL') THEN
       DO 374  III = NCHF, NCHL
        NF1(III-NCHF+1) = NFIRST(III)
        RECD1(III-NCHF+1) = RECDEP(III)
  374  CONTINUE
      ENDIF
C
C  MENU FOR PLOT OPTIONS
C
    2 DO 1001 J = 1, NCHR
        DO 1001 I = 1, NSAM
             RDAT(I,J) = R4DAT(I,J)
 1001 CONTINUE

      WRITE(6,*)
      WRITE(6,*) '           Plot Options Menu          '
      WRITE(6,*) '           ~~~~ ~~~~~~~ ~~~~          '
      WRITE(6,*)
      WRITE(6,10) TITOFF
   10 FORMAT(/'   1  Titles off or on           :',A3)
      WRITE(6,20) NPLFST,NPLLST
   20 FORMAT('   2  First and last samples     :',I3,1X,I4)
      WRITE(6,30) CHNSPL
   30 FORMAT('   3  Channels to be plotted     :',A26)
      IF(CHNSPL .EQ. 'ALL') THEN
       NCHPL = NCHR
       DO 31  III = 1, NCHR
        NCHPLT(III) = III
   31  CONTINUE
      ENDIF
      WRITE(6,40) POLART
   40 FORMAT('   4  Normal/Reverse Polarity    :',A7)
      WRITE(6,50) VARARE
   50 FORMAT('   5  Variable Area              :',A3)
      WRITE(6,51) COLOUR
   51 FORMAT('  15 Variable Area Colour        :',A3)
      WRITE(6,60)
   60 FORMAT(/'   6  Plot Seismograms             ')
      WRITE(6,70) FLAGBR
   70 FORMAT(/'   7  First break flags          :',A3)
      WRITE(6,80)
   80 FORMAT('   8  Include comment              ')
      WRITE(6,90)
   90 FORMAT('   9  PRINT out window of trace   ')
      WRITE(6,95) AGCYN, RAMYN
   95 FORMAT('  10  AGC / Time ramp :',A3,X,A3)
      WRITE(6,97) SCAL
   97 FORMAT('  11  Vertical plot scaler      :',F6.2)
      WRITE(6,98) RMSYN
   98 FORMAT('  12  Apply RMS normalisation   :',A3)
      WRITE(6,981)
  981 FORMAT('  13  Interpolate by power of 2 ')
      WRITE(6,982) AMSPEC
  982 FORMAT('  14  Plot amp spectrums        :',A3)
      IF (FLAGBR.EQ.'ON') WRITE(6,983) SPYES
  983 FORMAT('  16  Add spikes at 1 sam to stop FB crash :',A3)
      WRITE(6,984) TRAMP
  984 FORMAT('  17  Plot true amplitudes      :',A3)
      WRITE(6,985)
  985 FORMAT('  18  Write out FBs and data around ')
      WRITE(6,99)
   99 FORMAT(/'   0  Return to MAIN MENU')
      READ(5,*) IANSWR
      IF (IANSWR .EQ. 1) THEN
        IF (TITOFF .EQ. 'OFF') THEN
              TITOFF = 'ON'
        ELSE
              TITOFF = 'OFF'
        ENDIF
      ENDIF
      IF (IANSWR .EQ. 2) THEN
        WRITE(6,*) '1st and last samples to be plotted :'
        READ(5,*) NPLFST,NPLLST
      ENDIF
      IF (IANSWR .EQ. 3) THEN
        WRITE(6,*) '        SELECT OPTION          '
        WRITE(6,*) ' 1    All channels          '
        WRITE(6,*) ' 2    Odd channels          '
        WRITE(6,*) ' 3    Even channels         '
        WRITE(6,*) ' 4    Some channels         '
        WRITE(6,*) ' 5    Sequential channels   '
        READ(5,*) IOPT
        IF (IOPT .EQ. 1) THEN
              CHNSPL = 'ALL'
              NCHPL = NCHR
              DO 300 III = 1, NCHR
                     NCHPLT(III) = III
  300         CONTINUE
        ELSE IF (IOPT .EQ. 2) THEN
              CHNSPL = 'ODD'
              NCHPL = 0
              DO 301 III = 1, NCHR,2
                     NCHPL = NCHPL + 1
                     NCHPLT(NCHPL) = III
                     NF1(NCHPL) = NFIRST(III)
                     RECD1(NCHPL) = RECDEP(III)
  301         CONTINUE
        ELSE IF (IOPT .EQ. 3) THEN
              CHNSPL = 'EVEN'
              NCHPL = 0
```

```
      DO 302 III = 2, NCHR, 2
            NCHPL = NCHPL + 1
            NCHPLT(NCHPL) = III
            NF1(NCHPL) = NFIRST(III)
            RECD1(NCHPL) = RECDEP(III)
302       CONTINUE
   ELSE IF (IOPT .EQ. 4) THEN
      CHNSPL = 'SOME'
      WRITE(6,*) 'Enter number of channels to be plotted :'
      READ(5,*) NCHPL
      DO 303  III = 1, NCHPL
            WRITE(6,*) 'Enter i.d. of channel #',III
            READ(5,*) NCHPLT(III)
            NF1(III) = NFIRST(NCHPLT(III))
            RECD1(III) = RECDEP(NCHPLT(III))
303       CONTINUE
   ELSE IF (IOPT .EQ. 5) THEN
      CHNSPL = 'SEQUENTIAL'
      WRITE(6,*) 'Enter first,last channel to be plotted :'
      READ(5,*) NCHF,NCHL
      NCHPL = NCHL - NCHF +1
      DO 304  III = NCHF, NCHL
            NCHPLT(III-NCHF+1) = III
            NF1(III-NCHF+1) = NFIRST(III)
            RECD1(III-NCHF+1) = RECDEP(III)
304       CONTINUE
   ENDIF
  ENDIF
  IF (IANSWR .EQ. 4) THEN
   IF (POLART .EQ. 'REVERSE') THEN
      POLART = 'NORMAL'
   ELSE
      POLART = 'REVERSE'
   ENDIF
  ENDIF
  IF (IANSWR .EQ. 5) THEN
   IF (VARARE .EQ. 'OFF') THEN
      VARARE = 'ON'
   ELSE
      VARARE = 'OFF'
   ENDIF
  ENDIF
  IF (IANSWR .EQ. 15) THEN
   IF (COLOUR .EQ. 'OFF') THEN
      COLOUR = 'ON'
   ELSE
      COLOUR = 'OFF'
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 7) THEN
   IF(FLAGBR .EQ. 'OFF') THEN
      FLAGBR = 'ON'
   ELSE
      FLAGBR = 'OFF'
   ENDIF
  ELSE IF(IANSWR .EQ. 8) THEN
   PRINT*,'COMMENT :'
   READ(5,888) COMENT
888  FORMAT(A40)
  ENDIF
  IF(IANSWR .EQ. 9) THEN
   WRITE(6,*) ' No. of channel to be printed :'
   READ(5,*) IDCHAN
  WRITE(6,*) 'Range of sams to be printed: fb=',NFIRST(IDCHAN)
   READ(5,*) NSAM0,NSAM1
   DO 992 I=NSAM0, NSAM1
    WRITE(6,990) I,R4DAT(I,IDCHAN)
990    FORMAT(4(I4,1X,F12.4,2X))
992  CONTINUE
   WRITE(6,*) 'Select first break sample no. (currently ',
  1        NFIRST(IDCHAN),' ) :'
   READ(5,*) NF1(IDCHAN)
   NFIRST(IDCHAN) = NF1(IDCHAN)
  ENDIF
  IF(IANSWR .EQ. 10) THEN
   WRITE(6,*) 'Ramp or AGC     :'
   READ(5,991) ANS
   IF(ANS .EQ. 'R' .OR. ANS .EQ. 'r') THEN
    IF(RAMYN .EQ. 'NO') THEN
      RAMYN = 'YES'
    ELSE
      RAMYN = 'NO'
    ENDIF
   ELSE
    IF(AGCYN .EQ. 'NO') THEN
      AGCYN = 'YES'
    ELSE
      AGCYN = 'NO'
    ENDIF
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 11) THEN
   WRITE(6,*) 'Enter new plot scaling factor :'
   READ(5,*) SCAL
  ENDIF
  IF(IANSWR .EQ. 12) THEN
   IF(RMSYN .EQ. 'NO') THEN
      RMSYN = 'YES'
   ELSE
      RMSYN = 'NO'
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 13) THEN
   WRITE(6,*) 'Enter number of samples to transform :'
   READ(5,*) NSAMFT
   WRITE(6,*) 'Enter number of samples for back transform :'
   READ(5,*) NSAMIT
   DO 8000 JJ = 1, NCHR
    CALL FINT2(NSAMFT,NSAMIT,RDAT(1,JJ),RDAT(1,JJ),
  1      NS1,NS1)
8000  CONTINUE
   DT = DT * NSAMFT/NSAMIT
   DO 8100 JJ = 1, NCHR
      NF1(JJ) = ANINT(REAL(NF1(JJ) * NSAMIT/NSAMFT))
8100  CONTINUE
  ENDIF
  IF(IANSWR .EQ. 14) THEN
   IF(AMSPEC .EQ. 'NO') THEN
      AMSPEC = 'YES'
   ELSE
      AMSPEC = 'NO'
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 16) THEN
   IF(SPYES .EQ. 'NO') THEN
      SPYES = 'YES'
   ELSE
      SPYES = 'NO'
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 17) THEN
   IF(TRAMP .EQ. 'NO') THEN
      TRAMP = 'YES'
   ELSE
      TRAMP = 'NO'
   ENDIF
  ENDIF
  IF(IANSWR .EQ. 18) THEN
   PRINT*,'What filename?'
   READ(5,817) FNAME
817  FORMAT (A30)
   PRINT*,'Shift fbs to peak within +/- 3 sams (1=y)?'
   READ(5,*) ISHIFT
   PRINT*,'Writing to file ',FNAME
```

```
      OPEN (4,FILE=FNAME,STATUS='UNKNOWN')
       DO 819 I = 1, NCHR
        IN = NFIRST(I)
        RMAXFB = 1E-9
        AQW = '.'
        DO 816 II = IN-3,IN+3
         IF(R4DAT(II,I).GT.RMAXFB) THEN
          RMAXFB = R4DAT(II,I)
          IF (ISHIFT.EQ.1) NFIRST(I) = II
         ENDIF
816     CONTINUE
        IF (RMAXFB.NE.R4DAT(IN,I)) AQW = '*'
        WRITE(4,818) I,IN,(R4DAT(J,I),J=IN-3,IN+3),AQW
818     FORMAT(I2,X,I3,7(2X,F6.4),X,A1)
819    CONTINUE
      CLOSE (4)
     ENDIF
     IF (IANSWR .EQ. 6) THEN
      IF(RMSYN .EQ. 'NO') THEN
       IF(AGCYN .EQ. 'YES') THEN
        WRITE(6,*) 'Length of AGC to be applied :'
        READ(5,*) LENAGC
        DO 891 J = 1, NCHR
         CALL AGC(NSAM,1,RDAT(1,J),LENAGC)
891      CONTINUE
       ELSE
        IF(RAMYN .EQ. 'YES') THEN
         WRITE(6,*) 'T-Squared or Exponential (S or E) ?'
         READ(5,991)ANS
         IF(ANS .EQ. 'S' .OR. ANS .EQ. 's') THEN
          DO 880 J = 1, NCHR
           DO 879 I = 1, NSAM
            RDAT(I,J) = RDAT(I,J) * (I-1)**2
879         CONTINUE
880        CONTINUE
         ELSE
          WRITE(6,*) 'Enter Alpha (Exponential constant) :'
          READ(5,*) ALPHA
          DO 890 J = 1, NCHR
           DO 889 I = 1, NSAM
            TI = REAL(I - 1)
            TI = EXP(ALPHA*TI)
            RDAT(I,J) = RDAT(I,J) * TI * TI
889         CONTINUE
890        CONTINUE
         ENDIF
        ENDIF
       ENDIF
        PKMAX = 0.0
        DO 8010 JJJ = 1, NCHPL
         DO 8010 III = NPLFST,NPLLST
          IF(ABS(RDAT(III,JJJ)).GT.PKMAX)
     1     PKMAX=ABS(RDAT(III,JJJ))
8010    CONTINUE
        DO 8011 JJJ = 1, NCHPL
         DO 8011 III = NPLFST, NPLLST
          IF(TRAMP .EQ. 'NO') THEN
           RDAT(III,JJJ) = RDAT(III,JJJ) / PKMAX
           IF (POLART .EQ. 'REVERSE') THEN
            B(III,JJJ) = - RDAT(III,NCHPLT(JJJ))/REAL(NCHPL)
           ELSE
            B(III,JJJ) = RDAT(III,NCHPLT(JJJ))/REAL(NCHPL)
           ENDIF
          ELSE
           IF (POLART .EQ. 'REVERSE') THEN
            B(III,JJJ) = - RDAT(III,NCHPLT(JJJ))
           ELSE
            B(III,JJJ) = RDAT(III,NCHPLT(JJJ))
           ENDIF
          ENDIF
          IF(SPYES .EQ. 'YES') THEN
           B(NPLFST,JJJ) = 0.0001*PKMAX
          ENDIF
8011    CONTINUE
      ELSE
       CALL ZERO(NS2,TEMP)
       DO 801 JJJ = 1, NCHPL
        CALL RMSERR(NSAM,RDAT(1,NCHPLT(JJJ)),TEMP(JJJ))
801    CONTINUE
       CALL MAXSN(NCHPL,TEMP,RMSMAX,MAXTRC)
       DO 803 JJJ = 1, NCHPL
        IF(TEMP(JJJ) .EQ. 0.0) GO TO 803
        SCALE = 1.0 / TEMP(JJJ)
        DO 802 I = NPLFST,NPLLST
         RDAT(I,NCHPLT(JJJ)) = RDAT(I,NCHPLT(JJJ)) * SCALE
         IF(TRAMP .EQ. 'NO') THEN
          IF (POLART .EQ. 'REVERSE') THEN
           B(I,JJJ) = - RDAT(I,NCHPLT(JJJ))/REAL(NCHPL)
          ELSE
           B(I,JJJ) = RDAT(I,NCHPLT(JJJ))/REAL(NCHPL)
          ENDIF
         ELSE
          IF (POLART .EQ. 'REVERSE') THEN
           B(I,JJJ) = - RDAT(I,NCHPLT(JJJ))/REAL(NCHPL)
          ELSE
           B(I,JJJ) = RDAT(I,NCHPLT(JJJ))/REAL(NCHPL)
          ENDIF
         ENDIF
802     CONTINUE
        IF(SPYES .EQ. 'YES') THEN
         B(NPLFST,JJJ) = 0.0001*RMSMAX
        ENDIF
803    CONTINUE
      ENDIF
      DTPLOT = DT
      IF(AMSPEC .EQ. 'YES') THEN
       FNYQ = .5E6/DT
       DTPLOT = FNYQ/NSAMR*2000
       XLB = 'Frequency in hertz'
      ENDIF
C Call the UNIRAS plotting subroutine
      CALL TRACPL(NPLFST,NPLLST,NCHPL,NCHPLT,DTPLOT,
     1 VARARE,COLOUR,FLAGBR,TITLE,XLB,YLB,B,NF1, RECD1,
     2 SCAL,TITOFF,TRAMP)
      GO TO 2
     ENDIF
     IF (IANSWR .NE. 0) GO TO 2
     RETURN
     END

c----------------------------------------------------------------------
     SUBROUTINE MENUD(NSAM, NCHR, NSAMR, NCHRR,
    1 R4DAT, DT, NSHOT,AMSPEC, CSX, CSX1, TEMP2,
    2 P, Y, S, G, P1, P2)
C
C Subroutine to plot FFT of individual traces in current array
     DIMENSION R4DAT(NSAMR,NCHRR), G(NSAMR)
     DIMENSION P1(NSAMR),P2(NSAMR),TEMP2(NSAMR,4)
     DIMENSION P(NSAMR), Y(NSAMR), S(NSAMR),
     COMPLEX CSX(NSAMR), CSX1(NSAMR)
     CHARACTER*3 AMSPEC
     NTRACE = 1
     FNYQ = 0.5E6 / DT
     DF = FNYQ / REAL(NSAMR/2)
     R1 = 100*REAL(NSAMR/2)/FNYQ
     PRINT*, 'NSAM =',NSAM
  10 WRITE (6,*)
     CALL ZERO(NSAMR,G)
     CALL ZERO(NSAMR,Y)
     WRITE (6,*)'          Trace Spectral Plot Menu   '
     WRITE (6,*)'          ~~~~~ ~~~~~~~~ ~~~~ ~~~~   '
     WRITE (6,*)
     WRITE (6,20) NTRACE
  20 FORMAT (/'   1  Trace number (99 plots 2 traces):', 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 (6,81)
81 FORMAT (/'   8  Plot trace against Butterworth spectra ')
      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
        IF (NTRACE.EQ.99) THEN
          WRITE (6,*)'Enter trace numbers  :'
          READ (5,*) NTRAC1,NTRAC2
        END IF
      END IF
      IF (NTRACE .LT. 1 .OR. NTRACE .GT. NCHR) THEN
        IF (NTRACE.NE.99) GO TO 10
      ENDIF
      IF (IOPT .EQ. 2 .OR. IOPT .EQ. 3) THEN
        DO 100 I = 1, NSAM
          IF (NTRACE.EQ.99) THEN
            CSX(I) = CMPLX(R4DAT(I,NTRAC1),0.0)
            CSX1(I) = CMPLX(R4DAT(I,NTRAC2),0.0)
          ELSE
            CSX(I) = CMPLX(R4DAT(I,NTRACE),0.0)
          ENDIF
100   CONTINUE
        DO 110 I = NSAM + 1, NSAMR
          CSX(I) = 0.0
          IF (NTRACE.EQ.99) CSX1(I) = 0.0
110   CONTINUE
        CALL FFT(NSAMR, CSX, -1.0)
        CALL POLAR(NSAMR,CSX,P1,P2)
        CALL DRUM(NSAMR,P2)
        CALL NORMAN(NSAMR, P1)
        IF (NTRACE.EQ.99) THEN
          CALL FFT(NSAMR, CSX1, -1.0)
          CALL POLAR(NSAMR,CSX1,G,Y)
          CALL DRUM(NSAMR,Y)
          CALL NORMAN(NSAMR, G)
        ENDIF
        IF (IOPT .EQ. 2) THEN
          DO 600 I=1,NSAMR
            TEMP2(I,1)= P1(I)
            TEMP2(I,2)= P2(I)
            IF (NTRACE.EQ.99) THEN
              TEMP2(I,3)= G(I)
              TEMP2(I,4)= Y(I)
            ENDIF
600     CONTINUE
          IF (NTRACE.EQ.99) THEN
            CALL WVSHPL(NSAMR/2+1,4,NSAMR,TEMP2,0.0,100.0,R1,
     1'   spectra     ','frequency (Hz)')
          ELSE
            CALL WVSHPL(NSAMR/2+1,2,NSAMR,TEMP2,0.0,100.0,R1,
     1'   spectra     ','frequency (Hz)')
          ENDIF
        ELSE
          IF (NTRACE.EQ.99) THEN
            DO 601 I=1,NSAMR
              TEMP2(I,1)= P1(I)
              TEMP2(I,2)= G(I)
601       CONTINUE
            CALL WVSHPL(NSAMR/2+1,2,NSAMR,TEMP2,0.0,100.0,R1,
     1  'Amplitude spectrum','frequency (Hz)')
          ELSE
            CALL WVSHPL(NSAMR/2+1,1,NSAMR,P1,0.0,100.0,R1,
     1  'Amplitude spectrum','frequency (Hz)')
          ENDIF
        ENDIF
      END IF
      IF (IOPT .EQ. 4) THEN
        DO 120 I = 1, NSAM
          G(I) = R4DAT(I,NTRACE)
120   CONTINUE
        DO 130 I = NSAM + 1, NSAMR
          G(I) = 0.0
130   CONTINUE
        CALL POWERS(G, P, NSAMR, Y, S)
        FMIN = 0.0
        FMAX = 0.5E6 / DT
        CALL MINSN(NSAMR/2 + 1, P, PMIN, II)
        CALL MAXSN(NSAMR/2 + 1, P, PMAX, II)
        PRINT *, 'FMIN/MAX,PMIN/MAX', FMIN, FMAX, PMIN, PMAX
        DF = 2.0 * FMAX / REAL(NSAMR)
        DO 140 I = 1, NSAMR
          Y(I) = REAL(I - 1) * DF
140   CONTINUE
      END IF
      IF (IOPT .EQ. 5) THEN
        DO 180 JJ = 1, NCHR
          DO 150 II = 1, NSAM
            CSX(II) = CMPLX(R4DAT(II,JJ),0.0)
150     CONTINUE
          DO 160 II = NSAM + 1, NSAMR
            CSX(II) = 0.0
160     CONTINUE
          CALL FFT(NSAMR, CSX, -1.0)
          DO 170 II = 1, NSAM
            R4DAT(II,JJ) = CABS(CSX(II))
170     CONTINUE
180   CONTINUE
        AMSPEC = 'YES'
      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, NSAM, NLAGS, NORM, I2ZC)
          CALL ZERO(NSAM, R4DAT(1,JJ))
          DO 190 II = 1, NLAGS
            R4DAT(II,JJ) = S(II)
190     CONTINUE
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(NSAM, R4DAT(1,ID1), NSAM, R4DAT(1,ID2),
     1  LG, G)
        CALL MAXSN(LG, G, GMAX, II)
        CALL CROSS(NSAM, R4DAT(1,ID2), NSAM, R4DAT(1,ID1),
     1  LG, G)
        CALL MAXSN(LG, G, GMAX1, III)
        WRITE (6,*)'OPTIMUM LAG = ', GMAX, ' AT ', II - 1
        WRITE (6,*)'OPTIMUM LEAD = ', GMAX1, ' AT ', III - 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 ', III - 1
```

```
      WRITE (7,*)'LAG implies should increase time on 2nd trace.'
     END IF
     IF (IOPT .EQ. 8) THEN
      WRITE (6,*)'Enter lo-cut,slope,hi-cut,slope'
      READ (5,*) BUT1, BUT2, BUT3, BUT4
      RNL = ALOG10((2.*(10.**(BUT4/10.))) - 1.0)
      RNL = RNL / (2.*ALOG10(2.))
      DO 240 J = 1, NSAMR / 2 + 1
       RFR = DF * REAL(J - 1)
       TEM = 1. / (1. + ((RFR/BUT3)**(2.*RNL)))
       Y(J) = SQRT(TEM)
240   CONTINUE
      RNL = ALOG10((2.*(10.**(BUT2/10.))) - 1.0)
      RNL = RNL / (2.*ALOG10(2.))
      G(1) = 0.0
      DO 250 J = 2, NSAMR / 2 + 1
       RFR = DF * REAL(J - 1)
       TEM = 1. / (1. + ((BUT1/RFR)**(2.*RNL)))
       G(J) = SQRT(TEM)
250   CONTINUE
      DO 260 J = 1, NSAMR / 2 + 1
       Y(J) = Y(J) * G(J)
260   CONTINUE
      ISAM = 2
      DO 270 I = NSAMR, NSAMR / 2 + 2, -1
       Y(I) = Y(ISAM)
       ISAM = ISAM + 1
270   CONTINUE
271   DO 280 I = 1, NSAMR
       CSX(I) = CMPLX(Y(I),0.0)
280   CONTINUE
      CALL FFT(NSAMR, CSX, 1.)
      DO 290 I = 1, NSAMR / 2
       G(I + NSAMR/2) = REAL(CSX(I))
       G(I) = REAL(CSX(I + NSAMR/2))
290   CONTINUE
      LD = NSAMR
      CALL ZEROC(NSAMR,CSX)
      CALL NORMAN(NSAMR, G)
      LD = LD + NSAMR / 2
      DO 300 I = 1, NSAM
        CSX(I) = CMPLX(R4DAT(I,NTRACE),0.0)
        CSX1(I) = CMPLX(G(I),0.0)
300   CONTINUE
      CALL ZERO(NSAMR,G)
      CALL ZERO(NSAMR,Y)
      DO 310 I = NSAM + 1, NSAMR
       CSX(I) = 0.0
       CSX1(I) = 0.0
310   CONTINUE
      CALL FFT(NSAMR, CSX, -1.0)
      CALL POLAR(NSAMR,CSX,P1,P2)
      CALL NORMAN(NSAMR, P1)
      CALL FFT(NSAMR, CSX1, -1.0)
      CALL POLAR(NSAMR,CSX1,G,Y)
      CALL NORMAN(NSAMR, G)
      DO 301 I=1,NSAMR
       TEMP2(I,1)= P1(I)
       TEMP2(I,2)= G(I)
301   CONTINUE
      CALL WVSHPL(NSAMR/2+1,2,NSAMR,TEMP2,0.0,100.0,R1,
     1 'Amplitude spectrum','frequency (Hz)')
     ENDIF
     IF (IOPT .NE. 0) GO TO 10
     RETURN
     END

c--------------------------------------------------------------------
      SUBROUTINE MENUF(NSAM, NCHR, NSAMR, NCHRR, N1,
     1 N5, R4DAT,DZ, DT, NCR, CP, CW, FILSPC, AMPSPC,IBAT)
C
C  Subroutine to obtain F-K spectrum of recorded data
      DIMENSION R4DAT(NSAMR,NCHRR), NCR(2)
      CHARACTER*1 ANS, LOGYN, ANSWER, YN3D, SQRDAT
      CHARACTER FNAME*10, ASHADE*6
      COMPLEX CP(NSAMR,NCHRR), CW(NCHRR), CTEMP
      REAL KNYQ, FILSPC(N1,N5), CHTS(8), DBCHTS(8)
      REAL AMPSPC(N5,N1)
      REAL TEMP, XM
      DATA CHTS / 10.,20.,50.,100.,200.,500.,0,0 /
      DATA DBCHTS / 0.01,3.0,6.0,20.0,40.0,0,0,0 /
      NCHTS = 4
      NCHTDB = 5
      SQRDAT = 'N'
      ASHADE = 'colour'
      NF = NSAMR / 2 + 1
      NK = NCHRR + 1
      FNYQ = .5E6 / DT
      KNYQ = .5 / DZ
      DK = 2 * KNYQ / NCHRR
      DF = 2 * FNYQ / NSAMR
      IF (IBAT.NE.1)
     1 PRINT *,'KNYQ,FNYQ,NK,NF,DZ,DT',KNYQ,FNYQ,NK,NF
   10 IF (IBAT.NE.1) THEN
      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 (6,101)
  101 FORMAT (/'  10  Apply contour slice filter ')
      WRITE (6,102) SQRDAT
  102 FORMAT (/'  11  Square root of data for plot :',A1)
      WRITE (6,103) ASHADE
  103 FORMAT (/'  12  Colour/greyscale          :',A6)
      WRITE (6,110)
  110 FORMAT (/'  0  Return to MAIN MENU        ')
      ENDIF
      READ (5,*) IANSWR
      IF (IANSWR .EQ. 0) RETURN
      IF (IANSWR .EQ. 1) THEN
       IF (IBAT.NE.1) WRITE (6,*)'To or From  F-K space (T/F) :'
       READ (5,120) ANS
  120  FORMAT (A1)
       NCEN = NSAMR / 2 + 1
       KCEN = NCHRR / 2 + 1
       IF (ANS .EQ. 'F' .OR. ANS .EQ. 'f') THEN
        SIGNA = +1.0
        SIGNB = +1.0
        DO 130 J = 1, NCHRR
         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
C     SWAP AROUND TWO UPPER QUADRANTS
        DO 150 J = 1, NCHRR / 2 - 1
         DO 140 I = NCEN + 1, NSAMR
```

```
      CTEMP = CP(I,J + KCEN)
      CP(I,KCEN + J) = CP(I,KCEN - J)
      CP(I,KCEN - J) = CTEMP
140    CONTINUE
150   CONTINUE
   ELSE
    IF (IBAT.NE.1) WRITE (6,*)'RESPONSE OF FILTER?'
    READ (5,160) ANSWER
160   FORMAT (A1)
    IF (ANSWER .EQ. 'Y' .OR. ANSWER .EQ. 'y') THEN
     NCP = NSAM * NCHR
     CALL ZERO(NCP, R4DAT(1,1))
     R4DAT(512,12) = 1.0
    END IF
    SIGNA = -1.0
    SIGNB = -1.0
    NZERO = NCHRR * NSAMR * 2
    CALL ZERO(NZERO, CP(1,1))
    DO 180 J = 1, NCHR
     DO 170 I = 1, NSAM
      CP(I,J) = CMPLX(R4DAT(I,J),0.0)
170    CONTINUE
180   CONTINUE
190   FORMAT (4(I4,1X,F7.3,1X,F7.3))
C Remove common receiver channel
     IF (NCR(1) .NE. 0) THEN
      IF (IBAT.NE.1) PRINT*,'Enter com rec chan 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
    CALL FT2D(NSAMR, NCHRR, CP, SIGNA, SIGNB, CW)
    IF (ANS .EQ. 'F' .OR. ANS .EQ. 'f') THEN
     IF (IBAT.NE.1) PRINT *, 'WRITING CP TO R4DAT'
220    DO 230 J = 1, NCHR
      DO 230 I = 1, NSAM
       R4DAT(I,J) = REAL(CP(I,J))
230   CONTINUE
    END IF
   ELSE IF (IANSWR .EQ. 4 .OR. IANSWR .EQ. 10) THEN
C  PUT filter response into array FILSPC
    IF (IBAT.NE.1) PRINT *, 'NF,NK,DK,DF', NF, NK, DK, DF
    IF (IBAT.NE.1) WRITE (6,*)'PASS or REJECT filter (P/R)?'
    READ (5,160) ANS
    IF(IANSWR .EQ. 4) THEN
     CALL PIE(NK, NF, FILSPC, DK, DF, IBAT)
    ELSE IF (IANSWR .EQ. 10) THEN
     DO 440 J = 1, NCHRR
      DO 450 I = 1, NF
       AMPSPC(J,I) = CABS(CP(I,J))
450     CONTINUE
440    CONTINUE
     CALL SLICE(NK, NF, AMPSPC, FILSPC, DK, DF)
    ENDIF
    IF (ANS .EQ. 'R') THEN
     DO 250 I = 1, NCHRR
      DO 240 J = 1, NF
       FILSPC(J,I) = 1.0 - FILSPC(J,I)
240     CONTINUE
250    CONTINUE
    END IF
C
C APPLY FILTER FILSPC
C
     DO 270 I = 1, NCHRR / 2
      DO 260 J = 1, NF
       CP(J,I) = CP(J,I) * FILSPC(J,I + NCHRR/2)
260    CONTINUE
270   CONTINUE
     DO 290 I = NCHRR / 2 + 1, NCHRR
      DO 280 J = 1, NF
       CP(J,I) = CP(J,I) * FILSPC(J,I - NCHRR/2)
280    CONTINUE
290   CONTINUE
    ELSE IF (IANSWR .EQ. 2) THEN
     DO 300 J = 1, NCHRR
      DO 300 I = 1, NF
       AMPSPC(J,I) = CABS(CP(I,J))
300   CONTINUE
C
C  rearrange array so that K=0 axis lies at sample NCHRR/2+1
C
     DO 310 I = 1, NCHRR / 2
      DO 310 J = 1, NF
       TEMP = AMPSPC(I,J)
       AMPSPC(I,J) = AMPSPC(I + NCHRR/2,J)
       AMPSPC(I + NCHRR/2,J) = TEMP
310   CONTINUE
     IF (IBAT.NE.1) WRITE (6,*)'Enter name of file to write to :'
     READ (5,320) FNAME
320   FORMAT (A10)
     OPEN (4,FILE=FNAME,STATUS='NEW',FORM=
   1 'UNFORMATTED')
     WRITE (4) ((AMPSPC(J,I),J=1,NCHRR),I=1,NF)
     CLOSE (4)
    ELSE IF (IANSWR .EQ. 5 .OR. IANSWR .EQ. 7 .OR. IANSWR
   & .EQ. 9 ) THEN
C  write amplitude spectrum to correctly orientated array AMPSPC
     DO 340 J = 1, NCHRR
      DO 330 I = 1, NF
       IF (SQRDAT .EQ. 'Y') THEN
        AMPSPC(J,I) = SQRT(CABS(CP(I,J)))
       ELSE
        AMPSPC(J,I) = CABS(CP(I,J))
       ENDIF
330    CONTINUE
340   CONTINUE
     ISOLOG = 0
     IF (IANSWR .EQ. 9) THEN
      IF (IBAT.NE.1) 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 = 0.0
      DO 360 J = 1, NF
       DO 350 I = 1, NCHRR
        IF (AMPSPC(I,J) .GT. XM) XM = AMPSPC(I,J)
350     CONTINUE
360    CONTINUE
      IF (IBAT.NE.1) PRINT *, 'MAX AMPLITUDE = ', XM
      DO 380 I = 1, NF
       DO 370 J = 1, NCHRR
        IF (AMPSPC(J,I) .LE. 1.0E-8) THEN
         AMPSPC(J,I) = 1000.0
        ELSE
         AMPSPC(J,I) = -20 * LOG10(AMPSPC(J,I)/XM)
        END IF
370     CONTINUE
380    CONTINUE
     END IF
C  Now rearrange K space so that 0.0,0.0 is in centre of K axis
      DO 400 I = 1, NCHRR / 2
       DO 390 J = 1, NF
        TEMP = AMPSPC(I,J)
        AMPSPC(I,J) = AMPSPC(I + NCHRR/2,J)
        AMPSPC(I + NCHRR/2,J) = TEMP
390    CONTINUE
400   CONTINUE
```

```
      DO 410 J = 1, NF
       AMPSPC(NK,J) = AMPSPC(1,J)
  410  CONTINUE
C  Now call plotting routine
      LOGYN = 'N'
      IF (IANSWR .EQ. 7) THEN
       LOGYN = 'Y'
       YN3D = 'N'
       CALL FKPLOT(NK, NF, AMPSPC, DT, DZ, NCHTDB,
     1      DBCHTS, LOGYN, YN3D, SQRDAT, ISHADE)
      ELSE IF (IANSWR .EQ. 5) THEN
       LOGYN = 'N'
       YN3D = 'N'
       CALL FKPLOT(NK, NF, AMPSPC, DT, DZ, NCHTS, CHTS,
     1      LOGYN, YN3D, SQRDAT, ISHADE)
      ELSE
       YN3D = 'Y'
       IF (ISOLOG .EQ. 1) THEN
        LOGYN = 'Y'
        CALL FKPLOT(NK, NF, AMPSPC, DT, DZ, NCHTDB,
     1       DBCHTS, LOGYN ,YN3D, SQRDAT, ISHADE)
       ELSE
        LOGYN = 'N'
        CALL FKPLOT(NK, NF, AMPSPC, DT, DZ, NCHTS, CHTS,
     1       LOGYN ,YN3D, SQRDAT, ISHADE)
       END IF
      END IF
     ELSE IF (IANSWR .EQ. 6) THEN
      IF (IBAT.NE.1) PRINT *, ' No. of contours required (<=8) :'
      READ (5,*) NCHTS
      DO 420 J = 1, NCHTS
       IF (IBAT.NE.1) WRITE (6,*)' Contour level ', J, ' : '
       READ (5,*) CHTS(J)
  420  CONTINUE
     ELSE IF (IANSWR .EQ. 8) THEN
      IF (IBAT.NE.1) PRINT *, ' No. of contours required (<=8) :'
      READ (5,*) NCHTDB
      DO 430 J = 1, NCHTDB
       IF (IBAT.NE.1) WRITE (6,*)' Contour level ', J, ' : '
       READ (5,*) DBCHTS(J)
  430  CONTINUE
     ELSE IF (IANSWR .EQ. 11) THEN
      IF (SQRDAT .EQ. 'N') THEN
       SQRDAT = 'Y'
      ELSE IF (SQRDAT .EQ. 'Y') THEN
       SQRDAT = 'N'
      END IF
     ELSE IF (IANSWR .EQ. 12) THEN
       print*,'Enter 1 for a colour plot, 3 for greyscale '
       read(5,*) ISHADE
       if(ISHADE.eq.1)ASHADE = 'Colour'
       if(ISHADE.eq.3)ASHADE = 'Greysc'
      END IF
      IF (IANSWR .NE. 0) GO TO 10
      RETURN
      END

c-----------------------------------------------------------------------
      SUBROUTINE MENUIO(NSAM, NCHR, NRECS, NSHOT, LEN,

     1 NFILES, NFILE,IPDISC1, IPDISC2, IHFORM, INPOPT, OPDISC,
     2 IPFORM, OPFORM, IHEAD, IFPROC, IFNOHD, IFILE2,
     3 ICSCRG, ISNRN, IBAT)
C
C  Menu for adjustment of I/O parameters
C
      DIMENSION NFILE(2)
      CHARACTER*25 IPDISC1, IPDISC2, OPDISC
      CHARACTER INPOPT*3, IPFORM*6, OPFORM*6, IHFORM*4
C
   10 IF (IBAT.NE.1) THEN
      WRITE (6,*)
      WRITE (6,*)'          I/O Setup Menu          '
      WRITE (6,*)'          ~~~ ~~~~~ ~~~~          '
      WRITE (6,20) IHFORM
   20 FORMAT (/'  1  File format (heads in/out)       :', A4)
      IF (IFILE2.EQ.0) WRITE (6,30) IPDISC1
   30 FORMAT ('  2  Input File Name (in /ts/global/psr)  :', A25)
      WRITE (6,40) OPDISC
   40 FORMAT ('  3  Output File Name (in /ts/global/psr) :', A25)
      WRITE (6,51) IPFORM
   51 FORMAT (/' 13  Input Format          :', A6)
      WRITE (6,50) OPFORM
   50 FORMAT ('  4  Output Format          :', A6)
      WRITE (6,70) NSAM
   70 FORMAT (/'  5  Record Length (samples)  :', I4)
      WRITE (6,80) NCHR
   80 FORMAT ('  6  Number of channels      :', I3)
      IF (IFILE2 .EQ. 0) THEN
       WRITE (6,90) (NFILE(I),I=1,NFILES)
      ELSE
       WRITE (6,90) NFILE(2)
      ENDIF
   90 FORMAT ('  7  I.D. nos. of files      :', 1X, I2, 1X, I2)
      WRITE (6,91) NFILES
   91 FORMAT ('  8  Number of input files    :', I1)
      IF (NFILES.EQ.2.OR.IFILE2.EQ.1) WRITE (6,92) IPDISC2
   92 FORMAT ('  9  Input Second File Name    :', A25)
      WRITE (6,61) IHEAD
   61 FORMAT (/' 15  Header? (1=short,0=normal):', I2)
      WRITE (6,62) IFPROC
   62 FORMAT (' 16  Read in PROC format? (1=y) :', I2)
      WRITE (6,63) IFNOHD
   63 FORMAT (' 17  Header in file? (1=n)      :', I2)
      WRITE (6,64) IFILE2
   64 FORMAT (' 18  Just 2ary file? (1=y)      :', I2)
      WRITE (6,65) ICSCRG
   65 FORMAT (' 19  CRG <-> CSG?   (1=y)      :', I2)
      WRITE (6,66) ISNRN
   66 FORMAT (' 20  shot no = rec no? (1=y)   :', I2)
      WRITE (6,120)
  120 FORMAT (/'  0  Return to MAIN MENU          ')
      ENDIF
C
      READ (5,*) IANSWR
  130 FORMAT (A25)
      IF (NSAM.LT.1024) THEN
        LEN = 4104
      ELSE
       LEN = (NSAM + 2) * 4
      ENDIF
      IF (IANSWR .EQ. 1) THEN
        IF (IHFORM .EQ. 'IN') THEN
          IHFORM = 'OUT'
        ELSE
          IHFORM = 'IN'
        END IF
      END IF
      IF (IANSWR .EQ. 2) THEN
        IF (IBAT.NE.1) WRITE (6,*)
        IF (IBAT.NE.1) WRITE (6,*)'      Input Disc name   :'
        READ (5,130) IPDISC1
      END IF
      IF (IANSWR .EQ. 3) THEN
        IF (IBAT.NE.1) WRITE (6,*)
        IF (IBAT.NE.1) WRITE (6,*)'      Output Disc name   :'
        READ (5,130) OPDISC
      END IF
      IF (IANSWR .EQ. 13) THEN
        IF (IPFORM .EQ. 'DIRECT') THEN
          IPFORM = 'ASCII'
        ELSE
          IPFORM = 'DIRECT'
        END IF
```

```
      END IF
      IF (IANSWR .EQ. 4) THEN
        IF (OPFORM .EQ. 'DIRECT') THEN
          OPFORM = 'ASCII'
        ELSE
          OPFORM = 'DIRECT'
        END IF
      END IF
      IF (IANSWR .EQ. 5) THEN
        IF (IBAT.NE.1) WRITE (6,*)
        IF (IBAT.NE.1) WRITE (6,*)'     Number of samples  :'
        READ (5,*) NSAM
      END IF
      IF (IANSWR .EQ. 6) THEN
        IF (IBAT.NE.1) WRITE (6,*)
        IF (IBAT.NE.1) WRITE (6,*)'     Number of channels :'
        READ (5,*) NCHR
      END IF
      IF (IANSWR .EQ. 7) THEN
        IF(NFILES.EQ.2) THEN
          IF (IBAT.NE.1) WRITE (6,*)'     Shot I.D. 1        :'
          READ (5,*) NFILE(1)
          IF (IBAT.NE.1) WRITE (6,*)'     Shot I.D. 2        :'
          READ (5,*) NFILE(2)
        ELSE
          IF (IBAT.NE.1) WRITE (6,*)'     Shot I.D.          :'
          IF (IFILE2 .EQ. 0) THEN
            READ (5,*) NFILE(1)
          ELSE
            READ (5,*) NFILE(2)
          ENDIF
        END IF
        NSHOT = NFILE(1)
      END IF
      IF (IANSWR .EQ. 8) THEN
       IF (NFILES .EQ. 1) THEN
        NFILES = 2
       ELSE
        NFILES = 1
       END IF
      END IF
      IF (IANSWR .EQ. 9) THEN
        IF (IBAT.NE.1) WRITE (6,*)
        IF (IBAT.NE.1) WRITE (6,*)'     Input Disc name   :'
        READ (5,130) IPDISC2
      END IF
      IF (IANSWR .EQ. 14) THEN
        OPFORM = 'SEG-Y '
      END IF
      IF (IANSWR .EQ. 15) THEN
        IF (IHEAD .EQ. 0) THEN
          IHEAD = 1
        ELSE
          IHEAD = 0
        END IF
      END IF
      IF (IANSWR .EQ. 16) THEN
        IF (IFPROC .EQ. 0) THEN
          IFPROC = 1
        ELSE
          IFPROC = 0
        END IF
      END IF
      IF (IANSWR .EQ. 17) THEN
        IF (IFNOHD .EQ. 0) THEN
          IFNOHD = 1
        ELSE
          IFNOHD = 0
        END IF
      END IF
      IF (IANSWR .EQ. 18) THEN
        IF (IFILE2 .EQ. 0) THEN
          IFILE2 = 1
        ELSE
          IFILE2 = 0
        END IF
      END IF
      IF (IANSWR .EQ. 19) THEN
        IF (ICSCRG .EQ. 0) THEN
          ICSCRG = 1
        ELSE
          ICSCRG = 0
        END IF
      END IF
      IF (IANSWR .EQ. 20) THEN
        IF (ISNRN .EQ. 0) THEN
          ISNRN = 1
        ELSE
          ISNRN = 0
        END IF
      END IF
      IF (IANSWR .NE. 0) GO TO 10
      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
C MAIN ROUTINE (this advice seems to be for when you leave the
C routine) (ie. rearrange FILSPC as you apply it to CP  PSR 92)
C   sample ordering in K space is 1 = -KNQ
C                               nk/2+1 = 0    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
      SUBROUTINE PIE(NK, NF, PIEFLT, DK, DF,IBAT)
      REAL PIEFLT(NF,NK), DIST, RHISLO, RHICTS, RLOSLO,
     1     RLOCTS, RM3, RM6, KKNYQ
      DOUBLE PRECISION HISLOP, HICTSL, LOSLOP, LOCTSL, M3
      DOUBLE PRECISION M6
      INTEGER KNYQ, K0, KL, F0, FL
      CHARACTER*1 ANS
      PI = 3.1415926535
      KNYQ = NK / 2 + 1
      KKNYQ = NK / 2 * DK
      IF (IBAT.NE.1) WRITE (6,*)'ZERO QUAD (L/R/No)?(L=-Kspace) '
      READ (5,10) ANS
   10 FORMAT (A1)
      F0 = 1
      FL = NF
      K0 = 1
      KL = NK
      DO 20 K = 1, NK
       CALL ZERO(NF, PIEFLT(1,K))
   20 CONTINUE
      IF (ANS .EQ. 'L' .OR. ANS .EQ. 'l') THEN
       DO 30 K = 1, KNYQ - 1
        CALL ZERO(NF, PIEFLT(1,K))
   30  CONTINUE
       K0 = 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
      IF (IBAT.NE.1)
     1 WRITE (6,*)'TAPER IN OR OUTWARDS FROM SLOPE (I/O) ?'
      READ (5,10) ANS
      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/p HIGH CUTOFF SLOPE (m/s) (-VE for L quad):'
      READ (5,*) HISLOP
```

```
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1)
     1   WRITE (6,*)'INPUT HI 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
       IF (IBAT.NE.1)
     1 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
      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/P LOW CUTOFF SLOPE (m/s) (-VE for L quad) :'
      READ (5,*) LOSLOP
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1)
     1   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
       IF (IBAT.NE.1)
     1 PRINT*,'LOSLOP,LOCTSL,M6',LOSLOP,LOCTSL,M6
      END IF
      IF (IBAT.NE.1) WRITE (6,*)'Enter no. of wraparounds required '
      READ (5,*) NWRAPS
      IF (ANS .EQ. 'I' .OR. ANS .EQ. 'i') THEN
       IF (IBAT.NE.1)
     1  WRITE (6,*)'INPUT DISTANCE IN SAMPLES FOR TAPER :'
       READ (5,*) NSAMTP
      END IF
      IF (IBAT.NE.1)
     1 WRITE (6,*)'INPUT LENGTH OF TAPER FOR K-SPACE (sams)'
      READ (5,*) LNTAPK
      IF (IBAT.NE.1)
     1 WRITE (6,*)'INPUT LENGTH OF TAPER FOR F-SPACE (sams)'
      READ (5,*) LNTAPF
      RHISLO = REAL(HISLOP)
      RHICTS = REAL(HICTSL)
      RLOSLO = REAL(LOSLOP)
      RLOCTS = REAL(LOCTSL)
      RM3 = REAL(M3)
      RM6 = REAL(M6)
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1) THEN
       WRITE (6,*)'Enter frequency axis intercepts Hicut, Hizero, '//
     1       'Locut, Lozero :'
       READ (5,*) RHSLIN, RHCTIN, RLSLIN, RLCTIN
      WRITE(6,*)'RHICTS,RHCTIN,MULT1',RHICTS,RHCTIN,MULT1
      WRITE (6,*)'RHISLO,RHSLIN,MULT2',RHISLO,RHSLIN,MULT2
       ELSE
        READ (5,*) RHSLIN, RHCTIN, RLSLIN, RLCTIN
       ENDIF
 50   DO 70 J = K0, KL
       JIND = 0
       IF (NWRAPS .EQ. 0 .AND. J .EQ. 33) JIND = 1
       IF (IBAT.NE.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)
     1       THEN
         D1 = (Y - HISLOP*X - RHSLIN) / (HISLOP - M3)
         D2 = (Y - HICTSL*X - RHCTIN) / (HICTSL - M3)
         D1 = ABS(D1)
         D2 = ABS(D2)
C NOTE : COSM3 TERM CANCELS OUT
         THETA = PI * D2 / (D1 + D2)
         RESP = 0.5 * (1 - COS(THETA))
        ELSE IF (DIST(X,Y,RLOSLO,RLSLIN,MULT3) .GT. 0.0)
     1       THEN
         RESP = 1.0
        ELSE IF (DIST(X,Y,RLOCTS,RLCTIN,MULT4) .GT. 0.0)
     1       THEN
         D1 = (Y - LOSLOP*X - RLSLIN) / (LOSLOP - M6)
         D2 = (Y - LOCTSL*X - RLCTIN) / (LOCTSL - M6)
         D1 = ABS(D1)
         D2 = ABS(D2)
         THETA = PI * D2 / (D1 + D2)
         RESP = 0.5 * (1 - COS(THETA))
        ELSE
         RESP = 0.0
        END IF
        IF (RESP .EQ. 0.0) GO TO 60
        IF (PIEFLT(I,J) .GT. 0.0) THEN
         PIEFLT(I,J) = PIEFLT(I,J) * RESP
        ELSE
         PIEFLT(I,J) = RESP
        END IF
 60     CONTINUE
 70   CONTINUE
      IF (NWRAPS .NE. 0) THEN
       NWRAPS = NWRAPS - 1
       RHCTIN = 2.0 * ABS(RHICTS) * KKNYQ + RHCTIN
       RHSLIN = 2.0 * ABS(RHISLO) * KKNYQ + RHSLIN
       RLSLIN = 2.0 * ABS(RLOSLO) * KKNYQ + RLSLIN
       RLCTIN = 2.0 * ABS(RLOCTS) * KKNYQ + RLCTIN
       IF (IBAT.NE.1)
     1  PRINT *, 'INTERCEPTS :',RHCTIN,RHSLIN,RLSLIN,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
      D0 = -REAL(NSAMTP)
      D1 = -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.
     1       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
     IF (PIEFLT(I,J) .GT. 0.0) THEN
      PIEFLT(I,J) = PIEFLT(I,J) * RESP
     ELSE
      PIEFLT(I,J) = RESP
     END IF
80   CONTINUE
90  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 NYQ (-KNYQ) TO HIGH NYQ (+KNYQ)
C
   DO 140 I = 1, NF
    PIEFLT(I,1) = PIEFLT(I,NK)
140 CONTINUE
   RETURN
   END

c-----------------------------------------------------------------
   SUBROUTINE PIKSRT(N, ARR)
C
C SOURCE : Numerical recipes
C
   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)
10   CONTINUE
    I = 0
20   ARR(I + 1) = A
30 CONTINUE
   RETURN
   END

c-----------------------------------------------------------------
   SUBROUTINE PRO16A(NSAM,NCHR,NSAMR,NCHRR,NA,
  1 R4DAT,A,NFIRST,IBAT)

C  Crosscorrelation of data .... calls CROSS
c  Writes out to unit=2 the 'fb' where best cc with the
c  known wavelet A occured.
c  From Kragh (1990)
   REAL TEMP(1024),TEMP2(1024)
   INTEGER NF(64)
   DIMENSION NFIRST(NCHRR)
   DIMENSION R4DAT(NSAMR,NCHRR),A(NSAMR)
   IF (IBAT.NE.1) PRINT*,'Enter Crosscorrelation window start'
   IF (IBAT.NE.1) PRINT*,'(0=RETURN,99=FB+?)'
   READ*, NWIN1
   IF(NWIN1.EQ.0)RETURN
   IF(NWIN1.EQ.99) THEN
    IFB = 1
    IF (IBAT.NE.1) PRINT*,'Window start (0=FB)'
    READ*, NWIN1
    IF (IBAT.NE.1) PRINT*,'Window finish (0=FB)'
    READ*, NWIN2
   ELSE
    IFB = 0
    IF (IBAT.NE.1) PRINT*,'Enter Crosscorrelation window finish'
    READ*, NWIN2
   ENDIF
   NAUTO = NWIN2-NWIN1+1
   CALL ZERO(NSAMR,TEMP)
   DO 100 I = 1,NCHR
    K = 1
    CALL ZERO(1024,TEMP2)
    IF (IFB.EQ.1) THEN
     N1 = NWIN1 + NFIRST(I)
     N2 = NWIN2 + NFIRST(I)
    ELSE
     N1 = NWIN1
     N2 = NWIN2
    ENDIF
    DO 200 J = N1,N2
     TEMP(K) = R4DAT(J,I)
     K = K+1
200  CONTINUE
    CALL CROSS(NAUTO,TEMP,NA,A,NAUTO,TEMP2)
    CALL MAXSN(NAUTO, TEMP2, X1, NF(I))
    NF(I) = NF(I) + N1 - 1
    WRITE(2,*) '1'
    WRITE(2,*) NF(I)
    WRITE(3,*)I,NFIRST(I),NF(I)
100 CONTINUE
   CLOSE (2)
   IF (IBAT.NE.1)
  1 PRINT*,'Correlation of ',NCHR,' traces complete'
   RETURN
   END

c-----------------------------------------------------------------
   SUBROUTINE RDFILE(NSAM,NCHR,NSAMR,NCHRR,R4DAT,
  1 NREAD,NFIRST,NRECS,RECDEP,DBGAIN,GCMSCL,
  2 NFILE,NCR,NSHOT,DT,LEN,SORPOS,NPROCS,IDPROC,
  3 IPDISC1,IPDISC2,TEMP,IHFORM,INPOPT,IPFORM,
  4 IFPROC,IFNOHD,A,PROC,ICSCRG,ISNRN,IBAT)
C  reads in arrays to R4DAT
   DIMENSION R4DAT(NSAMR,NCHRR), NFIRST(NCHRR),
  1 RECDEP(NCHRR), DBGAIN(NCHRR),TEMP(NSAMR),NCR(2)
   DIMENSION GCMSCL(NCHRR), NFILE(2), IDPROC(5,NCHRR)
   INTEGER NF(64)
   CHARACTER IHFORM*4, INPOPT*3, B(9)*20, IPFORM*6
   CHARACTER*25 PROC(20), IPDISC1, IPDISC2, IPDISC
   CHARACTER ANS*1, CRAP*10, IP1*50, IP2*50,A*180
C
C NSHOT = NO. OF SHOT (i.d.)
C
   IF (NREAD.EQ.1) THEN
    IPDISC = IPDISC1
    NSHOT = NFILE(1)
   ELSE
    IPDISC = IPDISC2
    NSHOT = NFILE(2)
   ENDIF
   IF (IFNOHD.EQ.1) THEN
    IDCODE = (NSHOT - 1) * NCHR
   ELSE
    IDCODE = (NSHOT - 1) * (NCHR + 1) + 1
   ENDIF
   IF (IBAT.NE.1) WRITE (6,*)'Stack or New (1 0 -1 -- REV POL ) :'
```

```
      READ (5,*) ISTACK
      K24 = NSAMR * NCHRR
      IF (ISTACK .EQ. 0) CALL ZERO(K24, R4DAT(1,1))
C
C READ IN HEADER PARAMETERS
C
      DO 90 J = 1, NCHR
        DO 80 I = 1, 5
          IDPROC(I,J) = 0
 80     CONTINUE
 90   CONTINUE
      NPROCS = 0
      IF (IBAT.NE.1) PRINT *, 'READING HEADERS ......'
C.................................................................
      IF (IPFORM .EQ. 'DIRECT') THEN
       IP1 = '/ts/global/psr/'//IPDISC
       IF (IHFORM .EQ. 'OUT') THEN
        IDCODE = NSHOT
        IP1 = IP1(1:lnb(IP1))//'.h'
        LEN = (NSAM + 2) * 4
       ENDIF
       OPEN (10,FILE=IP1,STATUS='OLD',ACCESS='DIRECT',
     1      RECL=LEN, IOSTAT=K2,ERR=92)
 92    IF (K2.NE.0) THEN
        PRINT*,'FILE DOES NOT EXIST'
        PRINT *, 'Enter a character'
        READ (5,120) ANS
        RETURN
       ENDIF
       IF (IFPROC.EQ.1) THEN
        READ (10,REC=IDCODE,IOSTAT=K1,END=91)A,NSAMS,
     1     NRECS, DT, NFIRST
        IF (IBAT.NE.1) THEN
         PRINT*,'======================================'
         PRINT*,'=    The header says there are ',NRECS,' traces ='
         PRINT*,'=    with ',NSAMS,' samples / trace          ='
         PRINT*,'======================================'
        ENDIF
       ELSE IF (IFNOHD.EQ.1) THEN
        IF (IBAT.NE.1) PRINT*,'No header read in'
        NRECS=NCHR
       ELSE
        IF (IHFORM .EQ. 'OUT') THEN
         IF (ICSCRG.EQ.1.OR.ISNRN.EQ.1) THEN
          DO 96 I = 1,NCHR
           READ (10,REC=I,IOSTAT=K1,END=91) A, NSAMS,
     1      NRECS, DT, NF, SORPOS, RECDEP
           IF (ICSCRG.EQ.1) NFIRST(I) = NF(NSHOT)
           IF (ISNRN.EQ.1) NFIRST(I) = NF(I)
 96       CONTINUE
         ELSE
          READ (10,REC=IDCODE,IOSTAT=K1,END=91) A, NSAMS,
     1     NRECS, DT, NFIRST, SORPOS, RECDEP
         ENDIF
         IF (IBAT.NE.1) THEN
          PRINT*,'======================================'
          PRINT*,'=   The header says there are ',NRECS,' traces ='
          PRINT*,'=   with ',NSAMS,' samples / trace         ='
          PRINT*,'======================================'
         ENDIF
         NCHR = NRECS
         NSAM = NSAMS
        ELSE
         READ (10,REC=IDCODE,IOSTAT=K1,END=91) A, SORPOS,
     1    NRECS,RECDEP,DBGAIN,GCMSCL,NFIRST, NCR,
     2    NPROCS, IDPROC, DT
        ENDIF
       ENDIF
C
 91    IF (K1.NE.0.AND.IBAT.NE.1) PRINT*,'PROBLEM HEADER'
       IF (GCMSCL(1) .EQ. 0.0) GCMSCL(1) = 1.0
       IF (IHFORM .EQ. 'OUT') THEN
        CLOSE (10)
        IDCODE = (NSHOT-1) * NCHR
        IP2 = '/ts/global/psr/'//IPDISC
        IP2 = IP2(1:lnb(IP2))//'.d'
        OPEN (10,FILE=IP2,STATUS='OLD',ACCESS='DIRECT',
     1     RECL=LEN,IOSTAT=K2,ERR=92)
       ENDIF
c
       DO 110 J = 1, NCHR
        IF (ICSCRG.EQ.0.AND.ISNRN.EQ.0) THEN
         NREC = IDCODE + J
        ELSE
         IF (IHFORM .EQ. 'OUT') THEN
          IF (ICSCRG.EQ.1) NREC = (J-1)*NCHR + NSHOT
          IF (ISNRN.EQ.1) NREC = (J-1)*NCHR + J
         ELSE
          IF (ICSCRG.EQ.1) NREC = (J-1)*(NCHR+1) + NSHOT +1
          IF (ISNRN.EQ.1) NREC = (J-1)*(NCHR+1) +1 + J
         ENDIF
        ENDIF
        IF(ISTACK.EQ.0)THEN
         READ(10,REC=NREC,IOSTAT=K4,END=93)
     1      (R4DAT(I,J),I=1,NSAM)
 93      IF (K4.NE.0.AND.IBAT.NE.1) PRINT*,'PROBLEM RECORD',
     1      NREC,IDCODE,K4
        ELSE
         READ(10,REC=NREC,IOSTAT=K4,END=95)
     1      (TEMP(I),I=1,NSAM)
 95      IF (K4.NE.0.AND.IBAT.NE.1)PRINT*,'PROBLEM RECORD',
     1      NREC,IDCODE,K4
         DO 100 I = 1, NSAM
          IF (ISTACK .GE. 0) THEN
           R4DAT(I,J) = R4DAT(I,J) + TEMP(I)
          ELSE
           R4DAT(I,J) = R4DAT(I,J) - TEMP(I)
          END IF
 100     CONTINUE
        ENDIF
 110   CONTINUE
c.................................................................
      ELSE
       IF (IBAT.NE.1) PRINT*,'Ignore header (Y/N)?'
       READ(5,120) ANS
       OPEN(10,FILE=IPDISC,STATUS='OLD',FORM='FORMATTED')
       READ(10,901) IDCODE
 901   FORMAT(I4)
       IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
        READ(10,904) CRAP
 904    FORMAT(A10)
        GOTO 111
       ELSE
        READ (10,902) A, SORPOS, NRECS, RECDEP, DBGAIN,
     1  GCMSCL, NFIRST, NCR, NPROCS, IDPROC, DT
 902    FORMAT(A180,F6.2,I2,23F6.2,23F6.2,23F6.2,23I4,2I2,I2,
     1     115I4,F6.2)
       ENDIF
 111   DO 20 J = 1, NCHR
        READ(10,903) (R4DAT(I,J),I=1,NSAM)
 903    FORMAT(6(F13.7))
 20    CONTINUE
      ENDIF
c.................................................................
      CLOSE(10)
      IF (NRECS.NE.NCHR.AND.IBAT.NE.1) THEN
       PRINT*,'NCHR and NRECS are not equal'
       PRINT*,'NCHR, NRECS =',NCHR,NRECS
      ENDIF
C READ IN SEISMOGRAM RECORDS
      IF (IBAT.NE.1) PRINT *, 'Reset parameters to defaults (Y/N)?'
      READ (5,120) ANS
 120  FORMAT (A1)
      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
```

```
      OPEN(UNIT=1,FILE='xhrp.dfault1')
        READ (1,130) B, INPOPT, IPDISC1, IPDISC2, OPDISC,
     1 IPFORM, OPFORM, IHFORM
130    FORMAT (A20, 8(/A20)/A3/A25/A25/A25/A6/A6/A4)
        A = B(1) // B(2) // B(3) // B(4) // B(5) // B(6) // B(7) // B(
     1  8) // B(9)
        READ (1,140) NA, NCHR, NA, NA, NA, NCR(1), NCR(2),
     1 SORPOS, TOPREC, RECSEP, DT, SA
140    FORMAT (I4/I3/I1/I3/I3/I2/I2/F6.1/F6.1/F6.2/F6.1/F6.2)
        RECDEP(1) = TOPREC
        DO 150 I = 2, NCHR
          IF (I .EQ. NCR(2)) THEN
            RECDEP(I) = RECDEP(I - 1)
          ELSE
            RECDEP(I) = RECDEP(I - 1) + RECSEP
          END IF
150    CONTINUE
      CLOSE(1)
      END IF
170 RETURN
      END

c-----------------------------------------------------------------------
C. Subroutine to save traces into a temporary array, and once done, to
C  move this array to the active R4DAT array
      SUBROUTINE SAVETR(NSAM,NCHR,NSAMR,NCHRR,R4DAT,
     1      NFIRST,RTEMP,NFTEMP,NCHSV,NCHDEF,IBAT)
      DIMENSION R4DAT(NSAMR,NCHRR),NFIRST(NCHRR)
      DIMENSIONRTEMP(NSAMR,NCHRR), NFTEMP(NCHRR)
      IF (IBAT.NE.1) THEN
       PRINT*,'  1.-----  Save a specified trace to temp array'
       PRINT*,'  2.-----  Move saved traces from temp array'
       PRINT*,'  3.-----  Scrub temp array'
       PRINT*,'  4.-----  Save successive traces to temp array'
      ENDIF
      READ (5,*) IANSWR

      IF(IANSWR.EQ.1) THEN
       IF (IBAT.NE.1)
     1  PRINT*,' Enter trace no. to save - 0=trace',NCHDEF
       READ(5,*) NSAVE
       IF (NSAVE.EQ.0) NSAVE = NCHDEF
       NCHSV=NCHSV+1
       NFTEMP(NCHSV) = NFIRST(NSAVE)
       DO 1 J = 1,NSAM
        RTEMP(J,NCHSV) = R4DAT(J,NSAVE)
     1  CONTINUE
      IF (IBAT.NE.1) PRINT*,' OK! Trace',NSAVE,' saved to
     1 channel',NCHSV
      ELSEIF(IANSWR.EQ.2)  THEN
       CALL ZERO2(NSAMR,NCHRR,R4DAT)
       CALL ZEROI(NCHRR,NFIRST)
       DO 2 I = 1,NCHSV
        NFIRST(I) = NFTEMP(I)
        DO 3 J = 1,NSAM
         R4DAT(J,I) =RTEMP(J,I)
3     CONTINUE
2    CONTINUE
       NCHR=NCHSV
       IF (IBAT.NE.1) THEN
        PRINT*,' OK!',NCHSV,' traces moved to active array R4DAT'
       ENDIF
      ELSEIF(IANSWR.EQ.3)  THEN
       CALL ZERO2(NSAMR,NCHRR,RTEMP)
       NCHSV = 0
      ELSEIF(IANSWR.EQ.4) THEN
       IF (IBAT.NE.1)
     1  PRINT*,' Enter first,last trace no. to save'
       READ(5,*) NSAVE1,NSAVE2
       NCHNS = NSAVE2-NSAVE1+1
       DO 5 K = NSAVE1,NSAVE2
        NCHSV=NCHSV+1
        NFTEMP(NCHSV) = NFIRST(K)
        DO 4 J = 1,NSAM
         RTEMP(J,NCHSV) = R4DAT(J,K)
4     CONTINUE
5    CONTINUE
      IF (IBAT.NE.1) PRINT*,' OK! Traces',NSAVE1,'to',NSAVE2,' saved c
     1to channels',NCHSV-NCHNS+1,'to',NCHSV
      ENDIF
      RETURN
      END

c-----------------------------------------------------------------------
C . subroutine to calculate and perform contour slice filter for
C   application to FK spectrum
      SUBROUTINE SLICE(NK, NF, AMPSPC, FILSLI, DK, DF)

      REAL AMPSPC(NK,NF), FILSLI(NF, NK)
      DO 20 K = 1, NK
       CALL ZERO(NF, FILSLI(1,K))
20 CONTINUE
      DO 1 I=1,NK
       CALL MAXSN(NF, AMPSPC(I,1), X1, II)
       IF (X1 .GT. XMAX) XMAX = X1
1 CONTINUE
      PRINT*, 'XMAX = ', XMAX
      WRITE (6,*)'Enter cut-off contour level as % of max '
      READ (5,*) SLIC
      CUT = XMAX * SLIC/100.
      DO 3 I=1,NK
       DO 2 J=1,NF
        IF (AMPSPC(I,J).LT.CUT) THEN
         FILSLI(I,J) = 0.0
        ELSE
         FILSLI(I,J) = 1.0
        ENDIF
2  CONTINUE
3 CONTINUE
C  NOW SET LOW NYQ (-KNYQ) TO HIGH NYQ (+KNYQ)
      DO 140 I = 1, NF
       FILSLI(I,1) = FILSLI(I,NK)
140 CONTINUE
      RETURN
      END

c-----------------------------------------------------------------------
C  Subroutine to write out data to disc file
C
c This version (Jan 92) allows interpolation in depth
c Parameter NS,NC are set to the final nsams,nchans after interp
c IF loop flagged 1111 must be altered as well
c
c Jan 93 - ICSCRG option will only work for header out of file/direct
c

      SUBROUTINE WRFILA(NSAM, NCHR, NSAMR, NCHRR,
     1 R4DAT, NFIRST, NRECS,RECDEP, DBGAIN, GCMSCL, NCR,
     2 NSHOT, DT, LEN, SORPOS,NPROCS, IDPROC, OPDISC,
     3 IHFORM, OPFORM, A1, A2, A3, A4, A5, A6, A7, A8, A9, PROC,
     4 NTEMP, R1TEMP, IHEAD, ICSCRG, IBAT)
      PARAMETER (NS=1024,NC=32)
      DIMENSION R4DAT(NSAMR,NCHRR), NFIRST(NCHRR),  A(9)
      DIMENSION DBGAIN(NCHRR), GCMSCL(NCHRR), NCR(2)
      DIMENSION IDPROC(5,NCHRR),RECDEP(NCHRR)
      DIMENSION NTEMP(NCHRR), RDAT(NS,NC), R2TEMP(NC)
      DIMENSION R1TEMP(NCHRR)
      CHARACTER OPFORM*6, IHFORM*4
      CHARACTER*20 A1, A2, A3, A4, A5, A6, A7, A8, A9, A
      CHARACTER*25 PROC(20), OPDISC, F1*50, F2*50, OP1
      IF (IBAT.NE.1) WRITE (6,125)
125 FORMAT (/'  Interpolate by power of 2? (1=y)')
      READ (5,*) IINT
      IF (IINT.EQ.1) THEN
```

```
      FNYQ = 1./(2.*DT/1000000.)
1000  IF (IBAT.NE.1) THEN
       WRITE(6,500)
       WRITE(6,501)
       WRITE(6,502)INTERP
       WRITE(6,503)
      ENDIF
500   FORMAT(' 0           RETURN')
501   FORMAT(' 1           OK GO !')
502 FORMAT(/,' 2)  1=Interpolate first dimension of array (time):',I4)
503   FORMAT(/,'    0=Interpolate 2nd   dimension af array (dist)')
      READ*,IOPT
      IF(IOPT.EQ.0) RETURN
      IF(IOPT.EQ.1) GO TO 1111
      IF(IOPT.EQ.2) READ*,INTERP
      GO TO 1000
1111  IF(INTERP.EQ.1)THEN
       NSF=512
       NSR=1024
      ELSE
       NSF=16
       NSR=32
      ENDIF
      IF (IBAT.NE.1) THEN
       WRITE(6,700)
       WRITE(6,701)
       WRITE(6,702)NSF
       WRITE(6,703)NSR
       WRITE(6,704)
       WRITE(6,705)
       WRITE(6,707)FNYQ
      ENDIF
700   FORMAT(' 0           RETURN')
701   FORMAT(' 1           OK GO !')
702   FORMAT(/,' 2)  NS for forward transform :',I4)
703   FORMAT(/,' 3)  NS for reverse transform :',I4)
704   FORMAT(/,' Any other transformation requires changing the')
705   FORMAT(,'  parameter statement in wrfila.f')
707   FORMAT(/,' Nyquist frequency = ',F8.1)
      READ*,IOPT
      IF(IOPT.EQ.0) GO TO 1000
      IF(IOPT.EQ.1) GO TO 9999
      IF(IOPT.EQ.2) READ*,NSF
      IF(IOPT.EQ.3) READ*,NSR
      GO TO 1111
C..............................................................
9999  CONTINUE
      IF(INTERP.EQ.1)THEN
       DO 123 JJ = 1, NCHR
        CALL FINT2(NSF,NSR,R4DAT(1,JJ),RDAT(1,JJ),
     1  NSAMR,1024)
123    CONTINUE
       DT = DT * NSF/NSR
       NSAM = NSAM * NSR/NSF
       IF (NSAM.LT.1024) THEN
        LEN = 4104
       ELSE
        LEN = (NSAM + 2) * 4
       ENDIF
       DO 8100 JJ = 1, NCHR
        NFIRST(JJ) = ANINT(REAL(NFIRST(JJ)*NSAMIT/NSAMFT))
8100   CONTINUE
      ELSE
       DO 923 II = 1,NSAM
        DO 124 JJ = 1, NCHRR
         R1TEMP(JJ)=R4DAT(II,JJ)
124     CONTINUE
        CALL FINT2(NSF,NSR,R1TEMP,R2TEMP,NCHRR,NSR)
        DO 126 JJ = 1, NC
         RDAT(II,JJ)=R2TEMP(JJ)
126     CONTINUE
923    CONTINUE
       NCHR = NCHR * NSR/NSF
       PRINT*,'NCHR=',NCHR,'CHANGE TO?'
       READ(5,*) NCHR
      ENDIF
      ENDIF
      IF (IBAT.NE.1) PRINT *, 'WRITING ......'
      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
      IDCODE = (NSHOT-1)*(NCHR+1)+1
      IF (OPFORM .EQ. 'ASCII') THEN
        OPEN(11,FILE=OPDISC,STATUS='UNKNOWN',FORM=
     1 'FORMATTED')
        WRITE(11,901) IDCODE
901     FORMAT(I4)
        WRITE (11,902) A, SORPOS, NCHR, RECDEP, DBGAIN,
     1    GCMSCL,NFIRST, NCR, NPROCS, IDPROC, DT
C FORMAT needs changing for different number of channels
902   FORMAT(9A20,F6.2,I2,51F6.2,51F6.2,51F6.2,51I4,2I2,I2,
     1   255I4,F6.2)
       DO 20 J = 1, NCHR
        WRITE(11,903) (R4DAT(I,J),I=1,NSAM)
903     FORMAT(6(F13.7))
20     CONTINUE
      ELSE IF (OPFORM .EQ. 'DIRECT') THEN
C..............................................................
       IF (IHFORM .EQ. 'IN') THEN
        IF (IBAT.NE.1) PRINT *, IDCODE, LEN
        OP1 = '/ts/global/psr/'//OPDISC
        OPEN (11,FILE=OP1,STATUS='UNKNOWN',FORM=
     1     'UNFORMATTED',ACCESS='DIRECT',RECL=LEN)
C OUTPUT PROCESSING AND DISPLAY PARAMETERS
        IF (IHEAD.EQ.0) THEN
         WRITE (11,REC=IDCODE) A, SORPOS, NCHR, RECDEP,
     1    DBGAIN,GCMSCL, NFIRST, NCR, NPROCS, IDPROC, DT
        ELSE
C for use if header is too long
         WRITE (11,REC=IDCODE) A, SORPOS, NCHR
        ENDIF
C WRITE SEISMOGRAM RECORDS
        DO 10 J = 1, NCHR
          NREC = IDCODE + J
          IF (IINT.EQ.1) THEN
           WRITE (11,REC=NREC) (RDAT(I,J),I=1,NSAM)
          ELSE
           WRITE (11,REC=NREC) (R4DAT(I,J),I=1,NSAM)
          ENDIF
10      CONTINUE
C..............................................................
       ELSE IF (IHFORM .EQ. 'OUT') THEN
        LEN1 = (NSAM + 2) * 4
        IDCODE = NSHOT
        IDCODE1 = (NSHOT-1)*NCHR
        IF (IBAT.NE.1) PRINT *, IDCODE, LEN1
        F1 = '/ts/global/psr/'//OPDISC
        F2 = '/ts/global/psr/'//OPDISC
        F1 = F1(1:lnb(F1))//'.d'
        F2 = F2(1:lnb(F2))//'.h'
        OPEN (11,FILE=F1,STATUS='UNKNOWN',FORM=
     1     'UNFORMATTED',ACCESS='DIRECT',RECL=LEN1)
        OPEN (12,FILE=F2,STATUS='UNKNOWN',FORM=
     1     'UNFORMATTED',ACCESS='DIRECT',RECL=LEN1)
C OUTPUT PROCESSING AND DISPLAY PARAMETERS
       IF (IHEAD.EQ.0) THEN
         WRITE (12,REC=IDCODE) A,NSAM,NCHR,DT,NFIRST,
     1    SORPOS,RECDEP
```

```
      ELSE
C for use if header is too long
        WRITE (12,REC=IDCODE) A, SORPOS, NCHR
      ENDIF
C  WRITE SEISMOGRAM RECORDS
        DO 11 J = 1, NCHR
          NREC = IDCODE1 + J
          IF (ICSCRG.EQ.1) NREC = (J-1)*NCHR + NSHOT
          IF (IINT.EQ.1) THEN
            WRITE (11,REC=NREC) (RDAT(I,J),I=1,NSAM)
          ELSE
            WRITE (11,REC=NREC) (R4DAT(I,J),I=1,NSAM)
          ENDIF
 11     CONTINUE
        CLOSE (12)
      ENDIF
      END IF
      CLOSE (11)
      RETURN
      END

c-----------------------------------------------------------------------
      SUBROUTINE WVSHAP(NSAM,NCHR,NSAMR,NCHRR,
     1        R4DAT,R2DAT,NFILES,DT,NFIRST,NSHOT,IBAT)
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 seies
C  and as amplitude spectra.
c  To disable plotting - comment out all WVSHPL
c  NS1 is NSAMR, NS2 IS LIKELY SIZE OF I/O WAVELETS
c  the wavelets are put at NS2/2
c  eg NS1=512,NS2=256,wavelets start at 129
      PARAMETER (NS1=512,NS2=512)
c       PARAMETER (NS1=1024,NS2=512)
c       PARAMETER (NS1=1024,NS2=1024)
      DIMENSION R4DAT(NSAMR,NCHRR)
      DIMENSION R2DAT(NSAMR,NCHRR),NFIRST(NCHRR)
      REAL WKNOW(NS1), ACF(NS2), BUTT(NS2), BUTT2(NS2)
      REAL S1(NS1), S2(NS1), S3(NS1), S4(NS1)
      REAL A(NS1), B(NS1), C(NS1), D(NS1), ER(NS2)
      REAL PHMIN(NS1,3), PH(NS1,4), AMP(NS1), PHTEMP(NS1)
      REAL TEMP3(NS2), TEMP(NS1), TEMP2(NS1), TACBU(NS2,2)
      CHARACTER F4*18, AINV*1
c Arrays purely for plotting
      REAL FDAT(NS1,5), P1(NS1,4), TEMP1(NS1,4)
      COMPLEX CBUTT(NS2), CSX(NS1),CTEMP(NS1)
      COMPLEX CXA(NS1), CXB(NS1), CXC(NS1), CXD(NS1)
      INTEGER NDO(256)
      FNYQ = .5E6 / DT
      DF = FNYQ / REAL(NS2/2)
      IKEEP = 0
      RFC = 0.02
C defaults for plotting
      S = -REAL(NS2)/2
      R = REAL(NS2)/16
 20 IF (IBAT.NE.1) THEN
      WRITE (6,*)
      WRITE (6,*)'         WAVESHAPING FILTER MENU     (0 to
EXIT)'
      WRITE (6,*)'         ~~~~~~~~~~~ ~~~~~~ ~~~~               '
      WRITE (6,*)
      WRITE (6,30)
 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 (6,61)
 61 FORMAT (/' 5   Turn known wavelet into min/max-phase')
      WRITE (6,62)
 62 FORMAT (/' 6   Use Eds predictive decon')
      WRITE (6,63)
 63 FORMAT (/'     (only works for minph i/p)')
      WRITE (6,70)
 70 FORMAT (/' 0   Exit ')
      ENDIF
      READ (5,*) IANS
      IF (IANS .EQ. 1) THEN
 80   IF (IBAT.NE.1) THEN
        WRITE (6,*)
        WRITE (6,*)'          "Input" wavelet selection '
        WRITE (6,*)'          ~~~~~~~~~~~~~~~~~~~~~~~~~ '
        WRITE (6,90)
 90     FORMAT (/' 1      Window of single trace ')
        WRITE (6,100)
 100    FORMAT (/' 2      Extract Minimum phase wavelet ')
        WRITE (6,110)
 110    FORMAT (/' 3      Plot input wavelet ')
        WRITE (6,111)
 111    FORMAT (/' 4      Min-phase window of single trace ')
        WRITE (6,112)
 112    FORMAT (/' 5      Use prev calc filter as input ')
        WRITE (6,120)
 120    FORMAT (/' 0      Exit ')
      ENDIF
        READ (5,*) IOPT
        IF (IOPT.EQ.1.OR.IOPT.EQ.2.OR.IOPT.EQ.4) THEN
          IF (NFILES .EQ. 2) THEN
            IF (IBAT.NE.1) WRITE (6,*)' Enter file number 1 or 2:'
            READ (5,*) I1OR2
          ELSE
            I1OR2 = 1
          ENDIF
        ENDIF
        IF (IOPT .EQ. 1) THEN
          IF (IBAT.NE.1) WRITE (6,*)' Enter trace number :'
          READ (5,*) NTRACE
          IF (IBAT.NE.1) THEN
            WRITE (6,*)' ENTER window on trace in samples :'
            WRITE (6,*)'0,0 = FB->NSAMR 0,-1 = FB->FB+NS2/2'
            WRITE (6,*)'0,no = FB,FB+no:'
          ENDIF
          READ (5,*) NWT1, NWT2
          IF (NWT1.EQ.0) THEN
            IF (NWT2.EQ.0) THEN
              NWT1 = NFIRST(NTRACE)
              NWT2 = NSAMR
            ELSEIF (NWT2.EQ.-1) THEN
              NWT1 = NFIRST(NTRACE)
              NWT2 = NWT1+NS2/2
            ELSE
              NWT1 = NFIRST(NTRACE)
              NWT2 = NWT1+NWT2
            ENDIF
          ENDIF
          NWTDIF = NWT2 - NWT1 + 1
          CALL ZERO(NS2, ACF)
          IF (I1OR2.EQ.1) THEN
            DO 130 I = 1, NWTDIF
              ACF(I + NS2/2) = R4DAT(I + NWT1 - 1,NTRACE)
 130        CONTINUE
          ELSE
            DO 131 I = 1, NWTDIF
              ACF(I + NS2/2) = R2DAT(I + NWT1 - 1,NTRACE)
 131        CONTINUE
          ENDIF
          CALL NORMAN(NS2, ACF)
          LB = NS2 / 2 + NWTDIF
        ELSE IF (IOPT .EQ. 2 .OR. IOPT .EQ. 4) THEN
          IF (IOPT .EQ. 4) THEN
            IREP=1
```

```
      NAV=1
      IF (IBAT.NE.1) WRITE (6,*)'Enter trace number :'
      READ (5,*) NCHANN
      IF (IBAT.NE.1) THEN
       WRITE (6,*)'ENTER  window on traces :'
       WRITE (6,*)'0,0 = FB->NSAMR 0,-1 = FB->FB+NS2/2'
       WRITE (6,*)'0,no = FB,FB+no:'
      ENDIF
      READ (5,*) IW1, IW2
      IF (IW1.EQ.0) THEN
       IF (IW2.EQ.0) THEN
        IW1 = NFIRST(NCHANN)
        IW2 = NSAMR
       ELSEIF (IW2.EQ.-1) THEN
        IW1 = NFIRST(NCHANN)
        IW2 = IW1+NS2/2
       ELSE
        IW1 = NFIRST(NCHANN)
        IW2 = IW1+IW2
       ENDIF
      ENDIF
      ELSE
       IF (IBAT.NE.1) THEN
        WRITE (6,*)'Enter 1=same 0=different window :'
       ENDIF
       READ (5,*) IREP
       IF (IREP .EQ. 1) THEN
        IF (IBAT.NE.1) WRITE (6,*)'ENTER  window on traces :'
        IF (IBAT.NE.1) WRITE (6,*)'window >',NS2/2
        READ (5,*) IW1, IW2
        IF (IBAT.NE.1) WRITE (6,*)'ENTER NO OF CHANNELS :'
        READ (5,*) NAV
        IF (IBAT.NE.1) WRITE (6,*)'ENTER FIRST CHANNEL :'
        READ (5,*) NCHANN
       ELSE
        IF (IBAT.NE.1)
     1   WRITE (6,*)'Average over N traces ENTER N :'
        READ (5,*) NAV
       END IF
      END IF
      CALL ZERO(NS2, ACF)
      CALL ZERO(NS2, WKNOW)
      CALL ZERO(NS2, TEMP2)
      DO 160 J = 1, NAV
       CALL ZERO(NS2, TEMP)
       IF (IREP .NE. 1) THEN
        IF (IBAT.NE.1) WRITE (6,*)'Enter chan no(',J,'):'
        READ (5,*) NCHANN
        IF (IBAT.NE.1) WRITE (6,*)'Enter window on trace:'
        READ (5,*) IW1, IW2
       END IF
       IWTDIF=IW2-IW1+1
       DO 140 I = 1, IWTDIF
        IF (I1OR2 .EQ. 1) THEN
         TEMP(I) = R4DAT(I+IW1-1,NCHANN)
         IF (IOPT .EQ. 4) THEN
          WKNOW(NS2/2+I-1) = R4DAT(I+IW1-1,NCHANN)
c          WKNOW(NS2/2+I+1-IW1) = R4DAT(I,NCHANN)
         ENDIF
        ELSE
         TEMP(I) = R4DAT(I+IW1-1,NCHANN)
         IF (IOPT .EQ. 4) THEN
          WKNOW(NS2/2+I-1) = R2DAT(I+IW1-1,NCHANN)
         ENDIF
        ENDIF
140    CONTINUE
       CALL CROSS(NS2, TEMP, NS2, TEMP, NS2, ACF)
       IF (ACF(1) .NE. 0.0) THEN
        DO 150 I = 1, NS2
         TEMP2(I) = TEMP2(I) + (ACF(I)/ACF(1))
150     CONTINUE
       END IF
       IF (IREP .EQ. 1) NCHANN = NCHANN + 1
160   CONTINUE
C Do minimum phase conversion.
      CALL MINPH(TEMP2, NS2)
      IF (IBAT.NE.1) THEN
       WRITE (6,*)'Enter taper start and end for wavelet :'
       WRITE (6,*) 'No taper = 0,0'
      ENDIF
      READ (5,*) NWT1, NWT2
      LB = NS2/2
      IF (NWT1 .NE. 0 .AND. NWT2 .NE. 0) THEN
       LB = NWT2
       CALL LINTAP(NS2, TEMP2, 0, 0, NWT1, NWT2)
      ENDIF
C Shift so origin is at sample NS2/2
      DO 170 I = 1, NS2 / 2
       ACF(I + NS2/2) = TEMP2(I)
       ACF(I) = TEMP2(NS2/2 + I)
170   CONTINUE
C NORMALISE INPUT WAVELET
      CALL NORMAN(NS2, ACF)
      LB = LB + NS2 / 2
     ELSE IF (IOPT .EQ. 5) THEN
      NWTDIF = NS2/4
      IF (IBAT.NE.1) PRINT*,'Using filter length :',NWTDIF
      CALL ZERO(NS2, ACF)
       DO 171 I = 1, NWTDIF
        ACF(I + NS2/2) = A(I)
171    CONTINUE
      CALL NORMAN(NS2, ACF)
      CALL ZERO(NS1, A)
      LB = NS2 / 2 + NWTDIF
     ELSE IF (IOPT .EQ. 3) THEN
       CALL WVSHPL(NS2,1,NS2,ACF,S,R,R,
     1 ' input wavelet ',' sample number ')
     END IF
     IF (IOPT .NE. 0) THEN
       GO TO 80
     END IF
    ELSE IF (IANS .EQ. 2) THEN
180   IF (IBAT.NE.1) THEN
      WRITE (6,*)'         Desired output wavelet '
      WRITE (6,*)'         ~~~~~~~~~~~~~~~~~~~~~~ '
      WRITE (6,190)
190   FORMAT (/,' 1    Butterworth wavelet (ZP)')
      WRITE (6,200)
200   FORMAT (/,' 2    Trace in record  ')
      WRITE (6,210)
210   FORMAT (/,' 3    Plot output wavelet ')
      WRITE (6,220)
220   FORMAT (/,' 4    MINIMUM PHASE extracted wavelet ')
      WRITE (6,221) IKEEP
221   FORMAT (/,' 5    O/P amp spec = I/P (1-y):',I2)
      WRITE (6,229)
229   FORMAT ('     (select this first, then 1,4 or 6)')
      WRITE (6,222)
222   FORMAT (/,' 6    Add two Butterworths together')
      WRITE (6,223) RFC
223   FORMAT (/,' 7    White noise =',F5.3)
      WRITE (6,224) AINV
224   FORMAT (/,' 8    Apply inverse of the filter :',A1)
      WRITE (6,*)'    (this remembers until you leave xhr1!!)'
      WRITE (6,225)
225   FORMAT (/,' 9    Spike at t=0')
      WRITE (6,226)
226   FORMAT (/,' 10   Print out Butterworth to screen')
      WRITE (6,227)
227   FORMAT (/,' 11   Plot i/p and o/p wavelet')
      WRITE (6,230)
230   FORMAT (/,' 0    EXIT')
      ENDIF
      READ (5,*) IOPT
```

```
      IF (IOPT.EQ.2.OR.IOPT.EQ.4.AND.IKEEP.NE.1) THEN
       IF (NFILES .EQ. 2) THEN
        IF (IBAT.NE.1) WRITE (6,*)' Enter file number 1 or 2:'
        READ (5,*) I1OR2
       ELSE
        I1OR2 = 1
       ENDIF
      ENDIF
231    IF (IOPT.EQ.1.OR.IOPT.EQ.4.OR.IOPT.EQ.6) THEN
       IF(IKEEP.EQ.1) THEN
C Same amp spectrum as wavelet
        DO 3826 J = 1,NS2
         CTEMP(J) = CMPLX(ACF(J),0.0)
3826     CONTINUE
        CALL FFT(NS2,CTEMP,-1.)
        DO 6548 J = 1,NS2
         BUTT(J) = CABS(CTEMP(J))
6548     CONTINUE
        GOTO 271
       ELSEIF (IKEEP.NE.1.AND.IOPT.EQ.1.OR.IOPT.EQ.6) THEN
        IF (IBAT.NE.1)
     1    WRITE (6,*)'Enter lo-cut,slope,hi-cut,slope'
        READ (5,*) BUT1, BUT2, BUT3, BUT4
C HIGH CUT
        RNL = ALOG10((2.*(10.**(BUT4/10.))) - 1.0)
        IF (IBAT.NE.1) WRITE (6,*)'RNL = ', RNL
        RNL = RNL / (2.*ALOG10(2.))
        IF (IBAT.NE.1) WRITE (6,*)'RNL = ', RNL
        DO 240 J = 1, NS2 / 2 + 1
         RFR = DF * REAL(J - 1)
         TEM = 1. / (1. + ((RFR/BUT3)**(2.*RNL)))
         BUTT(J) = SQRT(TEM)
240      CONTINUE
C LOW CUT
        IF (IBAT.NE.1) WRITE (6,*)'NOW LOW CUT.. '
        RNL = ALOG10((2.*(10.**(BUT2/10.))) - 1.0)
        IF (IBAT.NE.1) WRITE (6,*)'RNL = ', RNL
        RNL = RNL / (2.*ALOG10(2.))
        IF (IBAT.NE.1) WRITE (6,*)'RNL = ', RNL
        BUTT2(1) = 0.0
        DO 250 J = 2, NS2 / 2 + 1
         RFR = DF * REAL(J - 1)
         TEM = 1. / (1. + ((BUT1/RFR)**(2.*RNL)))
         BUTT2(J) = SQRT(TEM)
250      CONTINUE
        DO 260 J = 1, NS2 / 2 + 1
         BUTT(J) = BUTT(J) * BUTT2(J)
260      CONTINUE
        IF (IOPT.EQ.6 .AND. NOPT6 .EQ. 0) THEN
         CALL ZERO(NS2,TEMP3)
         DO 261 J = 1, NS2 / 2 + 1
          TEMP3(J) = BUTT(J)
261       CONTINUE
         NOPT6 = 1
         GOTO 231
        ELSE IF (IOPT.EQ.6 .AND. NOPT6.EQ.1) THEN
         DO 262 J = 1, NS2 / 2 + 1
          BUTT(J) = TEMP3(J) + BUTT(J)
262       CONTINUE
         NOPT6 = 0
        ENDIF
C TRANSFORM TO TIME DOMAIN
       ENDIF
       ISAM = 2
       DO 270 I = NS2, NS2 / 2 + 2, -1
        BUTT(I) = BUTT(ISAM)
        ISAM = ISAM + 1
270     CONTINUE
271     DO 280 I = 1, NS2
        CBUTT(I) = CMPLX(BUTT(I),0.0)
280     CONTINUE
       CALL FFT(NS2, CBUTT, 1.)
C Shift origin to NS2/2
       DO 290 I = 1, NS2 / 2
        BUTT2(I + NS2/2) = REAL(CBUTT(I))
        BUTT2(I) = REAL(CBUTT(I + NS2/2))
290     CONTINUE
       LD = NS2
       IF (IOPT.EQ.2) THEN
C Normalise output wavelet
        CALL NORMAN(NS2, BUTT2)
        LD = NS2
       ENDIF
331     IF (IOPT .EQ. 4) THEN
        IF (IKEEP .EQ. 1) THEN
C Convert desired output to min phase if required Shift desired
C output in order to compute the correct autocorrelation
         CALL ZERO(NS2,TEMP3)
         DO 1469 J = 1,NS2/2
          TEMP3(J) = BUTT2(NS2-NS2/2+J)
1469      CONTINUE
         DO 1470 J = NS2/2+1,NS2
          TEMP3(J) = BUTT2(J-NS2/2)
1470      CONTINUE
         DO 1471 J = 1,NS2
          BUTT2(J) = TEMP3(J)
1471      CONTINUE
C     shift done
         CALL CROSS(NS2,BUTT2,NS2,BUTT2,NS2,TEMP3)
         CALL MINPH(TEMP3,NS2)
         CALL ZERO(NS2,BUTT2)
         IF (IBAT.NE.1) PRINT*,' Enter taper in samples '
         READ*,NN1,NN2
         CALL LINTAP(NS2,TEMP3,0,0,NN1,NN2)
C Shift so origin is at sample NS2/2
         DO 5498 I = 1, NS2 / 2
          BUTT2(I + NS2/2) = TEMP3(I)
          BUTT2(I) = TEMP3(NS2/2 + I)
5498      CONTINUE
        ELSE
         IF (IBAT.NE.1) THEN
         WRITE (6,*)'Enter 1=same window 0=selection for chans'
         ENDIF
         READ (5,*) IREP
         IF (IREP .EQ. 1) THEN
          IF (IBAT.NE.1) WRITE (6,*)'ENTER  window on traces :'
          IF (IBAT.NE.1) WRITE (6,*)' 0,0 = FB->NSAMR :'
          READ (5,*) IW1, IW2
          IF (IW1.EQ.0.AND.IW2.EQ.0) THEN
           IW1 = NFIRST(NTRACE)
           IW2 = NSAMR
          ENDIF
          IF (IBAT.NE.1) WRITE (6,*)'ENTER NO OF CHANNELS :'
          READ (5,*) NAV
          IF (IBAT.NE.1) WRITE (6,*)'ENTER FIRST CHANNEL :'
          READ (5,*) NCHANN
         END IF
         CALL ZERO(NS2, TEMP2)
         IF (IREP .NE. 1) THEN
          IF (IBAT.NE.1) THEN
           WRITE (6,*)' Average over N traces ENTER N :'
          ENDIF
          READ (5,*) NAV
         END IF
         DO 360 J = 1, NAV
          CALL ZERO(NS2, TEMP)
          IF (IREP .NE. 1) THEN
           IF (IBAT.NE.1) WRITE (6,*)'Enter chann no (',J,'):'
           READ (5,*) NCHANN
           IF (IBAT.NE.1) WRITE (6,*)'Enter window on trace :'
           READ (5,*) IW1, IW2
          END IF
          IF(I1OR2 .EQ. 1) THEN
           DO 340 I = IW1, IW2
```

```
          TEMP(I) = R4DAT(I,NCHANN)
340        CONTINUE
        ELSE
          DO 341 I = IW1, IW2
            TEMP(I) = R2DAT(I,NCHANN)
341        CONTINUE
        ENDIF
        CALL CROSS(NS2, TEMP, NS2, TEMP, NS2, BUTT2)
        IF (BUTT2(1) .NE. 0.0) THEN
          DO 350 I = 1, NS2
            TEMP2(I) = TEMP2(I) + (BUTT2(I)/BUTT2(1))
350        CONTINUE
        END IF
        IF (IREP .EQ. 1) NCHANN = NCHANN + 1
360     CONTINUE
C Do minimum phase conversion.
        CALL MINPH(TEMP2, NS2)
        IF (IBAT.NE.1)
     1    PRINT*,'Enter taper start/end for wavelet:'
        READ (5,*) NWT1, NWT2
        LD = NWT2
        CALL LINTAP(NS2, TEMP2, 0, 0, NWT1, NWT2)
C Shift so origin is at sample NS2/2
        DO 370 I = 1, NS2 / 2
          BUTT2(I + NS2/2) = TEMP2(I)
          BUTT2(I) = TEMP2(NS2/2 + I)
370      CONTINUE
       ENDIF
C NORMALISE INPUT WAVELET
       CALL NORMAN(NS2, BUTT2)
       LD = LD + NS2 / 2
      ENDIF
     ELSE IF (IOPT .EQ. 2) THEN
      IF (IBAT.NE.1) WRITE (6,*)'Enter trace i.d. number :'
      READ (5,*) NTRACE
      IF (IBAT.NE.1) WRITE (6,*)'Enter taper points (4) :'
      READ (5,*) NTAP0, NTAP1, NTAP2, NTAP3
      IF(I1OR2 .EQ. 1) THEN
       DO 300 I = 1, NS2
         BUTT2(I) = R4DAT(I,NTRACE)
300      CONTINUE
      ELSE
       DO 301 I = 1, NS2
         BUTT2(I) = R2DAT(I,NTRACE)
301      CONTINUE
      ENDIF
     CALL LINTAP(NS2, BUTT2, NTAP0, NTAP1, NTAP2, NTAP3)
      DO 310 I = NTAP0, NTAP3
         BUTT2(I - NTAP0 + NS2/2 + 1) = BUTT2(I)
310     CONTINUE
       DO 320 I = 1, NS2 / 2
         BUTT2(I) = 0.0
320     CONTINUE
       DO 330 I = NTAP3 + 1 + NS2 / 2, NS2
         BUTT2(I) = 0.0
330     CONTINUE
       CALL NORMAN(NS2, BUTT2)
       IF (IBAT.NE.1) WRITE (6,*)'NORMALISED OK'
       LD = NS2
      ELSE IF (IOPT .EQ. 3) THEN
        CALL WVSHPL(NS2,1,NS2,BUTT2,S,R,R,
     1' Desired output ',' sample number ')
      ELSE IF (IOPT .EQ. 11) THEN
        DO 333 I=1,NS2
          TACBU(I,1) = ACF(I)
          TACBU(I,2) = BUTT2(I)
333       CONTINUE
        CALL WVSHPL(NS2,2,NS2,TACBU,S,R,R,
     1' input / output ',' sample number ')
      ELSE IF (IOPT .EQ. 5) THEN
        IF (IKEEP.EQ.0) THEN
          IKEEP = 1
        ELSE
          IKEEP = 0
        ENDIF
      ELSE IF (IOPT .EQ. 7) THEN
       IF (IBAT.NE.1) WRITE (6,*) 'New white noise:'
       READ(5,*) RFC
      ELSE IF (IOPT .EQ. 8) THEN
       IF (AINV.NE.'y') THEN
        AINV = 'y'
       ELSE
        AINV = 'n'
       ENDIF
      ELSE IF (IOPT .EQ. 9) THEN
       CALL ZERO(NS2,BUTT2)
       BUTT2(NS2/2+1) = 1.0
      ELSE IF (IOPT .EQ. 10) THEN
       DO 332 I=1,NS2
         PRINT*,BUTT2(I)
332      CONTINUE
      END IF
       IF (IOPT .NE. 0) GO TO 180
     ELSE IF (IANS .EQ. 3) THEN
      IF (IBAT.NE.1) THEN
       WRITE (6,*)
         WRITE (6,*)'     Calculating filter'
         WRITE (6,*)'     ~~~~~~~~~~~~~~~~~~'
C Calculating the filter coefficients
C INPUT                        = B(NSAMR)
C DESIRED OUTPUT               = D(NSAMR)
C OUTPUT SHIFTED BY LAG        = C(NSAMR)
C FILTER                       = A(NSAMR)
C PREDICTION ERROR OPERATOR    = S3(NSAMR)
C ACF OF INPUT                 = S1(NSAMR)
C CCF OF INPUT AND DESIRED OUTPUT = S2(NSAMR)
         WRITE (6,*)'LD = ', LD
         WRITE (6,*)'LB = ', LB
        ENDIF
         LC = LD
         IF (IBAT.NE.1) THEN
          WRITE (6,*)'Enter filter length in samples :'
          READ (5,*) LA
          WRITE (6,*)'Enter no. of lags to try :'
          READ (5,*) NL
         ELSE
          READ (5,*) LA
          READ (5,*) NL
         ENDIF
         IF (IBAT.NE.1) WRITE (6,*)'Enter 1st lag to try:'
         IF (IBAT.NE.1) WRITE (6,*)'(-ve == OP leads IP)'
         READ (5,*) LAG1
         DO 380 I = 1, NS2
           B(I) = ACF(I)
C INPUT = B OUTPUT= D
           D(I) = BUTT2(I)
380     CONTINUE
C Get ACF of input trace ---> S1 LA is controlling length
         CALL CROSS(LB, B, LB, B, LA, S1)
         S1(1) = S1(1) * (1. + RFC)
C Loop over specified range of lags
          DO 480 L = 1, NL
           LAG = LAG1 + (L - 1)
           IF (IBAT.NE.1) WRITE (6,*)'LAG NO. :', LAG
C Move output shifted by desired lag to C
           DO 390 J = 1, LAG
             C(J) = D(LD - LAG + J)
390          CONTINUE
           DO 400 J = LAG + 1, LD
             C(J) = D(J - LAG)
400          CONTINUE
C GET XCF OF B & Desired OP  LA is controlling length
           CALL CROSS(LC, C, LB, B, LA, S2)
C Solve Normal equations
```

```
      CALL EUREKA(LA, S1, S2, A, S3)
C TRANSFORM FILTER AND INPUT
      IF (IBAT.NE.1) WRITE (6,*)'LA = ', LA
      DO 410 I = LA + 1, NS1
        A(I) = 0.0
 410    CONTINUE
      DO 430 I = 1, NS1
        CXA(I) = CMPLX(A(I),0.0)
        CXB(I) = CMPLX(B(I),0.0)
 430    CONTINUE
      CALL FFT(NS1, CXA, -1.0)
      CALL FFT(NS1, CXB, -1.0)
C Test filter
      DO 450 I = 1, NS1
        CTEMP(I) = CXB(I) * CXA(I)
 450    CONTINUE
      CALL FFT(NS1, CTEMP, +1.0)
      DO 460 I = 1, NS1
        TEMP(I) = REAL(CTEMP(I))
 460    CONTINUE
C Calculate ERROR
      ER(L) = 0.0
      CALL NORMAN(NS1, C)
      CALL NORMAN(NS1, TEMP)
      DO 470 I = 1, NS1
        ER(L) = ER(L) + ((C(I) - TEMP(I))**2)
 470    CONTINUE
      ER(L) = SQRT(ER(L)/REAL(NS1))
 480  CONTINUE
     IF (IBAT.NE.1)
   1  WRITE(6,*)'MEAN SQUARE ENERGIES:',(ER(I),I=1,NL)
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
     IF (IBAT.NE.1) THEN
       WRITE (6,*)'OPTIMUM LAG :', NLAGOP, '(', IND, ')'
       WRITE (6,*)'Minimum energy : ', ERMIN
     ENDIF
     CALL ZERO(NS1, C)
     DO 500 J = 1, NLAGOP
       C(J) = D(LD - NLAGOP + J)
 500  CONTINUE
      DO 510 J = NLAGOP + 1, LD
        C(J) = D(J - NLAGOP)
 510  CONTINUE
C Get ACF of B
      CALL ZERO(NS1, S1)
      CALL CROSS(LB, B, LB, B, LA, S1)
      S1(1) = S1(1) * (1. + RFC)
C GET XCF OF B & Desired OP
      CALL ZERO(NS1, S2)
      CALL CROSS(LC, C, LB, B, LA, S2)
C Solve Normal equations
      CALL EUREKA(LA, S1, S2, A, S3)
C TRANSFORM FILTER AND INPUT
      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 FFT(NS1, CXA, -1.0)
      CALL FFT(NS1, CXB, -1.0)
      CALL FFT(NS1, CXC, -1.0)
      CALL FFT(NS1, CXD, -1.0)
C Test filter
      DO 530 I = 1, NS1
        CTEMP(I) = CXB(I) * CXA(I)
 530  CONTINUE
      CALL FFT(NS1, CTEMP, +1.0)
      DO 540 I = 1, NS1
        TEMP(I) = REAL(CTEMP(I))
 540  CONTINUE
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 = SQRT(ERMIN/REAL(NS1))
 560  CONTINUE
     IF (IBAT.NE.1) WRITE (6,*)'MEAN SQUARE ENERGY:', ERMIN
      DO 580 I = 1, NS2 / 2
        S4(I + NS2/2) = A(I)
        S4(I) = A(I + NS2/2)
 580  CONTINUE
      IF (IBAT.NE.1) WRITE (6,*) '4 or 5 trace plot?'
      IF (IBAT.NE.1) READ(5,*) I4OR5
      DO 581 I=1,NS2
        FDAT(I,1)=B(I)
        FDAT(I,2)=D(I)
        FDAT(I,3)=S4(I)
        IF(I4OR5.EQ.5) FDAT(I,4)=C(I)
        FDAT(I,I4OR5)=TEMP(I)
 581  CONTINUE
      S= 1
      R= REAL(NS2)/8
      IF (IBAT.NE.1) CALL WVSHPL(NS2,I4OR5,NS1,FDAT,S,R,R,
    1'  wavelets   ',' sample number ')
      CALL ZEROC(NS1, CTEMP)
      DO 590 I = 1, NS1
        CTEMP(I) = CXA(I) * CXB(I)
 590  CONTINUE
      NSFT = NS1 / 2 + 1
      DO 600 I = 1, NSFT
        P1(I,1) = CABS(CXB(I))
        P1(I,2) = CABS(CXD(I))
        P1(I,3) = CABS(CXA(I))
        P1(I,4) = CABS(CTEMP(I))
 600  CONTINUE
      DO 601 I = 1, 4
        CALL NORMAN(NSFT, P1(1,I))
 601  CONTINUE
      R1 = 100*REAL(NSFT)/FNYQ
      IF (IBAT.NE.1) CALL WVSHPL(NSFT,4,NS1,P1,0.0,100.0,R1,
    1'    spectra    ','frequency (Hz)')
     ELSE IF (IANS .EQ. 4) THEN
 610  IF (IBAT.NE.1) THEN
      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 '
     ENDIF
     READ (5,*) IOPT
     IF (IOPT .EQ. 1) THEN
      IF (IBAT.NE.1) WRITE (6,*)'1 = apply to all:'
      READ (5,*) IREPL
      IF (IREPL .EQ. 1) THEN
       NAPPLY = NCHR
       DO 620 I = 1, NAPPLY
        NDO(I) = I
 620    CONTINUE
```

```
      ELSE
       IF (IBAT.NE.1) WRITE (6,*)' Enter number of channels to
     1apply filter :'
        READ (5,*) NAPPLY
        DO 630 I = 1, NAPPLY
         IF (IBAT.NE.1) WRITE (6,*)'Enter channel #', I
         READ (5,*) NDO(I)
630     CONTINUE
       END IF
      ELSE IF (IOPT .EQ. 2) THEN
       IF (AINV.EQ.'y') NLAGOP = - NLAGOP
       DO 700 I = 1, NAPPLY
        II = NDO(I)
        CALL RMSERR(NSAM, R4DAT(1,II), RMS0)
        DO 640 J = 1, NSAM
         CXB(J) = CMPLX(R4DAT(J,II))
640     CONTINUE
        DO 650 J = NSAM + 1, NSAMR
         CXB(J) = 0.0
650     CONTINUE
        CALL FFT(NSAMR, CXB, -1.0)
        DO 660 J = 1, NSAMR
         IF (AINV.NE.'y') THEN
          CXB(J) = CXB(J) * CXA(J)
         ELSE
          CXB(J) = CXB(J) / CXA(J)
         ENDIF
660     CONTINUE
        CALL FFT(NSAMR, CXB, +1.0)
        DO 670 J = 1, NLAGOP
         IF (NSAMR - NLAGOP + J .GT. NSAM) GO TO 670
         R4DAT(NSAMR - NLAGOP + J,II) = REAL(CXB(J))
670     CONTINUE
        DO 680 J = NLAGOP + 1, NSAMR
         IF (J - NLAGOP .GT. NSAM) GO TO 680
         R4DAT(J - NLAGOP,II) = REAL(CXB(J))
680     CONTINUE
        CALL RMSERR(NSAM, R4DAT(1,II), RMS1)
        SCALE = RMS0 / RMS1
        DO 690 J = 1, NSAM
         R4DAT(J,II) = R4DAT(J,II) * SCALE
690     CONTINUE
700    CONTINUE
       ENDIF
       IF (IOPT .NE. 0) GO TO 610
      ELSE IF (IANS .EQ. 5) THEN
710    IF (IBAT.NE.1) THEN
       WRITE (6,*)
       WRITE (6,*)'        MIN/MAX-PHASING'
       WRITE (6,*)'        ~~~~~~~~~~~~~~~'
       WRITE (6,*)' Need min phase input wavelet defined'
       WRITE (6,*)' 1  Min-phase wavelet in seismogram  '
       WRITE (6,*)' 2  Max-phase wavelet in seismogram  '
       WRITE (6,*)' 3  Inverse Min-phase '
       WRITE (6,*)' 4  Inverse Max-phase '
       WRITE (6,*)' 5  Min to Zero-phase '
       WRITE (6,*)' 6  Max to Zero-phase '
       WRITE (6,*)' 7  Straight to Zero-phase '
       WRITE (6,*)' 0  EXIT '
       ENDIF
       READ (5,*) IOPT
       IF (IOPT .NE. 0) THEN
        IF (IBAT.NE.1) THEN
         WRITE (6,*)' Which trace for displaying in plots? '
        ENDIF
        READ (5,*) NTRPL
        R2 = 100*REAL(NSAMR)/FNYQ
        DO 718 I = 1, NS2/2
         PHMIN(I+NS/2,2)=WKNOW(I)
         PHMIN(I,2)=WKNOW(NS2/2+I)
         PHMIN(I+NS2/2,3)=ACF(I)
         PHMIN(I,3)=ACF(NS2/2+I)
718      CONTINUE
        DO 720 J = 2, 3
         DO 716 I = 1, NSAMR
          CSX(I) = CMPLX(PHMIN(I,J),0.0)
716       CONTINUE
         CALL FFT(NSAMR, CSX, -1.0)
         CALL POLAR(NSAMR,CSX,AMP,PHTEMP)
         CALL DRUM(NSAMR,PHTEMP)
         DO 715 I = 1, NSAMR
          PH(I,J) = PHTEMP(I)
715       CONTINUE
720      CONTINUE
        DO 719 II = 1, NCHR
         DO 717 I = 1, NSAM
          PHMIN(I,1)=R4DAT(I,II)
          IF (II.EQ.NTRPL) TEMP1(I,2) = R4DAT(I,II)
717       CONTINUE
         DO 730 I = 1, NSAMR
          CSX(I) = CMPLX(PHMIN(I,1),0.0)
730       CONTINUE
         CALL FFT(NSAMR, CSX, -1.0)
         CALL POLAR(NSAMR,CSX,AMP,PHTEMP)
         CALL DRUM(NSAMR,PHTEMP)
         DO 750 I = 1, NSAMR
          PH(I,1) = PHTEMP(I)
          IF (IOPT .EQ. 1) THEN
           PH(I,4)=PH(I,1)-PH(I,2)+PH(I,3)
          ELSEIF (IOPT .EQ. 2) THEN
           PH(I,4)=PH(I,1)-PH(I,2)-PH(I,3)
          ELSEIF (IOPT .EQ. 3) THEN
           PH(I,4)=PH(I,1)+PH(I,2)-PH(I,3)
          ELSEIF (IOPT .EQ. 4) THEN
           PH(I,4)=PH(I,1)+PH(I,2)+PH(I,3)
          ELSEIF (IOPT .EQ. 5) THEN
           PH(I,4)=PH(I,1)-PH(I,3)
          ELSEIF (IOPT .EQ. 6) THEN
           PH(I,4)=PH(I,1)+PH(I,3)
          ELSEIF (IOPT .EQ. 7) THEN
           PH(I,4)=PH(I,1)-PH(I,2)
          ENDIF
750       CONTINUE
         CALL POLAR1(NSAMR,CSX,AMP,PH(1,4))
         CALL FFT(NSAMR, CSX, 1.0)
         DO 760 I = 1, NSAM
           R4DAT(I,II) = REAL(CSX(I))
760       CONTINUE
719      CONTINUE
        DO 761 I=1,NS1
         DO 762 II=2,3
          TEMP1(I,II+1) = PHMIN(I,II)
762       CONTINUE
         TEMP1(I,1) = R4DAT(I,NTRPL)
761      CONTINUE
        IF (IBAT.NE.1) CALL WVSHPL(NSAMR,4,NS1,TEMP1,
     1  S,R,R,F4,' sample number ')
        GO TO 20
       END IF
      ELSE IF (IANS .EQ. 6) THEN
       IF (IBAT.NE.1) WRITE (6,770)
770   FORMAT('Optimum-lag spiking with filter as input (1=y)?')
       READ (5,*) IPRO
       IF (IPRO.EQ.1) THEN
        NCPRO = 1
        CALL ZERO(NS2,TEMP)
        CALL PRO17(NS2,NCPRO,NS2,NCPRO,A,TEMP)
C Test filter
        DO 769 I = 1, NS1
          CTEMP(I) = CMPLX(TEMP(I),0.0)
769     CONTINUE
        CALL FFT(NS1, CTEMP, -1.0)
        DO 768 I = 1, NS1
          CTEMP(I) = CTEMP(I) * CXA(I)
```

```
768   CONTINUE
      CALL FFT(NS1, CTEMP, +1.0)
      DO 767 I = 1, NS1
        TEMP2(I) = REAL(CTEMP(I))
767   CONTINUE
      DO 772 I=1,NS2/2
       FDAT(I + NS2/2,1)=A(I)
       FDAT(I,1)=A(I + NS2/2)
       FDAT(I + NS2/2,2)=TEMP(I)
       FDAT(I,2)=TEMP(I + NS2/2)
       FDAT(I + NS2/2,3)=TEMP2(I)
       FDAT(I,3)=TEMP2(I + NS2/2)
772   CONTINUE
      S= 1
      R= REAL(NS2)/8
      IF (IBAT.NE.1) CALL WVSHPL(NS2,3,NS1,FDAT,S,R,R,
     1'   wavelets   ',' sample number ')
C put inverse filter into filter array
      DO 771 I = 1,512
       A(I) = TEMP(I)
       CXA(I) = CMPLX(A(I),0.0)
771   CONTINUE
      CALL FFT(NS1, CXA, -1.0)
     ELSE
      CALL PRO17(NSAM,NCHR,NSAMR,NCHRR,R4DAT)
     ENDIF
    END IF
    IF (IANS .NE. 0) THEN
      GO TO 20
    END IF
    RETURN
    END
```

Appendix C
xhrp.dfault1

Given here are four example **xhrp.dfault1** files required by the **xhr1** program. They are for the four surveys in this study, i.e. Groningen, Physical Model, Lowther South 3437-3436, Lowther South 3500-3496.

Groningen gasfield	I Site name	LOCATE	A20
19-24 Nov 1990	I Acquisition date	DATE	A20
Compu-log	I Seismograph type	DEVICE	A20
Stanford bender	I Source type	SORTYP	A20
X551.55,Y273.09,Z0.0	I Source location	SORLOC	A20
hydrophone	I Receiver type	RECTYP	A20
X553.60,Y133.12,Z0.0	I Receiver location	RECLOC	A20
	I Common shot traces	COMSHT	A20
BANDPASS 350-2000 Hz	I Analogue filters	FILREC	A20
NEW	I Method of data I/P	INPOPT	A3
delft/2350a.dat	I Input disc name	IPDISC	A25
delft/2350fkd	I Input disc name	IPDISC	A25
delft/2350out	I Output disc name	OPDISC	A25
DIRECT	I Input format	IPFORM	A6
DIRECT	I Output format	OPFORM	A6
IN	I Headers in with data	IHFORM	A4
2001	I # of samples/trace	NSAMR	I4
63	I # of channels	NCHR	I3
1	I # of files for I/P	NFILES	I1
1	I ID of first file	NFILE(1)	I3
0	I ID of 2nd file	NFILE(2)	I3
0	I} Common receiver	NCR(1)	I2
	I} channels	NCR(2)	I2
2350.0	I Source depth	SORDEP	F6.1
2233.0	I Top receiver depth	TOPREC	F6.1
3.00	I Receiver separation	RECSEP	F6.2
200.00	I Sampling interval	DT	F6.2
3.00	I Spatial interval met	DZ	F6.2
5.00	I Plot scaling factor	SCAL	F6.2
OFF	I Titles switch	TITOFF	A3
ALL	I Channel plot switch	CHNSPL	A26
NORMAL	I Polarity switch	POLART	A7
OFF	I Variable area swtch	VARARE	A3
300	I 1st plot sample	NPLFST	I4
500	I Last plot sample	NPLLST	I4
1	I Switch for CHNSPL	IOPT	I2
23	I Default trace save	NCHDEF	I2
Groningen CSG - depth 2350m	I Plot title	TITLE	A30
Travel time in milliseconds	I Xaxis label	XLB	A30
Receiver depth in metres	I Yaxis label	YLB	A30

Ultrasonic tank	I Site name	LOCATE	A20
1989	I Acquisition date	DATE	A20
	I Seismograph type	DEVICE	A20
Piezoelectric transducer	I Source type	SORTYP	A20
	I Source location	SORLOC	A20
Piezoelectric transducer	I Receiver type	RECTYP	A20
	I Receiver location	RECLOC	A20
	I Common shot traces	COMSHT	A20
ALLPASS	I Analogue filters	FILREC	A20
NEW	I Method of data I/P	INPOPT	A3
traw	I Input disc name	IPDISC	A25
traw	I Input disc name	IPDISC	A25
tout	I Output disc name	IPFORM	A6
OUT	I Headers in with data	IHFORM	A4
512	I # of samples/trace	NSAMR	I4
51	I # of channels	NCHR	I3
1	I # of files for I/P	NFILES	I1
1	I ID of first file	NFILE(1)	I3
0	I ID of 2nd file	NFILE(2)	I3
0	I} Common receiver	NCR(1)	I2
	I} channels	NCR(2)	I2
0.0	I Source depth	SORDEP	F6.1
0.0	I Top receiver depth	TOPREC	F6.1
2.50	I Receiver separation	RECSEP	F6.2
250.00	I Sampling interval	DT	F6.2
2.50	I Spatial interval met	DZ	F6.2
5.00	I Plot scaling factor	SCAL	F6.2
OFF	I Titles switch	TITOFF	A3
ALL	I Channel plot switch	CHNSPL	A26
NORMAL	I Polarity switch	POLART	A7
OFF	I Variable area swtch	VARARE	A3
20	I 1st plot sample	NPLFST	I4
300	I Last plot sample	NPLLST	I4
1	I Switch for CHNSPL	IOPT	I2
23	I Default trace save	NCHDEF	I2
Tank data	I Plot title	TITLE	A30
Travel time in milliseconds	I Xaxis label	XLB	A30
Receiver depth in metres	I Yaxis label	YLB	A30

```
Lowther South                 ! Site name               LOCATE    A20
11 OCT 1989                   ! Acquisition date        DATE      A20
ES-2401                       ! Seismograph type        DEVICE    A20
no.8 det                      ! Source type             SORTYP    A20
3436                          ! Source location         SORLOC    A20
hydrophone                    ! Receiver type           RECTYP    A20
3437                          ! Receiver location       RECLOC    A20
                              ! Common shot traces      COMSHT    A20
ALLPASS                       ! Analogue filters        FILREC    A20
NEW                           ! Method of data I/P      INPOPT    A3
lows3837                      ! Input disc name         IPDISC    A25
lows3837                      ! Input disc name         IPDISC    A25
lowsintfksh                   ! Output disc name        OPDISC    A25
DIRECT                        ! Input format            IPFORM    A6
DIRECT                        ! Output format           OPFORM    A6
IN                            ! Headers in with data    IHFORM    A4
 512                          ! # of samples/trace      NSAMR     I4
 23                           ! # of channels           NCHR      I3
1                             ! # of files for I/P      NFILES    I1
  1                           ! ID of first file        NFILE(1)  I3
  0                           ! ID of 2nd file          NFILE(2)  I3
 0                            !} Common receiver        NCR(1)    I2
                              !}    channels            NCR(2)    I2
  10.28                       ! Source depth            SORDEP    F6.1
  12.0                        ! Top receiver depth      TOPREC    F6.1
  2.00                        ! Receiver separation     RECSEP    F6.2
200.00                        ! Sampling interval       DT        F6.2
  2.00                        ! Spatial interval met    DZ        F6.2
 10.00                        ! Plot scaling factor     SCAL      F6.2
OFF                           ! Titles switch           TITOFF    A3
ALL                           ! Channel plot switch     CHNSPL    A26
NORMAL                        ! Polarity switch         POLART    A7
OFF                           ! Variable area swtch     VARARE    A3
  20                          ! 1st plot sample         NPLFST    I4
 300                          ! Last plot sample        NPLLST    I4
  1                           ! Switch for CHNSPL       IOPT      I2
 23                           ! Default trace save      NCHDEF    I2
Lowther South CSG             ! Plot title              TITLE     A30
Travel time in milliseconds   ! Xaxis label             XLB       A30
Receiver depth in metres      ! Yaxis label             YLB       A30
```

```
Lowther South                 ! Site name               LOCATE    A20
30 Nov 1989                   ! Acquisition date        DATE      A20
ES-2401                       ! Seismograph type        DEVICE    A20
no.8 det                      ! Source type             SORTYP    A20
3500                          ! Source location         SORLOC    A20
hydrophone                    ! Receiver type           RECTYP    A20
3496                          ! Receiver location       RECLOC    A20
                              ! Common shot traces      COMSHT    A20
ALLPASS                       ! Analogue filters        FILREC    A20
NEW                           ! Method of data I/P      INPOPT    A3
l3500gun                      ! Input disc name         IPDISC    A25
l3500a                        ! Input disc name         IPDISC    A25
l3500out                      ! Output disc name        OPDISC    A25
DIRECT                        ! Input format            IPFORM    A6
DIRECT                        ! Output format           OPFORM    A6
OUT                           ! Headers in with data    IHFORM    A4
 512                          ! # of samples/trace      NSAMR     I4
 23                           ! # of channels           NCHR      I3
1                             ! # of files for I/P      NFILES    I1
  1                           ! ID of first file        NFILE(1)  I3
  0                           ! ID of 2nd file          NFILE(2)  I3
 0                            !} Common receiver        NCR(1)    I2
                              !}    channels            NCR(2)    I2
  10.0                        ! Source depth            SORDEP    F6.1
  9.79                        ! Top receiver depth      TOPREC    F6.1
  2.00                        ! Receiver separation     RECSEP    F6.2
200.00                        ! Sampling interval       DT        F6.2
  2.00                        ! Spatial interval met    DZ        F6.2
  5.00                        ! Plot scaling factor     SCAL      F6.2
OFF                           ! Titles switch           TITOFF    A3
ALL                           ! Channel plot switch     CHNSPL    A26
NORMAL                        ! Polarity switch         POLART    A7
OFF                           ! Variable area swtch     VARARE    A3
  20                          ! 1st plot sample         NPLFST    I4
 300                          ! Last plot sample        NPLLST    I4
  1                           ! Switch for CHNSPL       IOPT      I2
 23                           ! Default trace save      NCHDEF    I2
Lowther South CSG             ! Plot title              TITLE     A30
Travel time in milliseconds   ! Xaxis label             XLB       A30
Receiver depth in metres      ! Yaxis label             YLB       A30
```

Appendix D

xhr3

This program was adapted from **xhr1** to perform 3-D *f-k-k* filtering and plotting. Also given is an example of the **xhr3.dfolt** file required by the program. All subroutines, including plotting, are given. Subroutines are in **libxh3.a** in the dgl3psr user area on the University of Durham Geological Sciences Department's SUN system. Plotting routines use UNIRAS.

FILE - xhr3.dfolt

```
/ts/global/psr/tdec          ! Input disc name          IPDISC  A25
/ts/global/psr/tout          ! Output disc name         OPDISC  A25
OUT                          ! Headers in with data     IHFORM  A4
512                          ! # of samples/trace       NSAM    I4
51                           ! # of sources             NSOR    I3
51                           ! # of receivers           NREC    I3
 0.0                         ! Top receiver depth       TOPSOR  F6.1
 0.0                         ! Top receiver depth       TOPREC  F6.1
 2.50                        ! Rec/Sor  separation      RECSEP  F6.2
250.00                       ! Sampling interval        DT      F6.2
```

```
      PROGRAM XHR3
C
C  Written by P S Rowbotham 1992-1993
C  Adapted from 2d xhr1 package written by M.J.Findlay 1989
C
C  Reads seismic data from disc in variety of formats
C  Dataprocessing software all Menu driven
C        uniras graphics library
C  Requires defaults file xhr3.dfolt.
C  NFK1,NFK2,NFK3 need to be one greater than
C  NSAMR/2,NSORS,NRECS respectively
C
c PARAMETER (NSAMR=512,NSORS=32,NRECS=32,
c  NFK1=257, NFK2=33, NFK3=33,NSNR=1024, NDUM=300)
      PARAMETER (NSAMR=512,NSORS=64,NRECS=64,
     ! NFK1=257,NFK2=65,NFK3=65,NSNR=4096, NDUM=300)
c     PARAMETER (NSAMR=512,NSORS=32,NRECS=32,
c  NFK1=257,NFK2=33, NFK3=33,NSNR=4096, NDUM=300)
c     PARAMETER (NSAMR=512,NSORS=16,NRECS=16,
c  NFK1=257,NFK2=17, NFK3=17,NSNR=256, NDUM=300)
      CHARACTER IPDISC*25, OPDISC*25,IHFORM*4
      CHARACTER*20 LOCATE, DATE, DEVICE, SORTYP, A(9)
      CHARACTER*20 SORLOC, RECTYP, RECLOC,FILREC,
     ! COMSHT
      COMPLEX CPDUM(NSAMR,NSORS,NRECS),CWDUM(NRECS)
      INTEGER NFIRST(NSORS,NRECS), IDPROC(5,NRECS),
     ! NF(NRECS)
      REAL R4DAT(NSAMR,NSORS,NRECS), SORDEP(NSORS)
      REAL RDUM(NSNR), GCMSCL(NRECS),TEMP(NSAMR)
      REAL RTEMP(NFK3,NFK2,NFK1), RDUM1(NFK2,NDUM)
      REAL KKDUM(NFK2,NFK1),AMPDUM(NFK3,NFK1)
      REAL FILDUM(NFK1,NFK2,NFK3), RDAT(NSAMR,NRECS)
      REAL SORPOS, RECDEP(NRECS), DT, DBGAIN(NRECS)
      EQUIVALENCE (LOCATE,A(1)), (DATE,A(2)), (DEVICE,A(3)),
     1        (SORTYP,A(4)), (SORLOC,A(5)), (RECTYP,A(6)),
     2        (RECLOC,A(7)), (COMSHT,A(8)), (FILREC,A(9))
C  Set up initial default values - READ IN FROM DEFAULT FILE
      OPEN(UNIT=1,FILE='xhr3.dfolt')
      READ (1,10) IPDISC, OPDISC, IHFORM,NSAM,NSOR,NREC
   10 FORMAT (A25/A25/A4/I4/I3/I3)
      READ (1,30) TOPSOR, TOPREC, RECSEP, DT
   30 FORMAT (F6.1/F6.1/F6.1/F6.2/F6.2)
      CLOSE (1)
      IF (NSAM.LT.1024) THEN
        LEN = 4104
      ELSE
        LEN = (NSAM + 2) * 4
      ENDIF
      RECDEP(1) = TOPREC
      SORDEP(1) = TOPSOR
      DO 50 I = 2, NREC
        RECDEP(I) = RECDEP(I - 1) + RECSEP
        SORDEP(I) = SORDEP(I - 1) + SORSEP
   50 CONTINUE
      NPROCS = 1
      IDPROC(1,NPROCS) = 1
      IHEAD = 0
      IFNOHD = 0
      IBAT = 0
C  DT in microsecs, SAMFRQ in kHz
   70 IF (IBAT.NE.1) THEN
       PRINT*,'DT=',DT
       WRITE (6,*)
       WRITE (6,*)'              3D XHR MENU  '
       WRITE (6,*)'              ~~~~~~~~~~~            '
       WRITE (6,*)
       WRITE (6,*)'              MASTER MENU    '
       WRITE (6,*)'              ~~~~~~ ~~~~            '
       WRITE (6,*)
       WRITE (6,*)'    1  Read data from file(s)           '
       WRITE (6,*)'    2  Write contents of R4DAT to file    '
       WRITE (6,*)'    3  F-K transform traces            '
       WRITE (6,*)
       WRITE (6,*)'    4  Adjust I/O parameters            '
       WRITE (6,71) IBAT
   71 FORMAT('    5  Batch mode (text off = 1)       :',I1)
       WRITE (6,*)
       WRITE (6,*)'    0  Exit      '
       ENDIF
       READ (5,*) IANSWR
       IF (IANSWR.EQ.1) CALL RDFILE3(NSAM,NSOR,NREC,
     1 NSAMR,NSORS,NRECS,R4DAT,NFIRST, RECDEP,
     2 SORDEP,DBGAIN,GCMSCL,DT,LEN,NF, SORPOS,
     3 NPROCS, IDPROC, IPDISC, IHFORM, IFNOHD, A,
     4   TEMP, IBAT)
      IF (IANSWR .EQ. 2) THEN
        DO 121 KK = 1,NSOR
         DO 122 J = 1,NREC
          NF(J) = NFIRST(KK,J)
          DO 122 I = 1,NSAM
           RDAT(I,J) = R4DAT(I,KK,J)
  122    CONTINUE
         CALL WRFIL3(NSAM, NREC, NSAMR, NRECS, RDAT,
     1 NF, NRECS,RECDEP, DBGAIN, GCMSCL, KK, DT, LEN,
     2 SORPOS, NPROCS, IDPROC, OPDISC, IHFORM, LOCATE,
     3 DATE, DEVICE, SORTYP,SORLOC,RECTYP,RECLOC,
     4 COMSHT,FILREC,DBGAIN,IHEAD,IBAT)
  121  CONTINUE
      ENDIF
      IF (IANSWR .EQ. 3) CALL MENUF3(NSAM, NSOR, NREC,
     1 NSAMR, NSORS, NRECS, NFK1, NFK2, NFK3, R4DAT,
     2 RTEMP, DZ, DZ, DT, NSNR,RDUM,CPDUM,CWDUM,
     3 FILDUM, AMPDUM,RDUM1,KKDUM,NDUM,IBAT,NFIRST)
      IF (IANSWR .EQ. 4) CALL MENUIO3(NSAM, NSOR, NREC,
     1 IPDISC, IHFORM, OPDISC, IHEAD, IFNOHD, IBAT)
      IF (IANSWR .EQ. 4) THEN
       IF (NSAM.LT.1024) THEN
         LEN = 4104
       ELSE
         LEN = (NSAM + 2) * 4
       ENDIF
      ENDIF
      IF (IANSWR .EQ. 5) THEN
       IF (IBAT.EQ.0) THEN
         IBAT = 1
       ELSE
         IBAT = 0
       ENDIF
      ENDIF
      IF (IANSWR .EQ. 0) CALL EXIT(0)
      DZ = ABS(RECDEP(2) - RECDEP(1))
      NFK=NFK1*NFK2*NFK3
      NFKA=NFK1*NFK3
      NFKB=NFK2*NFK3
      CALL ZERO(NFKB, KKDUM)
      CALL ZERO(NDUM, RDUM1)
      CALL ZERO(NSNR, RDUM)
      CALL ZERO(NFK, FILDUM)
      CALL ZERO(NFK, RTEMP)
      CALL ZERO(NFKA, AMPDUM)
```

```
      GO TO 70
      END

C------------------------------------------------------------------
      FUNCTION DIST(X, Y, M, INTERC, MULT)
      REAL DIST, M, INTERC
      DIST = ((Y - M*X - INTERC)/SQRT(1 + M**2)) * MULT
      END
      FUNCTION DIST1(Y, LIM, INTERC, MULT)
      REAL DIST1, LIM, INTERC
C since only looking for +/- have dropped +ve denominator
      DIST1 = (Y - LIM - INTERC) * MULT
      END

C------------------------------------------------------------------
      SUBROUTINE MENUF3(NSAM, NSOR, NREC, NSAMR,
     1 NSORS, NRECS, N1, N3, N5, R4DAT, RTEMP, DSZ, DRZ,
     2 DT, NSNR, TEMP1, CP, CW,  FILSPC, AMPSPC, TEMPLO,
     3 KKDUM, NDUM, IBAT,NFIRST)
C  Subroutine to obtain F-K-K spectrum of recorded data
      DIMENSION R4DAT(NSAMR,NSORS,NRECS), TEMP1(NSNR)
      DIMENSION  RTEMP(N5,N3,N1), FILSPC(N1,N3,N5)
      CHARACTER*1 ANS, ANSWER, KSORR, KORSSL, APONE
      CHARACTER*1 ANP, SQRDAT, LOGYN, YN3D
      CHARACTER FNAME*10
      COMPLEX CP(NSAMR,NSORS,NRECS), CW(NRECS), CTEMP
      REAL KSNYQ,KRNYQ,TEMP,CHTS(8),AMPSPC(N5,N1)
      REAL KKDUM(N3,N5), GAINAP(60), TEMPLO(N1,NDUM)
      INTEGER IDTRGA(60), NFIRST(NSORS,NRECS)
      DATA CHTS / 0.1,0.2,0.5,1.0,2.0,5.0,0,0 /
  490 FORMAT(A1)
      NCHTS = 4
      SQRDAT = 'N'
      ILEG = 1
      PRINT*,IBAT
      NN = N5*N3*N1
      CALL ZERO(NN,RTEMP)
      PRINT*,NSAM, NSOR, NREC, NSAMR, NSORS, NRECS, N1,
     2 N3, N5, DSZ, DRZ, IBAT
      NF = NSAMR / 2 + 1
      NSK = NSORS + 1
      NRK = NRECS + 1
      FNYQ = .5E6 / DT
      KSNYQ = .5 / DSZ
      KRNYQ = .5 / DRZ
      DSK = 2 * KSNYQ / NSORS
      DRK = 2 * KRNYQ / NRECS
      DF = 2 * FNYQ / NSAMR
      IF (IBAT.NE.1)
     1 PRINT *,'KSNYQ,KRNYQ,FNYQ,NSK,NRK,NF,DSZ,DRZ,DT',
     2       KSNYQ,KRNYQ,FNYQ,NSK,NRK,NF,DSZ,DRZ,DT
   10 IF (IBAT.NE.1) THEN
      WRITE (6,*)
      WRITE (6,*)'         F-K Spectra Menu          '
      WRITE (6,*)'         ~~~ ~~~~~~~ ~~~~          '
      WRITE (6,*)
      WRITE (6,20)
   20 FORMAT (/'  1  Calculate 3DFFT            ')
      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) SQRDAT
   80 FORMAT ('  7  Square root of data for plot :',A1)
      WRITE (6,90)
   90 FORMAT ('  8  Spatial taper')
      WRITE (6,100)
  100 FORMAT ('  9  Normalise each trace wrt rms energy  ')
      WRITE (6,101)
  101 FORMAT ('  10  Apply gains to traces 2D ')
      WRITE (6,102)
  102 FORMAT ('  11  Apply mutes to traces ')
      WRITE (6,103)
  103 FORMAT ('  12  Apply gains to traces 1D ')
      WRITE (6,104) ILEG
  104 FORMAT ('  13  Legend included on 3D plot (1=y) ':I1)
      WRITE (6,110)
  110 FORMAT (/'  0  Return to MAIN MENU         ')
      ENDIF
      READ (5,*) IANSWR
      IF (IANSWR .EQ. 0) RETURN
      IF (IANSWR .EQ. 1) THEN
       IF (IBAT.NE.1) WRITE (6,*)'To or From  F-K space (T/F) :'
       READ (5,490) ANS
       NCEN = NSAMR / 2 + 1
       KSCEN = NSORS / 2 + 1
       KRCEN = NRECS / 2 + 1
       IF (ANS .EQ. 'F' .OR. ANS .EQ. 'f') THEN
        SIGNA = +1.0
        SIGNB = +1.0
        SIGNC = +1.0
        DO 130 K = 1, NRECS
         DO 130 J = 1, NSORS
          CP(NCEN,J,K) = CONJG(CP(1,J,K))
          DO 130 I = 1, NSAMR / 2 - 1
           CP(NCEN + I,J,K) = CONJG(CP(NCEN - I,J,K))
  130   CONTINUE
C   SWAP AROUND TWO UPPER QUADRANTS
        DO 150 K = 1, NRECS / 2 - 1
         DO 150 J = 1, NSORS / 2 - 1
          DO 150 I = NCEN + 1, NSAMR
           CTEMP = CP(I,J + KSCEN,K + KRCEN)
           CP(I,KSCEN+J,KRCEN+K) = CP(I,KSCEN-J,KRCEN-K)
           CP(I,KSCEN-J,KRCEN-K) = CTEMP
  150   CONTINUE
       ELSE
        IF (IBAT.NE.1) WRITE (6,*)'FILTER RESPONSE WANTED ?'
        READ (5,490) ANSWER
        IF (ANSWER .EQ. 'Y' .OR. ANSWER .EQ. 'y') THEN
         NCP = NSAM * NREC * NSOR
         CALL ZERO(NCP, R4DAT(1,1,1))
         R4DAT(512,12,12) = 1.0
        END IF
        SIGNA = -1.0
        SIGNB = -1.0
        SIGNC = -1.0
        NZERO = NSORS * NRECS * NSAMR * 2
        CALL ZERO(NZERO, CP(1,1,1))
        DO 180 K = 1, NREC
         DO 180 J = 1, NSOR
          DO 180 I = 1, NSAM
           CP(I,J,K) = CMPLX(R4DAT(I,J,K),0.0)
  180   CONTINUE
       END IF
       CALL FFT3D(NSAMR, NSORS, NRECS, CP, SIGNA, SIGNB,
     1 SIGNC,CW)
       IF (ANS .EQ. 'F' .OR. ANS .EQ. 'f') THEN
        IF (IBAT.NE.1) PRINT *, 'WRITING CP TO R4DAT'
        DO 230 K = 1, NREC
         DO 230 J = 1, NSOR
          DO 230 I = 1, NSAM
           R4DAT(I,J,K) = REAL(CP(I,J,K))
  230   CONTINUE
       END IF
      ELSE IF (IANSWR .EQ. 4) THEN
C  PUT filter response into array FILSPC
       IF (IBAT.NE.1) THEN
        PRINT *,'NF,NSK,NRK,DF,DSK,DRK',NF,NSK,NRK,DF
        WRITE (6,*)'PASS or REJECT filter (P/R)?'
       ENDIF
       READ (5,490) ANS
       WRITE (6,*)'Peters or Neils filter (P/N)?'
```

```
      READ (5,490) ANP
      IF (ANP.EQ.'N'.OR.ANP.EQ.'n') THEN
       CALL PIE3B(NSK, NRK, NF, FILSPC, DSK, DRK, DF, IBAT )
      ELSE
       CALL PIE3(NSK, NRK, NF, FILSPC, DSK, DRK, DF, IBAT )
      ENDIF
      PRINT*,'PIE OK'
      IF (ANS .EQ. 'R') THEN
       DO 250 K = 1, NRECS
        DO 250 J = 1, NSORS
         DO 250 I = 1, NF
          FILSPC(I,J,K) = 1.0 - FILSPC(I,J,K)
 250  CONTINUE
      END IF
C APPLY FILTER FILSPC
       DO 270 K = 1, NRECS / 2
c -/- kr/ks quadrant
        DO 271 J = 1, NSORS / 2
         DO 271 I = 1, NF
          CP(I,J,K) = CP(I,J,K) * FILSPC(I,J + NSORS/2,K + NRECS/2)
 271    CONTINUE
c -/+ kr/ks quadrant
        DO 272 J = NSORS / 2 + 1, NSORS
         DO 272 I = 1, NF
          CP(I,J,K) = CP(I,J,K) * FILSPC(I,J - NSORS/2,K + NRECS/2)
 272    CONTINUE
 270  CONTINUE
       DO 290 K = NRECS / 2 + 1, NRECS
c +/- kr/ks quadrant
        DO 291 J = 1, NSORS / 2
         DO 291 I = 1, NF
          CP(I,J,K) = CP(I,J,K) * FILSPC(I,J + NSORS/2,K - NRECS/2)
 291    CONTINUE
c +/+ kr/ks quadrant
        DO 292 J = NSORS / 2 + 1, NSORS
         DO 292 I = 1, NF
          CP(I,J,K) = CP(I,J,K) * FILSPC(I,J - NSORS/2,K - NRECS/2)
 292    CONTINUE
 290  CONTINUE

      ELSE IF (IANSWR .EQ. 5 ) THEN
       IF (IBAT.NE.1) PRINT *, 'Constant KS,KR,diag k,freq slice'//
     1  ' or 3d (s/r/k/f/d)?'
       READ (5,*) KSORR
       IF (IBAT.NE.1) PRINT *, 'SELECT MAX FREQY FOR PLOT '
       READ (5,*) FMAX
       IF (KSORR.EQ.'D'.OR.KSORR.EQ.'d') THEN
        IF (IBAT.NE.1) PRINT *, 'KS/KR/Freq data/Freq filt'//
     1  'filter slices (s/r/q/p/f)?'
        READ (5,*) KORSSL
        IF (KORSSL.EQ.'f'.OR.KORSSL.EQ.'p') THEN
         DO 770 K = 1, NRECS / 2
c -/- kr/ks quadrant
          DO 771 J = 1, NSORS / 2
           DO 771 I = 1, NF
           RTEMP(K,J,I) = FILSPC(I,J + NSORS/2,K + NRECS/2)
 771       CONTINUE
c -/+ kr/ks quadrant
          DO 772 J = NSORS / 2 + 1, NSORS
           DO 772 I = 1, NF
           RTEMP(K,J,I) = FILSPC(I,J - NSORS/2,K + NRECS/2)
 772       CONTINUE
 770      CONTINUE
         DO 790 K = NRECS / 2 + 1, NRECS
c +/- kr/ks quadrant
          DO 791 J = 1, NSORS / 2
           DO 791 I = 1, NF
           RTEMP(K,J,I) = FILSPC(I,J + NSORS/2,K - NRECS/2)
 791       CONTINUE
c +/+ kr/ks quadrant
          DO 792 J = NSORS / 2 + 1, NSORS
           DO 792 I = 1, NF
            RTEMP(K,J,I) = FILSPC(I,J - NSORS/2,K - NRECS/2)
 792       CONTINUE
 790      CONTINUE
        ELSE
         DO 321 K = 1, NRECS
          DO 321 J = 1, NSORS
           DO 321 I = 1, NF
            IF (SQRDAT .EQ. 'Y') THEN
             RTEMP(K,J,I) = SQRT(CABS(CP(I,J,K)))
            ELSE
             RTEMP(K,J,I) = CABS(CP(I,J,K))
            ENDIF
 321      CONTINUE
        ENDIF
        IF (KORSSL.EQ.'f') KORSSL = 's'
        IF (KORSSL.EQ.'p') KORSSL = 'q'

        DO 322 I = 1, NF
         DO 323 J = 1, NSORS
          DO 324 K = 1, NRECS / 2
           TEMP = RTEMP(K,J,I)
           RTEMP(K,J,I) = RTEMP(K + NRECS/2,J,I)
           RTEMP(K + NRECS/2,J,I) = TEMP
 324       CONTINUE
          RTEMP(NRK,J,I) = RTEMP(1,J,I)
 323      CONTINUE
         DO 325 K = 1, NRECS
          DO 326 J = 1, NSORS / 2
           TEMP = RTEMP(K,J,I)
           RTEMP(K,J,I) = RTEMP(K,J + NSORS/2,I)
           RTEMP(K,J + NSORS/2,I) = TEMP
 326       CONTINUE
          RTEMP(K,NSK,I) = RTEMP(K,1,I)
 325      CONTINUE
 322     CONTINUE

        IF (IBAT.NE.1) PRINT *, 'How many slices (1,3,5,9,17)?'
        READ (5,*) NSLICE
        CALL FK3DPL(NRK, NSK, NF, RTEMP, DT, DSZ, DRZ,
     1  FMAX, NCHTS, CHTS, SQRDAT, KORSSL, NSLICE,
     2  TEMPLO, KKDUM, NDUM, ILEG)
       ELSEIF (KSORR.EQ.'F'.OR.KSORR.EQ.'f') THEN
        IF(IBAT.NE.1)PRINT*,'filter (1) or data (0)'
        READ (5,*) IFIDA
        IF(IBAT.NE.1)PRINT*,'What frequency?'
        READ (5,*) DFREQ
        NFI = INT(DFREQ/DF) +1
        PRINT*,'NFI=',NFI,DF,DFREQ
        DZ1 = DSZ
        DZ2 = DRZ
        NK2 = NRK
        NK1 = NSK
        DO 343 J = 1, NSORS
         DO 333 I = 1, NRECS
          IF (IFIDA .EQ. 1) THEN
           KKDUM(J,I) = FILSPC(NFI,J,I)
          ELSE
           IF (SQRDAT .EQ. 'Y') THEN
            KKDUM(J,I) = SQRT(CABS(CP(NFI,J,I)))
           ELSE
            KKDUM(J,I) = CABS(CP(NFI,J,I))
           ENDIF
          ENDIF
 333      CONTINUE
 343     CONTINUE
C  Now rearrange K space so that 0,0,0,0 is in centre of K axis
        DO 401 I = 1, NSORS / 2
         DO 391 J = 1, NRECS
          TEMP = KKDUM(I,J)
          KKDUM(I,J) = KKDUM(I + NRESO/2,J)
          KKDUM(I + NRESO/2,J) = TEMP
 391      CONTINUE
```

```
401    CONTINUE
       DO 411 J = 1, NK2
        KKDUM(NK1,J) = KKDUM(1,J)
411    CONTINUE
       CALL KKPLOT(NK1,NK2,KKDUM,DZ1,DZ2,
     1 NCHTS,CHTS,SQRDAT)
      ELSE
       LOGYN = 'N'
       YN3D = 'N'
C  write amplitude spectrum to correctly orientated array AMPSPC
       IF (KSORR.EQ.'S'.OR.KSORR.EQ.'s') THEN
        IF(IBAT.NE.1)PRINT*,'Which slice (1 -',NSK,')?'
        READ (5,*) KPL
        DZ = DRZ
        NK = NRK
        NRESO = NRECS
        DO 340 J = 1, NRECS
         DO 330 I = 1, NF
          IF (SQRDAT .EQ. 'Y') THEN
           AMPSPC(J,I) = SQRT(CABS(CP(I,KPL,J)))
          ELSE
           AMPSPC(J,I) = CABS(CP(I,KPL,J))
          ENDIF
330      CONTINUE
340     CONTINUE
       ELSEIF (KSORR.EQ.'R'.OR.KSORR.EQ.'d') THEN
        IF(IBAT.NE.1)PRINT*,'Which slice (1 -',NRK,')?'
        READ (5,*) KPL
        DZ = DSZ
        NK = NSK
        NRESO = NSORS
        DO 341 J = 1, NSORS
         DO 331 I = 1, NF
          IF (SQRDAT .EQ. 'Y') THEN
           AMPSPC(J,I) = SQRT(CABS(CP(I,J,KPL)))
          ELSE
           AMPSPC(J,I) = CABS(CP(I,J,KPL))
          ENDIF
331      CONTINUE
341     CONTINUE
       ELSE
        IF(IBAT.NE.1)PRINT*,'filter (1) or data (0)'
        READ (5,*) IFIDA
        IF(IBAT.NE.1)PRINT*,'++ diag (1=Y)?'
        READ (5,*) IPONE
        IF (IPONE.EQ.1) THEN
         APONE = 'Y'
        ELSE
         APONE = 'N'
        ENDIF
c this assumes DSZ = DRZ
        DZ = DSZ/SQRT(2.)
        NK = NSK
        NRESO = NSORS

        DO 342 J = 1, NSORS
         DO 332 I = 1, NF
          IF (IFIDA .EQ. 1) THEN
           IF (APONE .EQ. 'Y') THEN
            AMPSPC(J,I) = FILSPC(I,J,J)
           ELSE
            AMPSPC(J,I) = FILSPC(I,J,NRECS-J)
           ENDIF
          ELSE
           IF (APONE .EQ. 'Y') THEN
            IF (SQRDAT .EQ. 'Y') THEN
             AMPSPC(J,I) = SQRT(CABS(CP(I,J,J)))
            ELSE
             AMPSPC(J,I) = CABS(CP(I,J,J))
            ENDIF
           ELSE
            IF (SQRDAT .EQ. 'Y') THEN
             AMPSPC(J,I) = SQRT(CABS(CP(I,J,NRECS-J+1)))
            ELSE
             AMPSPC(J,I) = CABS(CP(I,J,NRECS-J+1))
            ENDIF
           ENDIF
          ENDIF
332      CONTINUE
342     CONTINUE
       ENDIF
C
C Now rearrange K space so that 0.0,0.0 is in centre of K axis
C
       DO 400 I = 1, NRESO / 2
        DO 390 J = 1, NF
         TEMP = AMPSPC(I,J)
         AMPSPC(I,J) = AMPSPC(I + NRESO/2,J)
         AMPSPC(I + NRESO/2,J) = TEMP
390     CONTINUE
400    CONTINUE
       DO 410 J = 1, NF
        AMPSPC(NK,J) = AMPSPC(1,J)
410    CONTINUE
       CALL FKPLOT(NK, NF, AMPSPC, DT, DZ, FMAX, NCHTS,
     1CHTS, LOGYN, YN3D, SQRDAT)
      ENDIF
      ELSE IF (IANSWR .EQ. 6) THEN
       IF (IBAT.NE.1) PRINT *, ' No. of contours required (<=8) :'
       READ (5,*) NCHTS
       DO 420 J = 1, NCHTS
        IF (IBAT.NE.1) WRITE (6,*)' Contour level ', J, ' : '
        READ (5,*) CHTS(J)
420   CONTINUE
      ELSE IF (IANSWR .EQ. 7) THEN
       IF (SQRDAT .EQ. 'N') THEN
        SQRDAT = 'Y'
       ELSE IF (SQRDAT .EQ. 'Y') THEN
        SQRDAT = 'N'
       END IF
      ELSE IF (IANSWR .EQ. 8) THEN
       IF (IBAT.NE.1) WRITE (6,*)'No of traces to apply taper over '//
     1  '(affects L-1 traces):'
       READ (5,*) LTAPER
       IF (IBAT.NE.1) WRITE (6,*)'Enter first and last traces with '//
     1  'significant data:'
       READ (5,*) NTRAC0, NTRAC1
       PI = 3.1415926535
       IF (IBAT.NE.1) WRITE (6,*)'Apply / Remove Taper (A/R):'
       READ (5,490) ANS
       DO 430 J = 1, LTAPER
        THETA = PI * REAL(J) / REAL(LTAPER)
        FAC = 0.5 * (1.0 - COS(THETA))
        IF (ANS .EQ. 'R' .OR. ANS .EQ. 'r') FAC = 1.0 / FAC
        NTR0 = NTRAC0 - 1 + J
        NTR1 = NTRAC1 + 1 - J
        IF (NTR0 .LE. 0 .OR. NTR1 .GT. NCHR) GO TO 430
        DO 440 K = 1, NSOR
         DO 440 I = 1, NSAM
          R4DAT(I,K,NTR0) = R4DAT(I,K,NTR0) * FAC
          R4DAT(I,K,NTR1) = R4DAT(I,K,NTR1) * FAC
440    CONTINUE
        DO 441 K = 1, NREC
         DO 441 I = 1, NSAM
          R4DAT(I,NTR0,K) = R4DAT(I,NTR0,K) * FAC
          R4DAT(I,NTR1,K) = R4DAT(I,NTR1,K) * FAC
441    CONTINUE
430   CONTINUE
      ELSE IF (IANSWR .EQ. 9) THEN
       DO 450 K = 1, NREC
        DO 450 J = 1, NSOR
         JK = (K-1)*NREC + J
         CALL RMSERR(NSAM, R4DAT(1,J,K), TEMP1(JK))
450   CONTINUE
```

```
      CALL MAXSN(NSNR, TEMP1, RMSMAX, MAXTRC)
      DO 460 K = 1, NREC
       DO 461 J = 1, NSOR
        JK = (K-1)*NREC + J
        IF (TEMP1(JK) .EQ. 0.0) GO TO 461
        SCALE = RMSMAX / TEMP1(JK)
        IF (IBAT.NE.1) PRINT *, 'SCALE #', JK, ' = ', SCALE
        DO 470 I = 1, NSAM
         R4DAT(I,J,K) = R4DAT(I,J,K) * SCALE
470     CONTINUE
461    CONTINUE
460   CONTINUE
     ELSE IF (IANSWR .EQ. 10.OR.IANSWR.EQ.12) THEN
      IF (IBAT.NE.1) PRINT*,'How many traces (in 1d) affected?'
      READ(5,*) NTRGA
      IF (IBAT.NE.1) PRINT*,'Enter trace/id no, gain :'
      DO 480 I = 1,NTRGA
       READ(5,*) IDTRGA(I),GAINAP(IDTRGA(I))
480   CONTINUE
      IF(IANSWR.EQ.10) THEN
       DO 481 K = IDTRGA(1),IDTRGA(NTRGA)
        DO 481 J = IDTRGA(1),IDTRGA(NTRGA)
         DO 481 I = 1,NSAM
          IF(GAINAP(K).GT.GAINAP(J)) THEN
           R4DAT(I,J,K) = R4DAT(I,J,K)*GAINAP(K)
          ELSE
           R4DAT(I,J,K) = R4DAT(I,J,K)*GAINAP(J)
          ENDIF
481    CONTINUE
      ELSE
       IF (IBAT.NE.1) PRINT*,'Shot (1) or Receiver(0) gathers :'
       READ(5,*) IRORS
       DO 482 K = 1,NREC
        DO 482 J = IDTRGA(1),IDTRGA(NTRGA)
         DO 482 I = 1,NSAM
          IF (IRORS.EQ.1) THEN
           R4DAT(I,J,K) = R4DAT(I,J,K)*GAINAP(J)
          ELSE
           R4DAT(I,K,J) = R4DAT(I,K,J)*GAINAP(J)
          ENDIF
482    CONTINUE
      ENDIF
     ELSE IF (IANSWR .EQ. 11) THEN
      IF (IBAT.NE.1) THEN
       WRITE (6,*)'APPLY SAME MUTE TO ALL gathers (1=Y;0=N) :'
      ENDIF
      READ (5,*) I2ANS
700   IF (IBAT.NE.1) THEN
        WRITE (6,*)'APPLY SAME MUTE TO ALL TRACES (1=Y) :'
      ENDIF
      READ (5,*) IOANS
      IF (IOANS .EQ. 1) THEN
       IF (IBAT.NE.1) WRITE (6,*)'Enter length of cos tap in sams :'
       READ (5,*) NTAP10
       IF (NTAP10 .EQ. 0) GO TO 730
       IF (IBAT.NE.1) WRITE (6,*)'Enter start of cos tap (0=FB):'
       READ (5,*) NTAP00
       DO 731 JJ=1,NSORS
        DO 730 J = 1, NCHR
         NTAP0 = NFIRST(JJ,J) + NTAP00
         NTAP1 = NTAP10 + NTAP0
         IF (NTAP1 - 1 .GT. NSAM .OR. NTAP0 .LT. 1) THEN
          IF (IBAT.NE.1) THEN
           WRITE (6,*)'CHANNEL ', J
           WRITE (6,*)'ERROR NTAP0,NTAP1 = ',NTAP0,NTAP1
          ENDIF
          GO TO 700
         END IF
         DO 710 I = NTAP0 + 1, NTAP1 - 1
          FAC = REAL(I - NTAP0) / REAL(NTAP1 - NTAP0)
          FAC = FAC * 3.1415926535
          FAC = 0.5 * (1.0 - COS(FAC))
          R4DAT(I,JJ,J) = R4DAT(I,JJ,J) * FAC
710      CONTINUE
         DO 720 I = 1, NTAP0
          R4DAT(I,JJ,J) = 0.0
720      CONTINUE
730     CONTINUE
        IF (I2ANS.EQ.0) THEN
         WRITE (6,*) 'Gather',JJ+1
         IF (IBAT.NE.1) WRITE (6,*)'Length of cos tap in sams:'
         READ (5,*) NTAP10
         IF (IBAT.NE.1) WRITE (6,*)'Start of cos tap (0=FB):'
         READ (5,*) NTAP00
        ENDIF
731    CONTINUE
      ELSE
740    IF (IBAT.NE.1)
     1  WRITE (6,*)'Length of cosine taper (0=NO mute):'
        READ (5,*) NTAP10
        IF (NTAP10 .EQ. 0) GO TO 781
        IF (IBAT.NE.1)
     1  WRITE (6,*)'Start of cosine taper (0=1st break):'
        READ (5,*) NTAP00
        IF (IBAT.NE.1)
     1  WRITE (6,*)'First and last traces to apply mute:'
        READ (5,*) NGO0, NGO1
        DO 701 JJ=1,NSORS
         DO 780 J = NGO0, NGO1
          NTAP0 = NFIRST(JJ,J) + NTAP00
          NTAP1 = NTAP10 + NTAP0
          IF (NTAP1 - 1 .GT. NSAM .OR. NTAP0 .LT. 1) THEN
           IF (IBAT.NE.1)
     1      WRITE (6,*)'ERROR NTAP0,NTAP1=',NTAP0,NTAP1
           GO TO 780
750        END IF
          DO 760 I = NTAP0 + 1, NTAP1 - 1
           FAC = REAL(I - NTAP0) / REAL(NTAP1 - NTAP0)
           FAC = FAC * 3.14159265
           FAC = 0.5 * (1.0 - COS(FAC))
           R4DAT(I,JJ,J) = R4DAT(I,JJ,J) * FAC
760        CONTINUE
          DO 871 I = 1, NTAP0
           R4DAT(I,JJ,J) = 0.0
871        CONTINUE
780       CONTINUE
         IF (I2ANS.EQ.0) THEN
          WRITE (6,*) 'Gather',JJ+1
          WRITE (6,*)'Length of cosine taper (0=NO mute):'
          READ (5,*) NTAP10
          WRITE (6,*)'Start of cosine taper (0=1st break):'
          READ (5,*) NTAP00
          WRITE (6,*)'First and last traces to apply mute:'
          READ (5,*) NGO0, NGO1
         ENDIF
701     CONTINUE
781    END IF
     ELSE IF (IANSWR .EQ. 13) THEN
      IF (ILEG.EQ.1) THEN
       ILEG = 0
      ELSE
       ILEG = 1
      ENDIF
     ENDIF
     IF (IANSWR .NE. 0) GO TO 10
     RETURN
     END

C-------------------------------------------------------------------------
      SUBROUTINE MENUIO3(NSAM, NSOR, NREC, IPDISC,
     1 IHFORM, OPDISC, IHEAD, IFNOHD, IBAT)
C Menu for adjustment of I/O parameters
      CHARACTER*25 IPDISC, OPDISC
      CHARACTER IHFORM*4
```

```
10 IF (IBAT.NE.1) THEN
  WRITE (6,*)
  WRITE (6,*)'          I/O Setup Menu          '
  WRITE (6,*)'          ~~~ ~~~~~~ ~~~~          '
  WRITE (6,20) IHFORM
20 FORMAT (/'   1  File format (heads in/out) :', A4)
  WRITE (6,30) IPDISC
30 FORMAT ('   2  Input File Name          :', A25)
  WRITE (6,40) OPDISC
40 FORMAT ('   3  Output File Name         :', A25)
  WRITE (6,50) NSAM
50 FORMAT (/'   4  Record Length (samples)   :', I4)
  WRITE (6,60) NSOR
60 FORMAT ('   5  Number of shots           :', I3)
  WRITE (6,70) NREC
70 FORMAT ('   6  Number of receivers       :', I3)
  WRITE (6,80) IHEAD
80 FORMAT (/'   7  Write out header? (1=no)   :', I2)
  WRITE (6,90) IFNOHD
90 FORMAT ('   8  Header in file?   (1=no)   :', I2)
  WRITE (6,100)
100 FORMAT (/'   0  Return to MAIN MENU        ')
  ENDIF
  READ (5,*) IANSWR
110 FORMAT (A25)
  IF (IANSWR .EQ. 1) THEN
    IF (IHFORM .EQ. 'IN') THEN
      IHFORM = 'OUT'
    ELSE
      IHFORM = 'IN'
    END IF
  END IF
  IF (IANSWR .EQ. 2) THEN
    IF (IBAT.NE.1) WRITE (6,*)
    IF (IBAT.NE.1) WRITE (6,*)'      Input Disc name   :'
    READ (5,110) IPDISC
  END IF
  IF (IANSWR .EQ. 3) THEN
    IF (IBAT.NE.1) WRITE (6,*)
    IF (IBAT.NE.1) WRITE (6,*)'      Output Disc name   :'
    READ (5,110) OPDISC
  END IF
  IF (IANSWR .EQ. 4) THEN
    IF (IBAT.NE.1) WRITE (6,*)
    IF (IBAT.NE.1) WRITE (6,*)'      Number of samples  :'
    READ (5,*) NSAM
  END IF
  IF (IANSWR .EQ. 5) THEN
    IF (IBAT.NE.1) WRITE (6,*)
    IF (IBAT.NE.1) WRITE (6,*)'      Number of shots :'
    READ (5,*) NSOR
  END IF
  IF (IANSWR .EQ. 6) THEN
    IF (IBAT.NE.1) WRITE (6,*)
    IF (IBAT.NE.1) WRITE (6,*)'      Number of receivers :'
    READ (5,*) NREC
  END IF
  IF (IANSWR .EQ. 7) THEN
    IF (IHEAD .EQ. 0) THEN
      IHEAD = 1
    ELSE
      IHEAD = 0
    END IF
  END IF
  IF (IANSWR .EQ. 8) THEN
    IF (IFNOHD .EQ. 0) THEN
      IFNOHD = 1
    ELSE
      IFNOHD = 0
    END IF
  END IF
  IF (IANSWR .NE. 0) GO TO 10
  RETURN
  END

C-----------------------------------------------------------------------
C . subroutine to calculate pie slice filter for application to FKK
C spectrum PASS IN KS - PASS IN KR
C
C  REMEMBER TO TRANSFORM TO CORRECT QUADS
c  (this advice seems to be for when you leave the routine)
c   (ie. rearrange FILSPC as you apply it to CP  PSR 92)
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
C  calling routine   must also have data arranged in appropriate
C sequence require an odd no. of samples e.g 65,129,257
    SUBROUTINE PIE3(NSK, NRK, NF, PIEFLT, DSK, DRK,
    1 DF, IBAT)
    REAL PIEFLT(NF,NSK,NRK), DIST, RHISLO, RHICTS,
    REAL RLOSLO, RLOCTS
    REAL SHISLO, SHICTS, SLOSLO, SLOCTS
    DOUBLE PRECISION HISLOS, HICTSS, LOSLOS, LOCTSS
    DOUBLE PRECISION MS3, MS6,MR3, MR6
    DOUBLE PRECISION HISLOR, HICTSR, LOSLOR, LOCTSR
    INTEGER KSNYQ, KRNYQ
    CHARACTER*1 ANS
    PI = 3.1415926535
    KSNYQ = NSK / 2 + 1
    KRNYQ = NRK / 2 + 1
10 FORMAT (A1)
    DO 20 K = 1, NRK
     DO 20 J = 1, NSK
      CALL ZERO(NF, PIEFLT(1,J,K))
20 CONTINUE
    IF (IBAT.NE.1)
    1  WRITE (6,*)'TAPER IN OR OUTWARDS FROM SLOPE (I/O) ?'
    READ (5,10) ANS

    IF (IBAT.NE.1)
    1  WRITE (6,*)'I/P HIGH CUTOFF SLOPE (m/s) (-VE for L quad):'
    READ (5,*) HISLOS
    IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
     IF (IBAT.NE.1)
    1    WRITE (6,*)'INPUT HI CUTOFF TAPER SLOPE(m/sec) :'
     READ (5,*) HICTSS
     IF (HISLOS + HICTSS .EQ. 0.0) THEN
      MS3 = 0.0
     ELSE
      MS3 = ((HISLOS*HICTSS-1)-SQRT(HISLOS**2+ HICTSS**2
    1  + (HISLOS*HICTSS)**2 + 1)) / (HISLOS + HICTSS)
     END IF
     IF (IBAT.NE.1) PRINT*,'HISLOS,HICTSS,MS3',
    1  HISLOS,HICTSS,MS3
     MULTS1 = 1
     MULTS2 = 1
     IF (HICTSS .LT. 0.0) MULTS1 = -1
     IF (HISLOS .LT. 0.0 .AND. HICTSS .GE. HISLOS) MULTS2 = -1
    END IF
    IF (IBAT.NE.1)
    1  WRITE (6,*)'I/P LOW CUTOFF SLOPE (m/s) (-VE for L quad) :'
    READ (5,*) LOSLOS
    IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
     IF (IBAT.NE.1)
    1    WRITE (6,*)'INPUT LOW CUTOFF TAPER SLOPE (m/sec) :'
     READ (5,*) LOCTSS
     IF (LOSLOS + LOCTSS .EQ. 0.0) THEN
      MS6 = 0.0
     ELSE
      MS6=((LOSLOS*LOCTSS-1)-SQRT(LOSLOS**2+ LOCTSS**2
    1  + (LOSLOS*LOCTSS)**2 + 1)) / (LOSLOS + LOCTSS)
     END IF
     MULTS3 = 1
```

```
      MULTS4 = 1
      IF (LOCTSS .LT. 0.0) MULTS3 = -1
      IF (LOSLOS .LT. 0.0.AND.LOCTSS .LE. LOSLOS) MULTS4 = -1
      IF (IBAT.NE.1) PRINT*,'LOSLOS,LOCTSS,MS6',
     1  LOSLOS,LOCTSS,MS6
      END IF
      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/P HIGH CUTOFF SLOPE (m/s) (-VE for L quad):'
      READ (5,*) HISLOR
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1)
     1   WRITE (6,*)'INPUT HI CUTOFF TAPER SLOPE(m/sec) :'
       READ (5,*) HICTSR
       IF (HISLOR + HICTSR .EQ. 0.0) THEN
        MR3 = 0.0
       ELSE
        MR3 = ((HISLOR*HICTSR-1)-SQRT(HISLOR**2 +HICTSR**2
     1  + (HISLOR*HICTSR)**2 + 1)) / (HISLOR + HICTSR)
       END IF
       IF (IBAT.NE.1) PRINT*,'HISLOR,HICTSR,MR3',
     1  HISLOR,HICTSR,MR3
       MULTR1 = 1
       MULTR2 = 1
       IF (HICTSR .LT. 0.0) MULTR1 = -1
       IF (HISLOR .LT. 0.0 .AND. HICTSR.GE.HISLOR) MULTR2 = -1
      END IF
      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/P LOW CUTOFF SLOPE (m/s) (-VE for L quad) :'
      READ (5,*) LOSLOR
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1)
     1   WRITE (6,*)'INPUT LOW CUTOFF TAPER SLOPE (m/sec) :'
       READ (5,*) LOCTSR
       IF (LOSLOR + LOCTSR .EQ. 0.0) THEN
        MR6 = 0.0
       ELSE
        MR6=((LOSLOR*LOCTSR-1)-SQRT(LOSLOR**2+LOCTSR**2
     1  + ( LOSLOR*LOCTSR)**2 + 1)) / (LOSLOR + LOCTSR)
       END IF
       MULTR3 = 1
       MULTR4 = 1
       IF (LOCTSR .LT. 0.0) MULTR3 = -1
       IF (LOSLOR.LT.0.0.AND.LOCTSR .LE. LOSLOR) MULTR4 = -1
       IF (IBAT.NE.1) PRINT*,'LOSLOR,LOCTSR,MR6',
     1  LOSLOR,LOCTSR,MR6
       END IF
       IF (IBAT.NE.1) WRITE (6,*)'Enter no. of wraparounds required '
       READ (5,*) NWRAPS
      IF (ANS .EQ. 'I' .OR. ANS .EQ. 'i') THEN
       IF (IBAT.NE.1)
     1  WRITE (6,*)'INPUT DISTANCE IN SAMPLES FOR TAPER :'
       READ (5,*) NSAMTP
      END IF

      SHISLO = REAL(HISLOS)
      SHICTS = REAL(HICTSS)
      SLOSLO = REAL(LOSLOS)
      SLOCTS = REAL(LOCTSS)
      RHISLO = REAL(HISLOR)
      RHICTS = REAL(HICTSR)
      RLOSLO = REAL(LOSLOR)
      RLOCTS = REAL(LOCTSR)
      IF (ANS .EQ. 'O' .OR. ANS .EQ. 'o') THEN
       IF (IBAT.NE.1) THEN
        WRITE (6,*)'Enter ks/frequency axis intercepts Hicut,' //
     1     'Hizero, Locut, Lozero :'
        READ (5,*) SHSLIN, SHCTIN, SLSLIN, SLCTIN
        WRITE (6,*)'SHICTS,SHCTIN,MULTS1 ',
     1  SHICTS, SHCTIN, MULTS1
        WRITE (6,*)'SHISLO,SHSLIN,MULTS2 ',
     1  SHISLO, SHSLIN, MULTS2
        WRITE (6,*)'Enter kr/frequency axis intercepts Hicut,' //
     1     'Hizero, Locut, Lozero :'
        READ (5,*) RHSLIN, RHCTIN, RLSLIN, RLCTIN
        WRITE (6,*)'RHICTS,RHCTIN,MULTS1 ',
     1  RHICTS, RHCTIN, MULTR1
        WRITE (6,*)'RHISLO,RHSLIN,MULTS2 ',
     1  RHISLO, RHSLIN, MULTR2
       ELSE
        READ (5,*) SHSLIN, SHCTIN, SLSLIN, SLCTIN
        READ (5,*) RHSLTH, RHCTTH, RLSLTH, RLCTTH
       ENDIF
50     DO 60 I = 1, NF
        Y = DF * (I - 1)
        DO 70 K = 1, NRK
         Z = DRK * (K - KRNYQ)
         IF (DIST(Z,Y,RHICTS,RHCTIN,MULTR1) .GT. 0.0) THEN
           RESP1 = 0.0
         ELSE IF (DIST(Z,Y,RHISLO,RHSLIN,MULTR2) .GT. 0.0)
     1      THEN
          D1 = (Y - HISLOR*Z - RHSLIN) / (HISLOR - MR3)
          D2 = (Y - HICTSR*Z - RHCTIN) / (HICTSR - MR3)
          D1 = ABS(D1)
          D2 = ABS(D2)
C NOTE : COSMS3 TERM CANCELS OUT
          THETA = PI * D2 / (D1 + D2)
          RESP1 = 0.5 * (1 - COS(THETA))
         ELSE IF (DIST(Z,Y,RLOSLO,RLSLIN,MULTR3) .GT. 0.0)
     1      THEN
          RESP1 = 1.0
         ELSE IF (DIST(Z,Y,RLOCTS,RLCTIN,MULTR4) .GT. 0.0)
     1      THEN
          D1 = (Y - LOSLOR*Z - RLSLIN) / (LOSLOR - MR6)
          D2 = (Y - LOCTSR*Z - RLCTIN) / (LOCTSR - MR6)
          D1 = ABS(D1)
          D2 = ABS(D2)
          THETA = PI * D2 / (D1 + D2)
          RESP1 = 0.5 * (1 - COS(THETA))
         ELSE
          RESP1 = 0.0
         END IF
         IF (RESP1 .EQ. 0.0) GO TO 70
         DO 71 J = 1, NSK
          X = DSK * (J - KSNYQ)
          IF (DIST(X,Y,SHICTS,SHCTIN,MULTS1) .GT. 0.0) THEN
           RESP = 0.0
          ELSE IF (DIST(X,Y,SHISLO,SHSLIN,MULTS2) .GT. 0.0)
     1       THEN
           D1 = (Y - HISLOS*X - SHSLIN) / (HISLOS - MS3)
           D2 = (Y - HICTSS*X - SHCTIN) / (HICTSS - MS3)
           D1 = ABS(D1)
           D2 = ABS(D2)
C NOTE : COSMS3 TERM CANCELS OUT
           THETA = PI * D2 / (D1 + D2)
           RESP = 0.5 * (1 - COS(THETA))
          ELSE IF (DIST(X,Y,SLOSLO,SLSLIN,MULTS3) .GT. 0.0)
     1       THEN
           RESP = 1.0
          ELSE IF (DIST(X,Y,SLOCTS,SLCTIN,MULTS4) .GT. 0.0)
     1       THEN
           D1 = (Y - LOSLOS*X - SLSLIN) / (LOSLOS - MS6)
           D2 = (Y - LOCTSS*X - SLCTIN) / (LOCTSS - MS6)
           D1 = ABS(D1)
           D2 = ABS(D2)
           THETA = PI * D2 / (D1 + D2)
           RESP = 0.5 * (1 - COS(THETA))
          ELSE
           RESP = 0.0
          END IF
c          IF (IBAT.NE.1)PRINT*,'RESP =',RESP
          IF (RESP .EQ. 0.0) GO TO 71
C RESP1 (pie taper over kr) only used when no pie taper over ks
c if line below is uncommented
c          IF (RESP .EQ. 1.0) RESP = RESP*RESP1
```

```
          RESP = RESP*RESP1
C ur, how can it be greater than 0, since it was zeroed at the top?
          IF (PIEFLT(I,J,K) .GT. 0.0) THEN
           PIEFLT(I,J,K) = PIEFLT(I,J,K) * RESP
          ELSE
           PIEFLT(I,J,K) = RESP
          END IF
 71       CONTINUE
 70      CONTINUE
 60     CONTINUE
       IF (NWRAPS .NE. 0) THEN
        NWRAPS = NWRAPS - 1
        SHCTIN = 2.0 * ABS(SHICTS) * KKNYQ + SHCTIN
        SHSLIN = 2.0 * ABS(SHISLO) * KKNYQ + SHSLIN
        SLSLIN = 2.0 * ABS(SLOSLO) * KKNYQ + SLSLIN
        SLCTIN = 2.0 * ABS(SLOCTS) * KKNYQ + SLCTIN
        RHCTIN = 2.0 * ABS(RHICTS) * KKNYQ + RHCTIN
        RHSLIN = 2.0 * ABS(RHISLO) * KKNYQ + RHSLIN
        RLSLIN = 2.0 * ABS(RLOSLO) * KKNYQ + RLSLIN
        RLCTIN = 2.0 * ABS(RLOCTS) * KKNYQ + RLCTIN
        IF (IBAT.NE.1)  PRINT *, 'INTERCEPTS :',
     1  RHCTIN, RHSLIN, RLSLIN, RLCTIN
        GO TO 50
       END IF
      ELSE
       HISLOS = HISLOS / (DF/DSK)
       LOSLOS = LOSLOS / (DF/DSK)
       HISLOR = HISLOR / (DF/DRK)
       LOSLOR = LOSLOR / (DF/DRK)
       MULTS = 1
       MULTR = 1
       IF (HISLOS .LT. 0.0 .AND. LOSLOS .GT. 0.0) MULTS = -1
       IF (HISLOR .LT. 0.0 .AND. LOSLOR .GT. 0.0) MULTR = -1
       D0 = -REAL(NSAMTP)
       D1 = -D0
       PRINT *, 'HISLOS,LOSLOS,D0,D1', HISLOS, LOSLOS, D0, D1
       PRINT *, 'HISLOR,LOSLOR,D0,D1', HISLOR, LOSLOR, D0, D1

       DO 80 I = 1, NF
        Y = REAL(I - 1)
        DO 90 K = 1, NRK
         Z = REAL(K - KRNYQ)
         IF (DIST(Z,Y,RHISLO,RHSLIN,MULTR) .GT. 0.0) THEN
          RESP1 = 0.0
         ELSE IF (DIST(Z,Y,RHISLO,RHSLIN,MULTR) .GT. D0 .AND.
     1      DIST(Z,Y,RLOSLO,RLSLIN,1) .GT. 0.0) THEN
          THETA = DIST(Z,Y,RHISLO,RHSLIN,MULTR) * PI / D0
          RESP1 = 0.5 * (1 - COS(THETA))
         IF (DIST(Z,Y,RLOSLO,RLSLIN,1) .LE. D1) THEN
          THETA = DIST(Z,Y,RLOSLO,RLSLIN,1) * PI / D1
          RESP1 = RESP1 * 0.5 * (1 - COS(THETA))
         END IF
        ELSE IF (DIST(Z,Y,RLOSLO,RLSLIN,1) .GT. D1) THEN
         RESP1 = 1.0
        ELSE IF (DIST(Z,Y,RLOSLO,RLSLIN,1) .GT. 0.0) THEN
         THETA = DIST(Z,Y,RLOSLO,RLSLIN,1) * PI / D1
         RESP1 = 0.5 * (1 - COS(THETA))
        ELSE
         RESP1 = 0.0
        END IF
        IF (RESP1 .EQ. 0.0) GOTO 90
        DO 91 J = 1, NSK
         X = REAL(J - KSNYQ)
         IF (DIST(X,Y,SHISLO,SHSLIN,MULTS) .GT. 0.0) THEN
          RESP = 0.0
         ELSE IF (DIST(X,Y,SHISLO,SHSLIN,MULTS) .GT. D0
     1 .AND. DIST(X,Y,SLOSLO,SLSLIN,1) .GT. 0.0) THEN
          THETA = DIST(X,Y,SHISLO,SHSLIN,MULTS) * PI / D0
          RESP = 0.5 * (1 - COS(THETA))
          IF (DIST(X,Y,SLOSLO,SLSLIN,1) .LE. D1) THEN
           THETA = DIST(X,Y,SLOSLO,SLSLIN,1) * PI / D1
           RESP = RESP * 0.5 * (1 - COS(THETA))
          END IF
         ELSE IF (DIST(X,Y,SLOSLO,SLSLIN,1) .GT. D1) THEN
          RESP = 1.0
         ELSE IF (DIST(X,Y,SLOSLO,SLSLIN,1) .GT. 0.0) THEN
          THETA = DIST(X,Y,SLOSLO,SLSLIN,1) * PI / D1
          RESP = 0.5 * (1 - COS(THETA))
         ELSE
          RESP = 0.0
         END IF
         IF (RESP .EQ. 0.0) GO TO 91
         IF (RESP .EQ. 1.0) RESP = RESP*RESP1
         IF (PIEFLT(I,J,K) .GT. 0.0) THEN
          PIEFLT(I,J,K) = PIEFLT(I,J,K) * RESP
         ELSE
          PIEFLT(I,J,K) = RESP
         END IF
 91      CONTINUE
 90     CONTINUE
 80    CONTINUE
      END IF


      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/P LENGTH OF TAPER FOR K-SPACE (sams) :'
      READ (5,*) LNTAPK
      IF (IBAT.NE.1)
     1 WRITE (6,*)'I/P LENGTH OF TAPER FOR F-SPACE (sams) :'
      READ (5,*) LNTAPF
      IF (LNTAPK .GT. 0) THEN
C DO FOR CSGS
       DO 101 J = NSK - LNTAPK, NSK
        THETA = PI*(1. - REAL(J - (NSK - LNTAPK)) / LNTAPK)
        LOWLT = NSK - J + 1
        FAC = 0.5 * (1.0 - COS(THETA))
        DO 100 K = 1,NRK
         DO 100 I = 1, NF
          PIEFLT(I,J,K) = PIEFLT(I,J,K) * FAC
          PIEFLT(I,LOWLT,K) = PIEFLT(I,LOWLT,K) * FAC
 100      CONTINUE
 101    CONTINUE
C DO FOR CRGS BUT NB THE LIMIT ON J
C ******
       DO 102 K = NRK - LNTAPK, NRK
        THETA = PI*(1. - REAL(K - (NRK - LNTAPK)) / LNTAPK)
        LOWLT = NRK - K + 1
        FAC = 0.5 * (1.0 - COS(THETA))
        DO 103 J = 1, NSK
         DO 103 I = 1, NF
          PIEFLT(I,J,K) = PIEFLT(I,J,K) * FAC
          PIEFLT(I,J,LOWLT) = PIEFLT(I,J,LOWLT) * FAC
 103      CONTINUE
 102    CONTINUE
       END IF
       IF (LNTAPF .GT. 0) THEN
        DO 130 K = 1, NRK
         DO 130 J = 1, NSK
          DO 130 I = 1, LNTAPF
           THETA = REAL(I - 1) / LNTAPF * PI
           FAC = 0.5 * (1.0 - COS(THETA))
           PIEFLT(I,J,K) = PIEFLT(I,J,K) * FAC
 130    CONTINUE
       END IF
C  NOW SET LOW NYQ (-KNYQ) TO HIGH NYQ (+KNYQ)
       DO 140 I = 1, NF
        DO 141 K = 1, NRK
         PIEFLT(I,1,K) = PIEFLT(I,NSK,K)
 141    CONTINUE
C lower limit set to 2 since 1 already done in do loop above
        DO 142 J = 2, NSK
         PIEFLT(I,J,1) = PIEFLT(I,J,NRK)
 142    CONTINUE
 140   CONTINUE
```

```
      RETURN
      END

C-----------------------------------------------------------------------
      SUBROUTINE RDFILE3(NSAM,NSOR,NREC,NSAMR, NSORS,
     1 NRECS,R4DAT,NFIRST,RECDEP,SORDEP, DBGAIN,
     2 GCMSCL,DT,LEN,NF, SORPOS,NPROCS,IDPROC, IPDISC,
     3 IHFORM,IFNOHD,A, TEMP,IBAT)
C   reads in arrays to R4DAT
      DIMENSION R4DAT(NSAMR,NSORS,NRECS), NF(NRECS)
      DIMENSION GCMSCL(NRECS) NFIRST(NSORS,NRECS)
      DIMENSION RECDEP(NRECS),DBGAIN(NRECS)
      DIMENSION TEMP(NSAMR)
      DIMENSION ,IDPROC(5,NRECS), SORDEP(NSORS)
      CHARACTER IHFORM*4, A*180, ANS*1
      CHARACTER*25 IPDISC, OPDISC, IP1
      IF (IBAT.NE.1) WRITE (6,*)'Stack or New (1 0 -1 == REV POL ) :'
      READ (5,*) ISTACK
      K24 = NSAMR * NRECS * NSORS
      IF (ISTACK .EQ. 0) CALL ZERO(K24, R4DAT(1,1,1))
      DO 111 KK=1,NSOR
        IDCODE = (KK - 1) * (NREC + 1) + 1
C READ IN HEADER PARAMETERS
        IF (IBAT.NE.1) PRINT *, 'READING HEADERS ..',KK
        IF (IHFORM .EQ. 'OUT') THEN
          IDCODE = KK
          IP1 = IPDISC(1:lnb(IPDISC))//'.h'
          LEN = (NSAM + 2) * 4
        ELSE
          IP1 = IPDISC
        ENDIF
        OPEN (10,FILE=IP1,STATUS='OLD',ACCESS='DIRECT',
     1  RECL=LEN, IOSTAT=K2,ERR=92)
   92   IF (K2.NE.0) THEN
          PRINT*,'FILE DOES NOT EXIST'
          PRINT *, 'Enter a character'
          READ (5,120) ANS
          RETURN
        ENDIF
         IF (IHFORM .EQ. 'OUT') THEN
          READ (10,REC=IDCODE,IOSTAT=K1) A, NSAMA, NRECA,
     1     DT, NF, SORPOS, RECDEP
          IF (IBAT.NE.1) THEN
          PRINT*,'======================================'
          PRINT*,'=   The header says there are ',NRECA,' traces ='
          PRINT*,'=   with ',NSAMA,' samples / trace          ='
          PRINT*,'======================================'
          ENDIF
          NREC = NRECA
          NSAM = NSAMA
         ELSE
          READ (10,REC=IDCODE,IOSTAT=K1) A, SORPOS, NRECA,
     1 RECDEP,DBGAIN, GCMSCL, NF,NCR, NPROCS, IDPROC, DT
         ENDIF
         DO 85 J=1,NRECS
          NFIRST(KK,J) = NF(J)
   85    CONTINUE
   91    IF (K1.NE.0.AND.IBAT.NE.1) PRINT*,'PROBLEM HEADER'
        IF (IHFORM .EQ. 'OUT') THEN
         CLOSE (10)
         IDCODE = (KK-1) * NREC
         IP1 = IPDISC(1:lnb(IPDISC))//'.d'
         OPEN (10,FILE=IP1,STATUS='OLD',ACCESS='DIRECT',
     1   RECL=LEN, IOSTAT=K2,ERR=92)
        ENDIF
C READ IN DATA
        DO 110 J = 1, NREC
         NLIN = IDCODE + J
         IF(ISTACK.EQ.0)THEN
          READ(10,REC=NLIN,IOSTAT=K4)
     1   (R4DAT(I,KK,J),I=1,NSAM)
   93     IF (K4.NE.0.AND.IBAT.NE.1) PRINT*,'PROBLEM RECORD',
     1       NLIN,IDCODE,K4
         ELSE
          READ(10,REC=NLIN,IOSTAT=K4)(TEMP(I),I=1,NSAM)
   95     IF (K4.NE.0.AND.IBAT.NE.1) PRINT*,'PROBLEM RECORD',
     1       NLIN,IDCODE,K4
          DO 100 I = 1, NSAM
           IF (ISTACK .GE. 0) THEN
            R4DAT(I,KK,J) = R4DAT(I,KK,J) + TEMP(I)
           ELSE
            R4DAT(I,KK,J) = R4DAT(I,KK,J) - TEMP(I)
           END IF
  100      CONTINUE
         ENDIF
  110   CONTINUE
        CLOSE(10)
        IF (NRECA.NE.NREC.AND.IBAT.NE.1) THEN
         PRINT*,'NREC and NRECA are not equal'
         PRINT*,'NREC, NRECA =',NREC,NRECA
        ENDIF
  111 CONTINUE
      IF (IBAT.NE.1) PRINT *, 'Reset parameters to defaults (Y/N)?'
      READ (5,120) ANS
  120 FORMAT (A1)
      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
        OPEN(UNIT=1,FILE='xhr3.dfolt')
        READ (1,130) IPDISC, OPDISC, IHFORM,NSAM,NSOR,NREC
  130   FORMAT (A25/A25/A4/I4/I3/I3)
        READ (1,140) TOPSOR, TOPREC, RECSEP, DT
  140   FORMAT (F6.1/F6.1/F6.1/F6.2/F6.2)
        CLOSE (1)
        RECDEP(1) = TOPREC
        SORDEP(1) = TOPSOR
        DO 150 I = 2, NREC
          RECDEP(I) = RECDEP(I - 1) + RECSEP
          SORDEP(I) = SORDEP(I - 1) + RECSEP
  150   CONTINUE
      END IF
  170 RETURN
      END

C-----------------------------------------------------------------------
C  Subroutine to write out data to disc file
c
      SUBROUTINE WRFIL3(NSAM, NCHR, NSAMR, NCHRR,
     1 R4DAT, NFIRST, NRECS, RECDEP, DBGAIN, GCMSCL,
     2 NSHOT, DT, LEN, SORPOS, NPROCS, IDPROC, OPDISC,
     3 IHFORM, A1,A2,A3,A4, A5, A6, A7, A8, A9,IHEAD,IBAT)
      DIMENSION R4DAT(NSAMR,NCHRR), NFIRST(NCHRR)
      DIMENSION RECDEP(NCHRR), A(9),IDPROC(5,NCHRR)
      DIMENSION DBGAIN(NCHRR), GCMSCL(NCHRR), NCR(2)
      CHARACTER IHFORM*4
      CHARACTER*20 A1, A2, A3, A4, A5, A6, A7, A8, A9, A
      CHARACTER*25 OPDISC, F1, F2
      IF (IBAT.NE.1) PRINT *, 'WRITING ......'
      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
      NCR(1) = 0
      NCR(2) = 0
      IDCODE = (NSHOT-1)*(NCHR+1)+1
       IF (IHFORM .EQ. 'IN') THEN
        IF (IBAT.NE.1) PRINT *, IDCODE, LEN
        OPEN (11,FILE=OPDISC,STATUS='UNKNOWN',
     1  FORM='UNFORMATTED',ACCESS='DIRECT',RECL=LEN)
C OUTPUT PROCESSING AND DISPLAY PARAMETERS
        IF (IHEAD.EQ.0) THEN
```

```
      WRITE (11,REC=IDCODE) A, SORPOS, NCHR, RECDEP,
     1 DBGAIN, GCMSCL, NFIRST, NCR, NPROCS, IDPROC, DT
     ELSE
C for use if header is too long
      WRITE (11,REC=IDCODE) A, SORPOS, NCHR
     ENDIF
C  WRITE SEISMOGRAM RECORDS
      DO 10 J = 1, NCHR
       NREC = IDCODE + J
       WRITE (11,REC=NREC) (R4DAT(I,J),I=1,NSAM)
 10    CONTINUE
     ELSE IF (IHFORM .EQ. 'OUT') THEN
      LEN1 = (NSAM + 2) * 4
      IDCODE = NSHOT
      IDCODE1 = (NSHOT-1)*NCHR
      IF (IBAT.NE.1) PRINT *, IDCODE, LEN1
      F1 = OPDISC(1:lnb(OPDISC))//'.d'
      F2 = OPDISC(1:lnb(OPDISC))//'.h'
      OPEN (11,FILE=F1,STATUS='UNKNOWN',
     1 FORM='UNFORMATTED',ACCESS='DIRECT',RECL=LEN1)
      OPEN (12,FILE=F2,STATUS='UNKNOWN',
     1 FORM='UNFORMATTED',ACCESS='DIRECT',RECL=LEN1)
C  OUTPUT PROCESSING AND DISPLAY PARAMETERS
     IF (IHEAD.EQ.0) THEN
      WRITE (12,REC=IDCODE) A,NSAM,NCHR,DT,NFIRST,
     1     SORPOS,RECDEP
     ELSE
C for use if header is too long
      WRITE (12,REC=IDCODE) A, SORPOS, NCHR
     ENDIF
C  WRITE SEISMOGRAM RECORDS
      DO 11 J = 1, NCHR
       NREC = IDCODE1 + J
       WRITE (11,REC=NREC) (R4DAT(I,J),I=1,NSAM)
 11    CONTINUE
      CLOSE (12)
     ENDIF
    CLOSE (11)
    RETURN
    END

c-------------------------------------------------------------
    SUBROUTINE FK3DPL(NRK, NSK, NF, FKDATA, DT, DSZ,
   1 DRZ, FMAX, NCHTS, CHTS, SQRDAT, KORSSL, NSLICE,
   2 TEMPLO, KKPLO, NDUM, ILEG)
C
C.   Routine to plot F-K-K amp. spectrum contained in the
C    array FKDATA(NK1,NK2,NF)
C    N.B. Sample order in K  --->  1        = -knyq
C          nk/2 + 1   = 0.0  nk       = +knyq
C    NF, NK  no. of frequencies ,  wavenumbers for plot
C    DT, DZ  temp and spat sampling intervals in microsecs and metres
C    NCHTS   no. of contour levels
C
     INTEGER NSK,NRK,NF,LENAR1(6),LENAR(4)
     INTEGER LENGTH(6),LENAR2(3),IPLANE(3)
     REAL DT, DRZ, DSZ, K1NYQ, K1MIN, K1MAX, FMIN, K2MAX
     REAL CHTS(NCHTS), FKDATA(NRK,NSK,NF), K2MIN, K2NYQ
     REAL TEMPLO(NRK,NDUM), KKPLO(NSK,NRK)
     CHARACTER TXTAR1(6)*10, TXTAR(4)*1,TXTAR2(3)*5
     CHARACTER*1 SQRDAT, KORSSL, TXTSTR(6)*10,DATE*24
C    Axis text strings and character string lengths
C    of axis texts.
     DATA TXTSTR/'KS','/m','KR','/m','Hz',
    .'Freq'/,LENGTH/6*-2/
     PRINT*,NRK, NSK, NF, DT, DSZ, DRZ, FMAX,
    1     SQRDAT, KORSSL, NSLICE, NDUM
C    Planes for axis annotation.
     DATA IPLANE/3,3,3/
c     DATA IPLANE /1,1,1/
c     DATA LENAR /4*0/
c     DATA TXTAR /4*''/
c     DATA LENAR1 /6*-1/
c     DATA TXTAR1 /'k (/m)','Wavenumber','f (Hz)','Frequency'
c    1     ,'units','Amplitude'/
     DATA LENAR2 /5,5,0/
     DATA TXTAR2 /'Below','Above',''/
     ICOUNT = 0
     RESERV = 30.0
     FNYQ = .5E6 / DT
     FMIN = 0.0
     AMIN = 0.0
     AMAX = 1.
     NLF = 1+NINT(FMAX*NF/FNYQ)
     IF (KORSSL.EQ.'Q'.OR.KORSSL.EQ.'q') THEN
      NKF = NLF
     ELSE
      NKF = NRK
     ENDIF
     IF (NSLICE.EQ.1) THEN
      ASPECT = 1.
      INTPL = (NKF-1)/2
     ELSEIF (NSLICE.EQ.3) THEN
      ASPECT = 1.5
      INTPL = (NKF-1)/2
     ELSEIF (NSLICE.EQ.5) THEN
      ASPECT = 3.
      INTPL = (NKF-1)/4
     ELSEIF (NSLICE.EQ.9) THEN
      ASPECT = 6.
      INTPL = (NKF-1)/8
     ELSE
      ASPECT = 12.
      INTPL = (NKF-1)/16
     ENDIF
     IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
      NK = NRK
      K1NYQ = .5 / DRZ
      K2NYQ = .5 / DSZ
     ELSE
      K1NYQ = .5 / DSZ
      K2NYQ = .5 / DRZ
      NK = NSK
     ENDIF
     K1MIN = -K1NYQ
     K1MAX = -K1MIN
     K2MIN = -K2NYQ
     K2MAX = -K2MIN

C  START PLOTTING ROUTINES
C  UNIRAS CALLS
     CALL GBEGIN('sel mx11;e','crap','fk.pic')
C  Plot size
     CALL GRPSIZ(XSIZE,ZSIZE)
     XOFF = RESERV
     ZOFF = RESERV
     CALL GLIMIT(K1MIN,K1MAX,K2MIN,K2MAX,FMIN,FMAX)
     CALL GVPORT(0.1*XSIZE,0.1*ZSIZE,0.7*XSIZE,0.8*ZSIZE)
C      CALL GVPORT(XOFF,ZOFF,XSIZE*.6,ZSIZE*.8)

C    Set the aspect ratio such that the aspect ratio of
C    X:Y:Z is 1:1:1. (For this program this means X:Z:Y).
C    Select an eye point far enough away
C    so as to not have too much perspective.
     IF(ASPECT.LT.1.0.AND.ASPECT.NE.0.0)THEN
       ASPECT = 1.0 / ASPECT
       CALL GWBOX(1.,1.,ASPECT)
     ELSE
       IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
        CALL GWBOX(ASPECT,1.,1.)
       ELSEIF (KORSSL .EQ. 'R' .OR. KORSSL .EQ. 'r') THEN
         CALL GWBOX(1.,ASPECT,1.)
       ELSE
         CALL GWBOX(1.,1.,ASPECT)
```

```
      ENDIF
    ENDIF
    CALL GEYE(22.,-20.,20.)

C   GFOCUS is the direction of view (default 0,0,0)
C    CALL GFOCUS(0.,0.,-9999999.)
C   Set a text height based on the dimensions of the
C   viewport.
    TXTHGT1=0.035*MIN(0.7*XSIZE,0.8*ZSIZE)
    TXTHGT2=0.1*MIN(0.7*XSIZE,0.8*ZSIZE)
C   Select z up coordinate system.
    CALL GVPROJ(2)
C   Set textfont for numeric axis labels.
    CALL RAXTEF(4,'ITAL',0)
C   Set textfont for axis text.
    CALL RAXTEF(6,'ITAL',0)
C   Suppress plotting of first numeric label.
c    CALL RAXDIS(4,1,1)
C   Mark axis texts for display. They are not displayed
C   by default. However, numeric axis labels
C   are displayed by default.
    CALL RAXDIS(6,1,0)
C   Plot 3D axis system.
    CALL RAXIS3(0,1.25*TXTHGT2,IPLANE,LENGTH,TXTSTR)

    DKF = 2.*K1NYQ/(NK-1)
    CALL RCLASS(CHTS,NCHTS,0)
    CALL GCONWI(-1.0,1)
    CALL GCONCO(1,1)
 99 IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
     IFPL = 1
     ILPL = NK
    ELSEIF (KORSSL .EQ. 'R' .OR. KORSSL .EQ. 'r') THEN
     INTPL = -INTPL
     IFPL = NK
     ILPL = 1
    ELSE
     IFPL = 1
     ILPL = NLF
     DKF = FMAX/(NLF-1)
     K1NYQ = 0.
    ENDIF
    IF (NSLICE.EQ.1) THEN
     IFPL = INT(NK/2)+1
     ILPL = INT(NK/2)+1
     INTPL = 1
    ENDIF
C   Set up the 3D scaling mode by calling RWALL. The
C   Y level for the cross section within the workbox
C   is also set.

    DO 22 IY = IFPL,ILPL,INTPL
     YLV = (IY-1)*DKF - K1NYQ
     PRINT*,YLV
     IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
      CALL RWALL(2,3,1,YLV)
     ELSEIF (KORSSL .EQ. 'R' .OR. KORSSL .EQ. 'r') THEN
      CALL RWALL(1,3,2,YLV)
     ELSE
      CALL RWALL(1,2,3,YLV)
     ENDIF
     CALL GPLINT(0.5*TXTHGT1,1)
     IF (KORSSL .EQ. 'Q' .OR. KORSSL .EQ. 'q') THEN
      DO 113 J = 1,NRK
       DO 113 I = 1,NSK
        KKPLO(I,J)=FKDATA(J,I,IY)
 113    CONTINUE
      CALL GCNR2S(KKPLO,NSK,NRK)
     ELSE
      DO 111 J = 1,NLF
       DO 111 I = 1,NK
        IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
         TEMPLO(I,J)=FKDATA(I,IY,J)
        ELSE
         TEMPLO(I,J)=FKDATA(IY,I,J)
        ENDIF
 111    CONTINUE
      CALL GCNR2S(TEMPLO,NK,NLF)
     ENDIF
 22 CONTINUE
    IF (NSLICE.EQ.1.AND.ICOUNT.EQ.0) THEN
      ICOUNT = 1
      IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
       KORSSL = 'r'
      ELSE
       KORSSL = 's'
      ENDIF
      GO TO 99
    ENDIF
C   The Z level of the workbox is set to 0.0.
C   Plot an outline of one wall of the 3d-work box.
c    CALL RWALL(1,2,3,0.0)
c    CALL G3DBOX(0)
C DRAW CONTOUR KEY
    CALL GSCAMM
    CALL RTXFON('MATR',2)
    CALL RTXHEI(5.)
c    CALL RTXFON('SWIM',1)
    CALL GCLOPT(LENAR2,TXTAR2,0.7*TXTHGT1,1,0.0,1)
    IF (ILEG.EQ.1) THEN
     IF (KORSSL .EQ. 'S' .OR. KORSSL .EQ. 's') THEN
      CALL GCOSCL(0.6*XSIZE,0.7*ZSIZE)
     ELSE
C       CALL GCOSCL(0.6*XSIZE,0.1*ZSIZE)
      CALL GCOSCL(0.65*XSIZE,0.1*ZSIZE)
     ENDIF
    ENDIF

C DRAW TITLE
    CALL FDATE(DATE)
    CALL RTXSPM(0)
    CALL RTXPAT(0)
    CALL RTXJUS(1,1)
    CALL RTXHEI(3.5)
c    CALL RTX(-1,DATE,XSIZE,RESERV-120)
    CALL RTXFON('ITAL',0)
    ZTITLE = FMAX + 100
    CALL RTXHEI(5.0)
C    CALL RTX(-1,'F-K Amplitude Spectrum (linear)',0,ZTITLE)
    CALL GEND
    RETURN
    END

c---------------------------------------------------------------
    SUBROUTINE FKPLOT(NK, NF, FKDATA, DT, DZ, FMAX,
   1 NCHTS, CHTS, LOGYN, YN3D, SQRDAT)
C.  Routine to plot F-K amp. spectrum contained in ARR(NK,NF)
C   N.B. Sample order in K  --->  1       = -knyq
C       nk/2 + 1  = 0.0      nk        = +knyq
C   NF, NK   no. of frequencies , wavenumbers for plot
C   DT, DZ   temporal/spatial sampling intervals (microsecs/metres)
C   NCHTS    no. of contour levels
    INTEGER NK, NF, LENAR1(6), LENAR(4), LENAR2(3), N10
    INTEGER IPLANE(3)
    REAL DT, DZ, KNYQ, KMIN, KMAX, FMIN, CHTS(NCHTS)
    REAL AMAX, AMIN, A1, TRACE(2100),FKDATA(NK,NF)
    CHARACTER LOGYN*1, TXTAR1(6)*10, TXTAR(4)*1
    CHARACTER DATE*24, SQRDAT*1,TXTAR2(3)*5, YN3D*1
    DATA IPLANE /1,1,1/
    DATA LENAR /4*0/
    DATA TXTAR /4*''/
    DATA LENAR1 /6*-1/
    DATA TXTAR1 /'k (/m)','Wavenumber','f (Hz)','Frequency'
   1   ,'units','Amplitude'/
```

```
      DATA LENAR2 /5,5,0/
      DATA TXTAR2 /'Below','Above',''/
      RESERV = 30.0
      FNYQ = .5E6 / DT
      KNYQ = .5 / DZ
      FMIN = 0.0
      KMIN = -KNYQ
      KMAX = -KMIN
      AMAX = 0.
      AMIN = 0.
      NLF = 1+NINT(FMAX*NF/FNYQ)
      IF (YN3D .EQ. 'Y') THEN
       DO 10 I=1,NK
        DO 20 J=1,NLF
         TRACE(J)=FKDATA(I,J)
 20     CONTINUE
        CALL MAXSN(NLF, TRACE, A1, II)
        IF (A1 .GT. AMAX) AMAX = A1
 10    CONTINUE
       PRINT*, 'Max amp = ', AMAX
       N10=0
 30    IF (AMAX .GT. 10.) THEN
        AMAX=AMAX/10
        N10=N10+1
        GOTO 30
       ENDIF
       AMAX=10**N10*INT(AMAX+1)
      ENDIF

C START PLOTTING ROUTINES
C  UNIRAS CALLS
      CALL GBEGIN('sel mx11;e','crap','fk.pic')
C  Plot size
      CALL GRPSIZ(XSIZE,ZSIZE)
      XOFF = RESERV
      ZOFF = RESERV
      CALL GLIMIT(KMIN,KMAX,FMIN,FMAX,AMIN,AMAX)
      CALL GVPORT(XOFF,ZOFF,XSIZE*.6,ZSIZE*.8)
      IF (YN3D .EQ. 'N'.OR.YN3D .EQ. 'n') THEN
       CALL GWBOX(1.0,1.0,0.0)
      ELSE
       CALL GWBOX(1.0,1.0,0.5)
      ENDIF
      CALL GSCALE
      CALL RCLASS(CHTS,NCHTS,0)
      CALL GCONWI(-1.0,1)
      CALL GCONCO(1,1)
      HEIGHT = 0.05*MIN(70.0,220.0)
      CALL RAXDIS(2,1,0)
      CALL RAXDIS(6,1,0)
      IF (YN3D .EQ. 'N') THEN
       CALL GCNR2S(FKDATA,NK,NLF)
       CALL RAXIS2(FMIN,KMIN,HEIGHT,LENAR1,TXTAR1)
       CALL RAXDIS(4,0,0)
       CALL RAXIS2(FMAX,KMAX,HEIGHT,LENAR,TXTAR)
      ELSE
       CALL GCONLI(1.0,0,0,2.0,2.0,0.1,1)
       CALL GVPROJ(2)
       CALL GCONR3(FKDATA,NK,NLF)
       CALL RAXIS3(1,HEIGHT,IPLANE,LENAR1,TXTAR1)
       CALL RAXIS3(2,HEIGHT,IPLANE,LENAR1,TXTAR1)
       CALL RAXIS3(3,HEIGHT,IPLANE,LENAR1,TXTAR1)
      ENDIF
C DRAW CONTOUR KEY
      CALL RTXFON('SWIM',1)
      CALL GCLOPT(LENAR2,TXTAR2,1.75*HEIGHT,1,0.0,1)
      CALL GCOSCL(KMAX*1.1,FMIN)
C DRAW TITLE
      CALL FDATE(DATE)
      CALL RTXSPM(0)
      CALL RTXPAT(0)
      CALL RTXJUS(1,1)
      CALL RTXHEI(3.5)
      CALL RTXFON('ITAL',0)
      ZTITLE = FMAX + 100
      CALL RTXHEI(5.0)
      IF (LOGYN .EQ. 'Y') THEN
c      CALL RTX(-1,'F-K Amplitude Spectrum (dB)',
c     1     0,ZTITLE)
      ELSE
       IF (SQRDAT .EQ. 'Y') THEN
c       CALL RTX(-1,'F-K SQRT Amp Spectrum (linear)',
c     1      0,ZTITLE)
       ELSE
c       CALL RTX(-1,'F-K Amplitude Spectrum (linear)',
c     1      0,ZTITLE)
       END IF
      END IF
      CALL GEND
      RETURN
      END

c-----------------------------------------------------------------
      SUBROUTINE KKPLOT(NK1, NK2, KKDATA, DZ1,
     1 DZ2, NCHTS,CHTS,SQRDAT)
C.  Routine to plot K-K amp. spectrum contained in ARR(NK1,NK2)
C   N.B. Sample order in K  --->  1         = -knyq
C     nk/2 + 1   = 0.0  nk      = +knyq
C   NK2, NK1  no. of wavenumbers for plot
C   DZ1, DZ2  spatial sampling intervals in metres
C   NCHTS   no. of contour levels
      INTEGER NK1, NK2, LENAR1(6), LENAR(4), LENAR2(3)
      INTEGER IPLANE(3)
      REAL DZ1, DZ2, K1NYQ, K1MIN, K1MAX, CHTS(NCHTS)
      REAL KKDATA(NK1,NK2)
      REAL AMAX, AMIN,K2NYQ, K2MIN, K2MAX
      CHARACTER TXTAR1(6)*10, TXTAR(4)*1,TXTAR2(3)*5
      CHARACTER DATE*24, SQRDAT*1
      DATA IPLANE /1,1,1/
      DATA LENAR /4*0/
      DATA TXTAR /4*''/
      DATA LENAR1 /6*-1/
      DATA TXTAR1 /'ks (/m)','Wavenumber','kr (/m)','Wavenumber'
     1    ,'units','Amplitude'/
      DATA LENAR2 /5,5,0/
      DATA TXTAR2 /'Below','Above',''/
      RESERV = 30.0
      K2NYQ = .5 / DZ1
      K1NYQ = .5 / DZ2
      K1MIN = -K1NYQ
      K1MAX = -K1MIN
      K2MIN = -K2NYQ
      K2MAX = -K2MIN
      AMAX = 0.
      AMIN = 0.
C  START PLOTTING ROUTINES
C  UNIRAS CALLS
      CALL GBEGIN('sel mx11;e','crap','kk.pic')
C  Plot size
      CALL GRPSIZ(XSIZE,ZSIZE)
      XOFF = RESERV
      ZOFF = RESERV
      CALL GLIMIT(K1MIN,K1MAX,K2MIN,K2MAX,AMIN,AMAX)
      CALL GVPORT(XOFF,ZOFF,XSIZE*.6,ZSIZE*.8)
      CALL GWBOX(1.0,1.0,0.0)
      CALL GSCALE
      CALL RCLASS(CHTS,NCHTS,0)
      CALL GCONWI(-1.0,1)
      CALL GCONCO(1,1)
      HEIGHT = 0.05*MIN(70.0,220.0)
      CALL RAXDIS(2,1,0)
      CALL RAXDIS(6,1,0)
      CALL GCNR2S(KKDATA,NK1,NK2)
      CALL RAXIS2(K2MIN,K1MIN,HEIGHT,LENAR1,TXTAR1)
```

```
      CALL RAXDIS(4,0,0)
      CALL RAXIS2(K2MAX,K1MAX,HEIGHT,LENAR,TXTAR)
C DRAW CONTOUR KEY
      CALL RTXFON('SWIM',1)
      CALL GCLOPT(LENAR2,TXTAR2,1.75*HEIGHT,1,0,0,1)
      CALL GCOSCL(K1MAX*1.1,K2MIN)
C DRAW TITLE
      CALL FDATE(DATE)
      CALL RTXSPM(0)
      CALL RTXPAT(0)
      CALL RTXJUS(1,1)
      CALL RTXHEI(3.5)
c     CALL RTX(-1,DATE,XSIZE,RESERV-120)
      CALL RTXFON('ITAL',0)
      ZTITLE = K2MAX + 100
      CALL RTXHEI(5.0)
      IF (SQRDAT .EQ. 'Y') THEN
c       CALL RTX(-1,'K-K SQRT Amp Spectrum (linear)',0,ZTITLE)
      ELSE
c       CALL RTX(-1,'K-K Amplitude Spectrum (linear)',0,ZTITLE)
      END IF
      CALL GEND
      RETURN
      END

C-----------------------                    -----------------------
C                   FAST FOURIER TRANSFORM
C-----------------------                    -----------------------
C
C Subroutine name : FFT
C
C Description :
C         This subroutine does a fast Fourier transform using
C
C                  LX
C           CX(K)=SQRT(1/LX)
C     SUM(CX(J)*EXP(2*PI*SIGNI*I*(J-1)*(K1)/LX))
C                  J=1
C
C         for K=1,2,...,LX2      (LX2 is 2**N)
C
C Arguments :
C       LX - Length of input string. (Must be a power of 2)
C       CX - Input string on entry, transform on exit.
C     SIGNI - If -1 will transform from time to frequency domain.
C             If +1 will transform from frequency to time domain.
C
C *********************
C * THIS ROUTINE EXPECTS SINGLE PRECISION ARGUMENTS
C *  THE ARRAY CX IS COMPLEX.              ***********
C ******************
C Subroutines called : None
C Source : Claerbout
      SUBROUTINE FFT(LX,CX,SIGNI)
      COMPLEX CX(LX),CARG,CEXP,CW,CTEMP
      J=1
      SC=SQRT(1.0/FLOAT(LX))
      DO 30 I=1,LX
      IF(I.GT.J)GO TO 10
      CTEMP=CX(J)*SC
      CX(J)=CX(I)*SC
      CX(I)=CTEMP
10    M=LX/2
20    IF(J.LE.M)GO TO 30
      J=J-M
      M=M/2
      IF(M.GE.1)GO TO 20
30    J=J+M
      L=1
40    ISTEP=2*L
      DO 50 M=1,L
      CARG=(0.0,1.0)*(3.14159265*SIGNI*(M-1))/FLOAT(L)
      CW=CEXP(CARG)
      DO 50 I=M,LX,ISTEP
      CTEMP=CW*CX(I+L)
      CX(I+L)=CX(I)-CTEMP
50    CX(I)=CX(I)+CTEMP
      L=ISTEP
      IF(L.LT.LX)GO TO 40
      RETURN
      END

c-----------------------------------------------------------------
      SUBROUTINE FFT2D(NSAMS,N2,CDAT,SIGNA,SIGNB,CW3)
C     2-d fft using fork
C     Signa -1.  t > f
C     Signb -1.  x > k
      COMPLEX CDAT(NSAMS,N2),CW3(N2)
      REAL SIGNA,SIGNB
      DO 100 I = 1,N2
c          PRINT*,'1d fft pass (1-64)',I
         CALL FFT(NSAMS,CDAT(1,I),SIGNA)
 100  CONTINUE
      DO 400 J = 1,NSAMS
c          PRINT*,'2nd d (1-512)',J
         DO 500 I = 1,N2
            CW3(I) = CDAT(J,I)
500      CONTINUE
         CALL FFT(N2,CW3,SIGNB)
         DO 600 I = 1,N2
            CDAT(J,I) = CW3(I)
600      CONTINUE
400   CONTINUE
      RETURN
      END

c-----------------------------------------------------------------
      SUBROUTINE FFT3D(NSZ,NSX,NSY,CDAT,SIGNZ,
     1 SIGNX, SIGNY,CW)
      COMPLEX CDAT(NSZ,NSX,NSY),CW(NSX)
      DO 10 IY = 1,NSY
        PRINT*,'2d fft pass (1-',NSY,')',IY
        CALL FFT2D(NSZ,NSX,CDAT(1,1,IY),SIGNZ,SIGNX,CW)
 10   CONTINUE
      PRINT*,'Started 3rd d'
      DO 20 IZ = 1,NSZ
c        PRINT*,'3rd d (1-512)',IZ
         DO 30 IX = 1,NSX
            DO 40 IY = 1,NSY
               CW(IY) = CDAT(IZ,IX,IY)
40          CONTINUE
            CALL FFT(NSY,CW,SIGNY)
            DO 50 IY = 1,NSY
               CDAT(IZ,IX,IY) = CW(IY)
50          CONTINUE
30       CONTINUE
20    CONTINUE
      RETURN
      END

c-----------------------------------------------------------------
      integer function lnb(string)
c-t determines last non blank character in a string
c-a dave stevenson 1986
c-l fortran77
c-d+
c  scans the string starting at the last column and checks each
c  character in turn to see if it is a space, if not it returns
c  the character position number. if all string is blank then
c  zero is returned.
c  routines called:-
c     len
c-d-
      character*(*) string
```

```
      l=len(string)
      lnb=0
      do 100 i=l,1,-1
        if(string(i:i).ne.' ') then
          lnb=i
          goto 10
        else
        endif
100   continue
10    return
      end

C------------------------                    ---------------------
C                  MAXIMUM ELEMENT OF ARRAY
C Subroutine name : MAXSN
C Description :
C           This subroutine finds the maximum element of an array.
C Arguments :
C        LX - Length of input time series in samples.
C         X - Array containing input time series.
C        XM - Value of largest element.
C        II - Subscript of largest element.
C Subroutines called : None   Source : Robinson
      SUBROUTINE MAXSN(LX,X,XM,II)
      IMPLICIT REAL (A-H,O-Z)
      DIMENSION X(LX)
      II=1
      DO 10 I=1,LX
10    IF(X(II).LT.X(I))II=I
      XM=X(II)
      RETURN
      END

C--------------------                        ---------------------
C                 SET ELEMENTS OF ARRAY TO ZERO
C Subroutine name : ZERO
C Description :
C           This subroutine sets all the elements of the
C           input array to zero. (Real elements only)
C Arguments :
C        LX - Length of time series in samples.
C         X - Array containing time series.
C Subroutines called : None
C Source : Robinson
      SUBROUTINE ZERO(LX,X)
      IMPLICIT REAL (A-H,O-Z)
      DIMENSION X(LX)
      IF(LX.LE.0)RETURN
      DO 10 I=1,LX
10    X(I)=0.0E0
      RETURN
      END
```

Appendix E
berrymig

This program was adapted from **kirchmig** and performs GK, GB and GRT migration, raytracing by the layer and boxel method, and image point dip calculations. Also given are raytra.f (layer raytracing) and raytl4.f (boxel raytracing). The results are plotted with **nscap.f** (not given), which uses UNIRAS. Subroutines are in **libkir.a** in the dgl3psr user area on the University of Durham Geological Sciences Department's SUN system.

```
C .  Program to do Kirchhoff migration using RAYTRA anisotropic
C  raytracing algorithm  Allows general source&receiver positions
C  Options of different diffraction stacks: Diffraction stack, Kirchhoff,
C  GRT,GK,GB   Selective raypath muting for shallow-dipping rays, and
C  geological dip range   Needs subroutine RAYTRA RAYT14 for
C  raytracing subroutine library libtsa.a  (Time series analysis -- Robinson,
C                                  Claerbout et al.)
C NB Roses and dip-weighting may not work if mute angle (APRMUT)
C  .NE.aperture angle (APRANG)
     PROGRAM BERRYMIG
C  DEPTH(ilay) = depth to top of layer ilay
C  VH(ilay) = horizontal velocity in layer ilay
C  VV(ilay) = vertical velocity in layer ilay
C  (XSOR(isor),ZSOR(isor)) = coordinates of source isor
C  (XREC(irec),ZREC(irec)) = coordinates of receiver irec
C  CSG(itime,irec,isor) = recorded data
C  IMAGE(idepth,ioffset) = IMAGED reflectivity at (idepth,ioffset)
C  VERSION 4.0 WITH FASTER RAYTRACING CODE
C  NIMAGX,NIMAGZ must equal NX,NZ (see 21)
c  PARAMETER (NSM=512,NSOR1=1,NREC1=23,NIMAGZ=249,
C     1  NIMAGX=24)
c      PARAMETER (NSM=512,NSOR1=1,NREC1=51,NIMAGZ=249,
C     1  NIMAGX=24)
c      PARAMETER (NSM=512,NSOR1=22,NREC1=23,NIMAGZ=149,
C     1  NIMAGX=82)
c      PARAMETER (NSM=512,NSOR1=1,NREC1=23,NIMAGZ=199,
C     1  NIMAGX=75)
c      PARAMETER (NSM=512,NSOR1=3,NREC1=22,NIMAGZ=199,
C     1  NIMAGX=38)
c      PARAMETER (NSM=512,NSOR1=22,NREC1=23,NIMAGZ=199,
C     1  NIMAGX=19)
c  PARAMETER (NSM=2001,NSOR1=1,NREC1=125,NIMAGZ=399,
C     1  NIMAGX=34)
c      PARAMETER (NSM=2001,NSOR1=1,NREC1=63,NIMAGZ=399,
C     1  NIMAGX=68)
c      PARAMETER (NSM=2001,NSOR1=1,NREC1=62,NIMAGZ=399,
C     1  NIMAGX=34)
     PARAMETER (NSM=512,NSOR1=51,NREC1=51, NIMAGZ=249,
    1  NIMAGX=24)
     REAL DEPTH(15), VH(15), VV(15), XSOR(NSOR1)
     REAL ZSOR(NSOR1),CSG(NSM,NREC1,NSOR1)
     REAL XREC(NREC1), ZREC(NREC1), IMHOLD(1200)
     REAL IMAGE(NIMAGZ,NIMAGX), TEMP(NSM+30), ZEND
     REAL TAUS(NSOR1), TAUR(NREC1), DTOTR(NREC1)
     REAL DTOTS(NSOR1),DIPARR(1200),ANG0(NSOR1)
     REAL ANG1(NREC1),ANGS(NSOR1), ANGG(NREC1)
     REAL ROSEXS(100), ROSEZS(100), DIP(600,1200)
     REAL SL(1600), VEL(1600), TK(1600), COORDS(6), SLM(1600)
     INTEGER NDIP(1200),XROS(1200),ZROS(1200),
     INTEGER NORAYS(1600)
     COMPLEX CX(NSM), C0, C1
     CHARACTER*25 FNAME, FROSE, VNAME*8, VLABEL(9)*72
     CHARACTER*1 A(256)
     PI = 3.1415926535
     PI2 = PI / 2.
     NTMAX = 0
     DO 10 I = 1, 256
       A(I) = '@'
 10  CONTINUE
     NIMAGE = NIMAGZ * NIMAGX
     CALL ZERO(NIMAGE, IMAGE)
C  SET UP VELOCITY MODEL
     WRITE (6,*)'Which method of raytracing(1=layers,2=grid)?'
     READ (5,*) METHR
     IF (METHR.EQ.2) THEN
       WRITE (6,*)'Include straight rays (1=y)?'
       READ (5,*) ISTINC
       PRINT*,'Most parameters are read from the following file.'
C  SET UP INITIAL VELOCITY FIELD*****************
       WRITE(6,9991)'&INPUT FILE NAME OF VELOCITY GRID>'
9991   FORMAT(A50)
       READ(5,9992) VNAME
9992   FORMAT(A8)
       CALL READV(VNAME,VEL,ICELLS,ICELLZ,ICELLX,
    1  VLABEL,NLABEL,BOXX,BOXZ,XMIN,ZMIN,XMAX,ZMAX)
       ICELXZ = ICELLX *(ICELLZ+1)
       DO 9993,I=1,ICELLS
         SL(I)=1.0/VEL(I)
9993   CONTINUE
       DO 9994,I=ICELLS+1,ICELXZ
         SL(I)=SL(I-ICELLX)
9994   CONTINUE
       ICOUNT=0
       DO 171 J=1,ICELLZ+1
         DO 172 I=1,ICELLX
           SLM(I+ICOUNT)=SL(ICELLX-I+ICOUNT+1)
172      CONTINUE
         ICOUNT=ICOUNT+ICELLX
171    CONTINUE
       PRINT*,'Enter constraint required for selective smoothing'
       READ(5,*) CONSTR
       PRINT *,'Angle (rads) for shallow ray arc of circle subroutine'
       PRINT *,'(Default=0.0) '
       READ(5,*) ASD
       IF(ASD.LT.0.001) ASD=0.0
       ASTEP=0.20
       PRINT *,'Ray search angle increment (radians, default=0.2) :'
       READ(5,*) ASTEP
       IF(ASTEP.LT.0.001) ASTEP=0.2
       PRINT *,'Maximum no. of trial rays to be used in each search :'
       READ(5,*) NTRIES
       WRITE (6,*),'Required accuracy to receiver points :'
       READ(5,*) ACC
       WRITE (6,*)'ENTER GRID POINT SPACINGS (DX,DZ) :'
       READ (5,*) DXGRID, DZGRID
C        WRITE (6,*)'ENTER MIN AND MAX X,Z VALUES :'
C        READ (5,*) XMIN, XMAX, ZMIN, ZMAX
       WRITE (6,*)'Enter Imaging grid area (X0,X1,Z0,Z1) :'
       READ (5,*) XIM0, XIM1, ZIM0, ZIM1
C 2/3/92 have inserted this line as we are imaging eg 2-48.5, not 0-46.5
C XMAX not actually used
       XMIN = XIM0
       XMAX = XIM1
       NX = INT((XIM1 - XIM0)/DXGRID) + 1
       NZ = INT((ZIM1 - ZIM0)/DZGRID) - 1
C  x grid
c      ICELLX = INT ((XMAX -XMIN)/ BOXX - 1.0E-5) + 1
C  z grid
c      ICELLZ = INT ((ZMAX -ZMIN)/ BOXZ + 1.0E-5) + 1
C  ADDING A SMALL AMOUNT TO Z ALLOWS FOR RAYS
C  TRAVELLING ALONG THE LOWER BOUNDARY WHICH ARE
C  PUT INTO A FURTHER ROW OF CELLS BY BOXER **
       IF (MOD (ICELLZ, 2).NE.0) THEN
         ICELLZ = ICELLZ + 1
       ENDIF
C  total no of cells
c       ICELXZ = ICELLX *(ICELLZ+1)
c       ICELLS = ICELLX *(ICELLZ)
     ELSE
       WRITE (6,*)'ENTER NO. OF LAYERS :'
       READ (5,*) NLAMAX
       NDEPTH = NLAMAX + 1
       DEPTH(1) = 0.0
       DO 20 J = 1, NLAMAX
         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  SET UP RAYTRACING PARAMETERS
       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
 21   NX = INT((XMAX - XMIN)/DXGRID) + 1
      NZ = INT(DEPTH(NDEPTH)/DZGRID) - 1
     ENDIF
C SET UP SOURCE AND RECEIVER COORDINATES
     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 rec posns wanted, no in file :'
     READ (5,*) NREC,NRFILE
     DO 40 J = 1, NREC
       WRITE (6,*)'ENTER COORDINATES OF RECEIVER ', J, ' :'
       READ (5,*) XREC(J), ZREC(J)
 40 CONTINUE
     CALL MAXSN(NSOR, ZSOR, ZSBOT, II)
     CALL MINSN(NSOR, ZSOR, ZSTOP, II)
     ZBOT=ZSBOT
     ZTOP=ZSTOP
     CALL MAXSN(NREC, ZREC, ZRBOT, II)
     CALL MINSN(NREC, ZREC, ZRTOP, II)
     IF (ZRBOT .LT. ZBOT) ZBOT=ZRBOT
     IF (ZRTOP .GT. ZTOP) ZTOP=ZRTOP
c      OR if you include reflections from the zone between top rec
c      and source for up, or bot rec and source for down
c     IF (ZRBOT .GT. ZBOT) ZBOT=ZRBOT
c     IF (ZRTOP .LT. ZTOP) ZTOP=ZRTOP
      DSG = SQRT((ZREC(2) - ZREC(1))**2 +(XREC(2)-XREC(1))**2)
      IF (NREC .EQ. 1) DSG = 0.0
      DSS = SQRT((ZSOR(2) - ZSOR(1))**2 + (XSOR(2)-XSOR(1))**2)
      IF (NSOR .EQ. 1) DSS = 0.0
      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
C  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=Kir stack,2=Kir int,3=GK,
     1 4=GRT,5=CDP gath, 9=Berryhill'
      READ (5,*) IMIGTY
      IF (IMIGTY .EQ. 5) NX = 1
      WRITE (6,*)'ENTER stacking cutoff :'
      READ (5,*) STKOFF
      WRITE (6,*)'DIPS OF SOR AND REC ARRAYS IN DEGS :'
      READ (5,*) DIPS, DIPR
      DIPS = DIPS * PI / 180.
      DIPR = DIPR * PI / 180.
C  READ IN DATA FILE
      WRITE (6,*)'Enter name of datafile to migrate : '
      READ (5,50) FNAME
 50 FORMAT (A25)
      WRITE (6,*)'Header IN/OUT of file : '
      READ (5,51) IHFORM
 51 FORMAT (A3)
      IF (IHFORM .EQ. 'OUT') FNAME = FNAME(1:lnb(FNAME))//'.d'
      WRITE (6,*)'UP (1) or DOWN (0) -going wavefield :'
      READ (5,*) IUP
      IF (FNAME .EQ. 'DUMMY') THEN
C  INVERSE IMPULSE RESPONSE
        WRITE (6,*)'Enter coordinates of diffracting point :'
        READ (5,*) XDIF, ZDIF
        DO 70 K = 1, NSOR
          D0 = 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
 60       CONTINUE
 70     CONTINUE
      ELSE
        WRITE (6,*)'Enter no. of samples per trace & NO. NEEDED :'
        WRITE (6,*)'(Include extra for static correction) '
        READ (5,*) NSAMR, NSAM0
        WRITE (6,*)'Enter sampling interval in millisecs :'
        READ (5,*) DT
        DT = DT * 1.E-3
        IF (NSAMR.LT.1024.AND.IHFORM.NE.'OUT') THEN
          LEN = 4104
        ELSE
          LEN = (NSAMR + 2) * 4
        ENDIF
        WRITE (6,*)'Enter I.D. of first csg, first rec trace:'
        READ (5,*) IDSHOT,IRECFI
        WRITE (6,*)'Enter spreading correction (power of T) to apply:'
        READ (5,*) TPOWER
        WRITE (6,*)'Static correction integer :'
        READ (5,*) NCOREC
        WRITE (6,*)'Static correction remainding decimal:'
        READ (5,*) RCOREC
c new option 7/5/93
        WRITE (6,*)'Equate rms energies of csgs (1=y)'
        READ (5,*) IEQRMS
        WRITE (6,*)'Weight the dips? 1=y'
        READ (5,*) IWTDIP
        WRITE (6,*)'Rose diagrams wanted? 1=y'
        READ (5,*) IROSE
        IF (IROSE.EQ.1) THEN
          WRITE (6,*)'Rose positioning...'
          WRITE (6,*)'Max roses = 10x10 ie 100'
          WRITE (6,*)'X increment...'
          READ (5,*) ROSEDX
          WRITE (6,*)'Z increment...'
          READ (5,*) ROSEDZ
          WRITE (6,*)'X start...'
          READ (5,*) ROSEXS(1)
          WRITE (6,*)'Z start...'
          READ (5,*) ROSEZS(1)
          NRX = INT(XIM1/ROSEDX+1)
          NRZ = INT(ZIM1/ROSEDZ)
          DO 71 I = 1,NRX
            ROSEXS(I+1) = ROSEXS(I) + ROSEDX
 71       CONTINUE
          DO 72 I = 1,NRZ-1
            ROSEZS(I+1) = ROSEZS(I) + ROSEDZ
 72       CONTINUE
          WRITE (6,*)'Roses output filename'
          READ (5,50) FROSE
          OPEN (12,FILE=FROSE,STATUS='UNKNOWN')
        ENDIF
        OPEN (10,FILE=FNAME,STATUS='OLD', ACCESS='DIRECT',
     1  FORM= 'UNFORMATTED',RECL=LEN,IOSTAT=K2)
C SUNs like the lines below, HPs do not!
c       OPEN (10,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',
C     1 FORM='UNFORMATTED',RECL=LEN,IOSTAT=K2,ERR=92)
c 92    IF (K2.NE.0) THEN
c         PRINT*,'FILE DOES NOT EXIST'
c         RETURN
```

```
c       ENDIF
      DO 160 J = 1, NSOR
c         CALL RAZOUT('READING SHOT ... ', J)
c         CALL FLUSH(6)
        IF (IHFORM .EQ. 'OUT') THEN
         JREC = NRFILE * (J - 2 + IDSHOT)
        ELSE
         JREC = (NRFILE + 1) * (J - 2 + IDSHOT) + 1
        ENDIF
        RMS = 0.0
c maybe insert a line here to read headers
        DO 90 K = 1, NREC
          KREC = JREC + IRECFI-1 + K
        READ (10,REC=KREC,IOSTAT=K4) (TEMP(I),I=1,NSAM0)
c        READ (10,REC=KREC,IOSTAT=K4,END=93)
C   1        (TEMP(I),I=1,NSAM0)
c 93     IF (K4.NE.0) PRINT*,'PROBLEM RECORD',KREC,K4
C CORRECT DATA FROM 3-D TO 2D
          DO 80 I = 1, NSAM0
            CSG(I,K,J)=TEMP(I+NCOREC)
            CSG(I,K,J) = CSG(I,K,J) * (REAL(I)) ** TPOWER
            RMS = RMS + CSG(I,K,J) ** 2
 80        CONTINUE
 90      CONTINUE
        RMS = SQRT(RMS/REAL(NSAM0*NREC))
C NOW EQUATE RMS ENERGIES OF CSG
        IF (IEQRMS.EQ.1) THEN
         DO 110 K = 1, NREC
          DO 100 I = 1, NSAM0
           CSG(I,K,J) = CSG(I,K,J) / RMS
 100        CONTINUE
 110       CONTINUE
        ENDIF
C NOW APPLY 1/2-DIFFERENTIAL OPERATOR IF NEEDED
        IF (NEWMAN .EQ. 1) THEN
          FNYQ = .5 / DT
          DF = FNYQ / REAL(NSAM0/2)
          DO 150 K = 1, NREC
            DO 120 I = 1, NSAM0
              CX(I) = CSG(I,K,J)
 120          CONTINUE
            CALL FFT(NSAM0, CX, -1.0)
C FORWARD FFT
            CX(1) = 0.0
            C0 = SQRT(REAL(NSAM0/2)) * (1.,+1.)
            C0 = 0.0
            CX(NSAM0/2 + 1) = CX(NSAM0/2 + 1) * C0
C  Add PI/4 to PHASE AND MULTIPLY CX BY OMEGA**.5 RAMP
            DO 130 I = 2, NSAM0 / 2
              FI = DF * (I - 1)
              C1 = SQRT(REAL(I - 1)) * (1.,+1.)
              IF (FI .GT. 3*FNYQ/4.) THEN
                C1 = SQRT(REAL(NSAM0)*3./8.) * (1.,+1.)
                C1 = C1 * 4. * (FNYQ - FI) / FNYQ
              END IF
              CX(I) = CX(I) * C1
              CX(NSAM0 + 2 - I) = CONJG(CX(I))
 130          CONTINUE
C  INVERSE FFT
            CALL FFT(NSAM0, CX, +1.0)
            DO 140 I = 1, NSAM0
              CSG(I,K,J) = CX(I)
 140          CONTINUE
 150        CONTINUE
        END IF
 160  CONTINUE
      CLOSE (10)
      END IF
C  DATA NOW IN ARRAY CSG(time,receiver,source)
      WRITE (6,*)'Enter name of output file for image :'
      READ (5,50) FNAME
      WRITE (6,*)'Enter I.D. of image to save :'
      READ (5,*) ISHOT
      NTRIES = MAXLOP
      NROSE = 1
      LOST = 0
      IFOUND = 0
C  LOOP OVER EACH IMAGE POINT DEPTH
      DO 220 JJ = 1, NZ
        ZSTART = JJ * DZGRID
        DO 161 I = 1,NRZ
         IF (ZSTART.EQ.ROSEZS(I)) THEN
          JROSEZ = 1
          GOTO 162
         ENDIF
 161    CONTINUE
        JROSEZ = 0
        JROSE = 0
 162    IF (ZSTART .LT. ZIM0 .OR. ZSTART .GT. ZIM1) GO TO 220
C  CHECK FOR UPGOING WAVEFIELD
        IF (IUP .EQ. 1 .AND. ZSTART .LT. ZTOP) THEN
          GO TO 167
C  CHECK FOR DOWNGOING WAVEFIELD
        ELSE IF (IUP .EQ. 0 .AND. ZSTART .GT. ZBOT) THEN
          GO TO 167
        ELSE
          GO TO 168
        ENDIF
 167    IF (JROSEZ.EQ.1) THEN
         DO 165 II = 1,NX
          XSTART = (II - 1) * DXGRID + XMIN
          DO 166 J = 1,NRX
           IF (XSTART.EQ.ROSEXS(J)) THEN
            NROANG = 0
            NDIP(NROSE) = NROANG
            XROS(NROSE) = XSTART
            ZROS(NROSE) = ZSTART
            WRITE (12,*) XROS(NROSE),ZROS(NROSE),
     1 NDIP(NROSE),(DIP(NROSE,I),I=1,NDIP(NROSE))
            NROSE = NROSE+1
           ENDIF
 166      CONTINUE
 165     CONTINUE
        ENDIF
        GO TO 220
C Image point within imaging area
c 168   CALL RAZREAL('Migrating depth ... ', ZSTART)
c       CALL FLUSH(6)
 168    continue
c WRITE (6,*)'Migrating depth ', ZSTART, '....'
C  LOOP OVER EACH IMAGE POINT HORIZONTALLY
        DO 210 II = 1, NX
          XSTART = (II - 1) * DXGRID + XMIN
          IF (JROSEZ.EQ.1) THEN
           DO 163 I = 1,NRX
            IF (XSTART.EQ.ROSEXS(I)) THEN
             JROSE = 1
             NROANG = 0
             GOTO 164
            ENDIF
 163        CONTINUE
           JROSE = 0
 164      ENDIF
          NMUTE = 0
          IF (IMIGTY .EQ. 5) XSTART = XIM0
C  COMMON DEPTH POINT GATHER
c This line commented out because of wanting to image past the receivers
c   IF (XSTART .LT. XIM0 .OR. XSTART .GT. XIM1) GO TO 210
          STACK = 0.0
C  LOOP OVER EACH SOURCE POINT
C  PUT TIMES INTO ARRAY TAUS(nsor), ANGLES into
C    ANG0,ANGS
          CALL ZERO(NSOR, TAUS)
          CALL ZERO(NSOR, ANG0)
```

```
      CALL ZERO(NSOR, ANGS)
      DO 170 KK = 1, NSOR
        XEND = XSOR(KK)
        ZEND = ZSOR(KK)
c         IF (IUP .EQ. 1) THEN
c           IF (ZSTART - 0.0 .LT. ZEND) GO TO 170
C CHECK FOR UPGOING WAVEFIELD
c         ELSE
c           IF (ZSTART + 0.0 .GT. ZEND) GO TO 170
C CHECK FOR DOWNGOING WAVEFIELD
c         END IF
C NOW RAY TRACE TO SOURCE FROM IMAGE POINT
        THETAP = 9.9
        TAU0 = 0.0
       IF(METHR.EQ.2) THEN
        COORDS(4)=BOXX*ICELLX-XSOR(KK)
        COORDS(1)=XSOR(KK)-XSTART+COORDS(4)
        COORDS(3)=ZSTART
        COORDS(6)=ZSOR(KK)
        IWRYN = 0
        ISTRT = 0
        CALL RAYT14(COORDS,SLM,TK,DTOT0,TAU0,
     1 NORAYS,BOXX,BOXZ,ICELLS,NIT, ICELXZ,ICELLX,
     2 ICELLZ,XMAX,ZMAX,ASD,ASTEP,ACC, NTRIES,
     3 NRAY,THETA0,THETAS,IWRYN,ISTRT)
        THETA0=PI-THETA0
        THETAS=PI-THETAS
        IST = IST+ISTRT
        IF (ISTINC.EQ.0.AND.ISTRT.EQ.1) GO TO 170
       ELSE
        CALL RAYTRA(NLAMAX, NDEPTH, DEPTH, VH, VV,
     1 XSTART, ZSTART, XEND, ZEND, DZGRID, ACC, NTRIES,
     2 THETAP, THETA0, TAU0, THETAS, DTOT0, NRAY)
       ENDIF
        IF (NRAY .EQ. NTRIES) THEN
C RAY NOT FOUND
          LOST = LOST + NREC
          GO TO 170
        END IF
        TAUS(KK) = TAU0
        DTOTS(KK) = DTOT0
        ANG0(KK) = THETA0
        ANGS(KK) = THETAS
        IFOUND = IFOUND + 1
 170    CONTINUE
C NOW RAY trace to receivers
       THETAP = 9.9
       CALL ZERO(NREC, TAUR)
      CALL ZERO(NREC, ANG1)
      CALL ZERO(NREC, ANGG)
      DO 180 LL = 1, NREC
        XEND = XREC(LL)
        ZEND = ZREC(LL)
c         IF (IUP .EQ. 1) THEN
c           IF (ZSTART - 0.0 .LT. ZEND) GO TO 180
C CHECK FOR UPGOING WAVEFIELD
c         ELSE
c           IF (ZSTART + 0.0 .GT. ZEND) GO TO 180
C CHECK FOR DOWNGOING WAVEFIELD
c         END IF
       IF(METHR.EQ.2) THEN
        COORDS(1)=XSTART
        COORDS(3)=ZSTART
        COORDS(4)=XREC(LL)
        COORDS(6)=ZREC(LL)
        IWRYN = 0
        ISTRT = 0
        CALL RAYT14(COORDS,SL,TK,DTOT1,TAU1,
     1 NORAYS,BOXX,BOXZ,ICELLS,NIT, ICELXZ, ICELLX,
     2 ICELLZ,XMAX,ZMAX,ASD,ASTEP,ACC, NTRIES,
     3 NRAY,THETA1,THETAG,IWRYN,ISTRT)
        IRT = IRT+ISTRT
        IF (ISTINC.EQ.0.AND.ISTRT.EQ.1) GO TO 180
       ELSE
        CALL RAYTRA(NLAMAX, NDEPTH, DEPTH, VH, VV,
     1 XSTART,ZSTART, XEND, ZEND, DZGRID, ACC, NTRIES,
     2 THETAP, THETA1, TAU1, THETAG, DTOT1, NRAY)
       ENDIF
        IF (NRAY .EQ. NTRIES) THEN
C RAY NOT FOUND
          LOST = LOST + 1
          GO TO 180
        END IF
        TAUR(LL) = TAU1
        DTOTR(LL) = DTOT1
        ANG1(LL) = THETA1
        ANGG(LL) = THETAG
        IFOUND = IFOUND + 1
 180    CONTINUE
C LOOP FOR EACH SOURCE
      DO 200 KK = 1, NSOR
        TAU0 = TAUS(KK)
        DTOT0 = DTOTS(KK)
        THETA0 = ANG0(KK)
        THETAS = ANGS(KK)
        IF (TAU0 .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
        IF (IUP .EQ. 1) THEN
          IF (ZSTART .LT. ZSOR(KK)) GO TO 200
C CHECK FOR UPGOING WAVEFIELD
        ELSE
          IF (ZSTART .GT. ZSOR(KK)) GO TO 200
C CHECK FOR DOWNGOING WAVEFIELD
        END IF
C LOOP FOR EACH RECEIVER
        DO 190 LL = 1, NREC
          TAU1 = TAUR(LL)
          DTOT1 = DTOTR(LL)
          THETA1 = ANG1(LL)
          THETAG = ANGG(LL)
          IF (TAU1 .EQ. 0.0) GO TO 190
          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
            IF (PI2 - THETA1 .GT. THETMU) GO TO 190
          ELSE
            IF (PI2 + THETA1 .GT. THETMU) GO TO 190
          END IF
          IF (IUP .EQ. 1) THEN
            IF (ZSTART .LT. ZREC(LL)) GO TO 190
C CHECK FOR UPGOING WAVEFIELD
          ELSE
            IF (ZSTART .GT. ZREC(LL)) GO TO 190
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)
     1          THEN
c             WRITE(6,*) 'XIANG OUT OF RANGE'
             GO TO 190
            ELSE
             IF (ABS(XIANG - 3.*PI2) .LT. APRMUT)
```

```
     1           NMUTE = NMUTE + 1
                DIPARR(INT(STACK)+1) = XIANG - 3.*PI2
                IF (JROSE.EQ.1) THEN
                 NROANG=NROANG+1
                 DIP(NROSE,NROANG)=XIANG - 3.*PI2
                ENDIF
               END IF
               XIANG = ABS(XIANG - 3.*PI2)
             ELSE IF (XIANG .GT. 0.0) THEN
               IF (ABS(XIANG - PI2) .GT. APRANG)
     1              THEN
c                WRITE(6,*) 'XIANG OUT OF RANGE'
                GO TO 190
               ELSE
                IF (ABS(XIANG - PI2) .LT. APRMUT)
     1           NMUTE = NMUTE + 1
                DIPARR(INT(STACK)+1) = XIANG - PI2
                IF (JROSE.EQ.1) THEN
                 NROANG=NROANG+1
                 DIP(NROSE,NROANG)=XIANG - PI2
                ENDIF
               END IF
               XIANG = ABS(XIANG - PI2)
             ELSE
               IF (ABS(XIANG + PI2) .GT. APRANG)
     1              THEN
c                WRITE(6,*) 'XIANG OUT OF RANGE'
                GO TO 190
               ELSE
                IF (ABS(XIANG + PI2) .LT. APRMUT)
     1           NMUTE = NMUTE + 1
                DIPARR(INT(STACK)+1) = XIANG + PI2
                IF (JROSE.EQ.1) THEN
                 NROANG=NROANG+1
                 DIP(NROSE,NROANG)=XIANG + PI2
                ENDIF
               END IF
               XIANG = ABS(XIANG + PI2)
             END IF
C  Source and receiver found so image appropriate travel time
             TTOT = TAU0 + TAU1
c  Possibly add something in here to compensate for a non integer
c  static correction.
c  eg. if stat corrn 22, then sam 23 moved back to 1, but if stat
c  corrn actually 22.2, then 23->0.8. We can't shift the data though.
c  But if TTOT/DT=1.3->1.49, then it should pick up the second sample
c  which would have been at 1.8, rather than the first. Therefore..
             NT = INT(TTOT/DT+RCOREC) +1
c  or perhaps even better, so that things round up
c                 NT = INT(TTOT/DT+RCOREC+.5) +1
c                 NT = INT(TTOT/DT) + 1
               IF (NT .GT. NSM) THEN
                IF (NT .GT. NTMAX) NTMAX=NT
               ENDIF
c NB that there is no theoretical justification for the ABS brackets
c around the COS in all these options
               IF (IMIGTY .EQ. 1) THEN
C  KIRCHHOFF DIFFRACTION STACK
                 FAC = 1.0
               ELSE IF (IMIGTY .EQ. 2) THEN
C  KIRCHHOFF INTEGRAL
                 IF (TAU1 .EQ. 0.0) GO TO 190
                 THETAG = THETAG - DIPR - PI / 2.
                 FAC = SQRT(TAU0/TAU1) * ABS(COS(THETAG))
               ELSE IF (IMIGTY .EQ. 6) THEN
C  SPLIT KIRCHHOFF INTEGRAL
                 IF (TAU1 .EQ. 0.0) GO TO 190
                 THETAG = THETAG - DIPR - PI / 2.
                 THETAS = THETAS - DIPS - PI / 2.
c to make THETAG and THETAS same polarity within intrabhole space
c
c I've removed the ABS bracket from the COS, as no theoretical
C justification for it to be there.  If you don't like all this, comment out the
C next 7 lines, uncomment the next 6 and comment out the 6 after that
                 IF(XSOR(1) .GT. XREC(1)) THEN
                  THES = PI - THETAS
                  THEG = THETAG
                 ELSE
                  THEG = PI - THETAG
                  THES = THETAS
                 ENDIF
c                 IF ( ABS(XSOR(1)-II*DXGRID) .LE.
c    1                 ABS(XREC(1)-II*DXGRID) ) THEN
c                  FAC = SQRT(TAU0/TAU1) * ABS(COS(THETAG))
c                 ELSE
c                  FAC = SQRT(TAU1/TAU0) * ABS(COS(THETAS))
c                 ENDIF
                 IF ( ABS(XSOR(1)-II*DXGRID) .LE.
     1                ABS(XREC(1)-II*DXGRID) ) THEN
                  FAC = SQRT(TAU0/TAU1) * COS(THEG)
                 ELSE
                  FAC = SQRT(TAU1/TAU0) * COS(THES)
                 ENDIF
               ELSE IF (IMIGTY .EQ. 3 .OR. IMIGTY .EQ. 4)
     1              THEN
C  GK Integral or GRT Integral
                 IF (TAU1 .EQ. 0.0 .OR. TAU0 .EQ. 0.0)
     1             GO TO 190
                 THETAS = THETAS - DIPS - 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))*
     1  SQRT(TAU1/TAU0) + DSG*ABS(COS(THETAG))
     2  *SQRT(TAU0/TAU1))
                 IF (IMIGTY .EQ. 4) THEN
                   SCAT = THETA0 - THETA1
                   CSCAT2 = COS(SCAT/2.0)
                   FAC = FAC * CSCAT2 ** 2
                 END IF
               ELSE IF (IMIGTY.GE.7.AND.IMIGTY.LE.9) THEN
C  7 = SPLIT BERRYHILL
C  8 = GENERALISED BERRYHILL
C  9 = BERRYHILL
                 IF (TAU1 .EQ. 0.0 .OR. TAU0 .EQ. 0.0)
     1             GO TO 190
c to make THETAG and THETAS same polarity within intrabhole space
c this will give reverse SEG polarity ie. peak = compression
                 IF(XSOR(1) .GT. XREC(1)) THEN
                  THES = PI - THETAS
                  THEG = THETAG
                 ELSE
                  THEG = PI - THETAG
                  THES = THETAS
                 ENDIF
                 QG = 0.0
                 QS = 0.0
c    for the synth wavelet no diff between klimit=20,50,100
                 KLIMIT = 20
                 KL = NT+KLIMIT-1
                 IF (KL .GT. NSM) THEN
                  IF (KL .GT. NTMAX) NTMAX=KL
                 ENDIF
                 DO 191 K = 1,KLIMIT
                  BG1 = SQRT(2*(K+1)*TAU1/DT+(K+1)**2)
                  BG2 = -2*SQRT(2*K*TAU1/DT+K**2)
                  BG3 = SQRT(2*(K-1)*TAU1/DT+(K-1)**2)
                  QG = QG + (BG1+BG2+BG3)*CSG(NT+K-1,LL,KK)
                  BS1 = SQRT(2*(K+1)*TAU0/DT+(K+1)**2)
                  BS2 = -2*SQRT(2*K*TAU0/DT+K**2)
                  BS3 = SQRT(2*(K-1)*TAU0/DT+(K-1)**2)
                  QS = QS + (BS1+BS2+BS3)*CSG(NT+K-1,LL,KK)
```

```
191       CONTINUE
        IF (IMIGTY .EQ. 7) THEN
         IF ( ABS(XSOR(1)-XSTART) .LE.
     1        ABS(XREC(1)-XSTART) ) THEN
          FAC = QG*DSG*COS(THEG)
     1  *SQRT(DTOT0)/DTOT1
         ELSE
          FAC = QS*DSS*COS(THES)*SQRT(DTOT1)/DTOT0
         ENDIF
        ELSE IF (IMIGTY .EQ. 8) THEN
C This if to prevent the polarity switch at the boreholes
         IF ( ((XSOR(1)-XSTART).LT.0.0.AND.
     1        (XREC(1)-XSTART).LT.0.0 ).OR.
     1        ((XSOR(1)-XSTART).GT.0.0.AND.
     1        (XREC(1)-XSTART).GT.0.0) ) THEN
          IF ( ABS(XSOR(1)-XSTART) .LE.
     1         ABS(XREC(1)-XSTART) ) THEN
           THES = PI-THES
          ELSE
           THEG = PI-THEG
          ENDIF
         ENDIF
         FAC = QS*DSS*COS(THES)*SQRT(DTOT1)/DTOT0
     1        + QG*DSG*COS(THEG)*SQRT(DTOT0)/DTOT1
        ELSE
         FAC = QG*DSG*COS(THEG)
     1   *SQRT(DTOT0)/DTOT1
        ENDIF
        END IF
        IF (ABS(XIANG) .LT. (APRANG - APRTAP))
     1        THEN
         RAMP = 1.0
        ELSE
         RAMP = 1.0 - (ABS(XIANG) - (APRANG - APRTAP))
     1        / (APRTAP)
        END IF
C  TAPER KIRCHHOFF OPERATOR
        IF (IMIGTY.GE.7.AND.IMIGTY.LE.9) THEN
         DELTAP = FAC * RAMP
        ELSE
         DELTAP = CSG(NT,LL,KK) * FAC * RAMP
        ENDIF
C  REVERSES SIGN OF CONTRIBUTION FROM DGW
        IF (IWTDIP.EQ.1) THEN
         IMHOLD(INT(STACK)+1) = DELTAP*REAL(IUP*2-1)
        ELSE
         IMAGE(JJ,II) = IMAGE(JJ,II) + DELTAP*
     1       REAL(IUP*2 - 1)
          ENDIF
          IF(ABS(CSG(NT,LL,KK)).GT.0.0)STACK = STACK + 1.0
C  NEXT RECEIVER TRACE
190       CONTINUE
C  NEXT SOURCE POSITION
200     CONTINUE
        IF (JROSE.EQ.1) THEN
         NDIP(NROSE) = NROANG
         XROS(NROSE) = XSTART
         ZROS(NROSE) = ZSTART
         WRITE (12,*) XROS(NROSE),ZROS(NROSE),
     1 NDIP(NROSE),(DIP(NROSE,I),I=1,NDIP(NROSE))
         NROSE = NROSE+1
        ENDIF
        IF (IWTDIP .EQ. 1 .AND. STACK .GT. 0.) THEN
         DO 201 I = 1, 36
          NWTDIP = 0
          TEMPIM = 0.
          ANGHI = (I*10 - 5) * PI/180
          ANGLO = (I*10 - 15) * PI/180
          DO 202 J = 1, STACK
           DO 203 K = 1,5
            IF (DIPARR(J).GE.(PI*(2-1/36))) THEN
             DIPARR(J) = DIPARR(J) - 2*PI
            ELSE IF (DIPARR(J).LT.(-PI/36)) THEN
             DIPARR(J) = DIPARR(J) + 2*PI
            ENDIF
203         CONTINUE
           IF (DIPARR(J) .LT. ANGHI .AND.
     1         DIPARR(J) .GE. ANGLO) THEN
            TEMPIM = TEMPIM + IMHOLD(J)
            NWTDIP = NWTDIP+1
           ENDIF
202        CONTINUE
          IF (NWTDIP .NE. 0) IMAGE(JJ,II) = IMAGE(JJ,II)
     1        + TEMPIM/NWTDIP
201       CONTINUE
        ELSE IF (STACK .GT. STKOFF) THEN
          IMAGE(JJ,II) = IMAGE(JJ,II) / STACK
        ELSE
          IMAGE(JJ,II) = 0.0
        END IF
C  MUTES point if NO contributions from narrow aperture
        IF (NMUTE .EQ. 0) IMAGE(JJ,II) = 0
C  NEXT GRID POSITION
210     CONTINUE
C  NEXT GRID DEPTH
220 CONTINUE
      CLOSE(12)
      IDCODE = (ISHOT - 1) * (NX + 1) + 1
      OPEN (11,FILE=FNAME,STATUS='UNKNOWN',
     1  ACCESS='DIRECT',FORM= '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
      CLOSE (11)
      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
      WRITE (6,*)'NSM =',NSM
      WRITE (6,*)'NTMAX (=0 if less than NSM)=',NTMAX
      WRITE (6,*)'(if >0, then NSM must be > NTMAX)'
      IF(METHR.EQ.2) WRITE(6,*)'straight rays s/r =',IST,IRT
      END

C-----------------------------------------------------------------
c  LAYER RAYTRACING
C
C. SUBROUTINE TO RAYTRACE FROM POINT
C (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 --> 3pi/2) 0.0=horizontal to right
C     THETAL = 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
C     MAXLOP = Maximum no. of raypath searches to be attempted
C    DTOT = Total distance travelled (Peter Rowbotham Aug 1991
C  N.B. NO LATERAL VELOCITY VARIATIONS USED
C  WRITTEN BY M J Findlay  1990
      SUBROUTINE RAYTRA(NLAMAX, NDEPTH, DEPTH, VH, VV,
     1  XSTART, ZSTART,XEND, ZEND, DZGRID, ACC, MAXLOP,
     2  THETAP, THETA0, TAU0,THETAL, DTOT, NRAY)
      REAL DEPTH(NDEPTH), VH(NLAMAX), VV(NLAMAX),
      REAL XINTS(31), DIST(31),TIME(31), THET(31)
      LOGICAL RECLAY
      IF (XSTART .EQ. 2.) THEN
```

```
      ENDIF
      IFLAG = 0
      IF (XSTART .EQ. 0.0 .AND. ZSTART .EQ. 11 .AND. XEND .EQ.
     1 82 .AND.ZEND .EQ. 0.0) IFLAG = 1
      PI = 3.1415926535
      PI2 = PI / 2.
      NRAY = 1
C  CHECK FOR ZERO-LENGTH RAYPATH
      IF (XSTART .EQ. XEND .AND. ZSTART .EQ. ZEND) THEN
        TAU0 = 0.0
        DTOT = 0.0
        GO TO 150
      END IF
C  DETERMINE IMAGE POINT LAYER
      DO 10 NL = 1, NLAMAX
        IF (DEPTH(NL + 1) .GT. ZSTART - DZGRID/100.) THEN
          LAYIMG = NL
          GO TO 20
        END IF
 10   CONTINUE
 20   CONTINUE
C  DETERMINE END POINT LAYER
      DO 30 NL = 1, NLAMAX
        IF (DEPTH(NL + 1) .GT. ZEND - DZGRID/100.) THEN
          LAYSOR = NL
          GO TO 40
        END IF
 30   CONTINUE
 40   CONTINUE
C  RAYTRACE FROM IMAGEPOINT TO SOURCE / RECEIVER
      IF (ABS(DEPTH(LAYIMG + 1) - ZSTART) .LT. DZGRID/100.)
     1     THEN
        IF (ZEND .LT. ZSTART) THEN
          ZSTART = ZSTART - DZGRID / 100.
        ELSE
          ZSTART = ZSTART + DZGRID / 100.
          LAYIMG = LAYIMG + 1
        END IF
      END IF
      IF (ABS(DEPTH(LAYSOR + 1) - ZEND) .LT. DZGRID/100.)
     1 THEN
        IF (ZEND .LT. ZSTART) THEN
          ZEND = ZEND + DZGRID / 100.
        ELSE
          ZEND = ZEND + DZGRID / 100.
          LAYSOR = LAYSOR + 1
        END IF
      END IF
C  TRACING FROM (XSTART,ZSTART) in layer LAYIMG  TO
C  (XEND,ZEND) in LAYSOR(isor)
      IF (THETAP .LT. 2.*PI) THEN
        THETA0 = THETAP
      ELSE IF (XEND .EQ. XSTART) THEN
        THETA0 = PI2
        IF (ZEND .LT. ZSTART) THETA0 = PI + PI2
      ELSE
        THETA0 = (ZEND - ZSTART) / (XEND - XSTART)
        THETA0 = ATAN(THETA0)
        IF (XEND .LT. XSTART) THETA0 = THETA0 + PI
      END IF
      IF (XEND .NE. XSTART) THEN
        IF (THETA0 .GT. PI) THEN
          THETAT = 3. * PI2 + .01
          THETAB = PI - .01
        ELSE IF (THETA0 .GT. PI2) THEN
          THETAT = PI + .01
          THETAB = PI2 - .01
        ELSE IF (THETA0 .GT. 0.0) THEN
          THETAT = -.01
          THETAB = PI2 + .01
        ELSE
          THETAT = -PI2 - .01
          THETAB = .01
        END IF
      ELSE
C  SPECIAL CASE FOR VERTICAL RAYPATHS
        IF (LAYSOR .EQ. LAYIMG) THEN
          D0 = ABS(ZEND - ZSTART)
          TAU0 = D0 / VV(LAYSOR)
          DTOT = D0
        ELSE IF (THETA0 .EQ. PI2) THEN
C  DOWN
          D0 = DEPTH(LAYIMG + 1) - ZSTART
          TAU0 = D0 / VV(LAYIMG)
          DTOT = D0
          DO 60 NL = LAYIMG + 1, LAYSOR - 1
            D0 = DEPTH(NL + 1) - DEPTH(NL)
            T0 = D0 / VV(NL)
            TAU0 = TAU0 + T0
            DTOT = DTOT + D0
 60       CONTINUE
          D0 = ZEND - DEPTH(LAYSOR)
          TAU0 = TAU0 + D0 / VV(LAYSOR)
          DTOT = DTOT + D0
        ELSE
C  UP
          D0 = ZSTART - DEPTH(LAYIMG)
          TAU0 = D0 / VV(LAYIMG)
          DTOT = D0
          DO 70 NL = LAYIMG - 1, LAYSOR + 1, -1
            D0 = DEPTH(NL + 1) - DEPTH(NL)
            T0 = D0 / VV(NL)
            TAU0 = TAU0 + T0
            DTOT = DTOT + D0
 70       CONTINUE
          D0 = DEPTH(LAYSOR + 1) - ZEND
          TAU0 = TAU0 + D0 / VV(LAYSOR)
          DTOT = DTOT + D0
        END IF
        THETA = THETA0
        GO TO 150
      END IF
      ZINTB = 1E9
      ZINTT = -1E9
      DTHETA = 0.05
      TAU0 = 0.0
      DTOT = 0.0
C  NOW START RAYTRACING
      DO 140 I = 1, MAXLOP
        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
          GO TO 140
        END IF
        XPO = XSTART
        ZPO = ZSTART
        THETA = THETA0
        NLAY = LAYIMG
 90     CONTINUE
        IF (ABS(THETA) .LT. 1.E-3 .OR. ABS(THETA - PI) .LT. 1.E-3)
     1       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
    TIME(NLAY) = DIST(NLAY) / VV(NLAY)
    XPO = XPO + DX
    IF (THETA .LT. 0.0 .OR. THETA .GT. PI) THEN
      XINTS(NLAY) = XPO
    ELSE
      XINTS(NLAY + 1) = XPO
    END IF
C FIRST CHECK FOR RAY WITHIN CAPTURE RADIUS
    ERR = SQRT((XPO - XEND)**2 + (ZPO - ZEND)**2)
    IF (ERR .LE. ACC) THEN
C  RAY FOUND
      TAU0 = 0.0
      DTOT = 0.0
      DO 100 NL = 1, NLAMAX
        TAU0 = TAU0 + TIME(NL)
        DTOT = DTOT + DIST(NL)
 100    CONTINUE
      GO TO 150
    END IF
C CHECK FOR RAY PASSING BEYOND ENDPOINT
    IF ((THETA .GT. PI2 .AND. XPO .LT. XEND) .OR. (THETA
   1    .LE.  PI2 .AND. XPO .GT. XEND)) THEN
C FIND INTERSECTION POINT
      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
      IF (ABS(ZINT - ZEND) .LT. ACC) THEN
C RAY FOUND
        DIST(NLAY) = SQRT(DXDASH**2 + DZDASH**2)
        THET(NLAY) = THETA
        TIME(NLAY) = DIST(NLAY) / VV(NLAY)
        TAU0 = 0
        DTOT = 0.0
        DO 110 NL = 1, NLAMAX
          TAU0 = TAU0 + TIME(NL)
          DTOT = DTOT + DIST(NL)
 110      CONTINUE
C RAY FOUND SO RETURN TO MAIN PROGRAM
        GO TO 150
C NEED TO STORE THETA0 FOR INTEGRAL KERNEL
      ELSE
C NEXT RAY (RAY NOT FOUND)
        IF (ZINT .GT. ZINTT .AND. ZINT .LT. ZEND)
   1          THEN
          THETAT = THETA0
          ZINTT = ZINT
        ELSE IF (ZINT .LT. ZINTB .AND. ZINT .GT. ZEND)
   1          THEN
          THETAB = THETA0
          ZINTB = ZINT
        END IF
        THETA0 = .5 * (THETAT + THETAB)
        GO TO 140
      END IF
C CHECK FOR RAY OUT TOP OR BOTTOM & CALCULATE NEW
C   INTERCEPTION POINTS
    ELSE IF (NLAY .EQ. 1 .AND. (THETA .LT. 0.0 .OR. THETA
   1 .GT. PI) ) THEN
C RAY OUT TOP
      DXNEW = XEND - XPO
      IF (ABS(THETA) .EQ. PI2) THEN
        ZINT = -1.E9
      ELSE
        ZINT = ZPO - ABS(DXNEW*TAN(THETA))
      END IF
      IF (ZINT .GT. ZINTT .AND. ZINT .LT. ZEND) THEN
        THETAT = THETA0
        ZINTT = ZINT
      ELSE IF (ZINT .LT. ZINTB .AND. ZINT .GT. ZEND)
   1       THEN
        THETAB = THETA0
        ZINTB = ZINT
      END IF
      THETA0 = .5 * (THETAT + THETAB)
      GO TO 140
    ELSE IF (NLAY .EQ. NLAMAX .AND. (THETA .GT. 0.0 .AND.
   1 THETA .  LT. PI)) THEN
C RAY OUT BOTTOM
      DXNEW = XEND - XPO
      IF (ABS(THETA) .EQ. PI2) THEN
        ZINT = 1.E9
      ELSE
        ZINT = ZPO + ABS(DXNEW*TAN(THETA))
      END IF
      IF (ZINT .GT. ZINTT .AND. ZINT .LT. ZEND) THEN
        THETAT = THETA0
        ZINTT = ZINT
      ELSE IF (ZINT .LT. ZINTB .AND. ZINT .GT. ZEND)
   1       THEN
        THETAB = THETA0
        ZINTB = ZINT
      END IF
      THETA0 = .5 * (THETAT + THETAB)
      GO TO 140
    END IF
C  RAY NOT FOUND AND STILL IN GRID SO APPLY SNELL'S
LAW AND CHECK FOR
C  SUPERCRITICAL RAYS
    V0 = VH(NLAY)
    V00 = VV(NLAY)
    IF (ABS(THETA) .EQ. PI2 .OR. THETA .EQ. 3*PI2) THEN
      NLAY = NLAY + 1
      IF (THETA .LT. 0.0 .OR. THETA .GT. PI) NLAY = NLAY - 2
      GO TO 130
    END IF
    IF (THETA .LT. 0.0 .OR. THETA .GT. PI) THEN
      V1 = VH(NLAY - 1)
    ELSE
      V1 = VH(NLAY + 1)
    END IF
    IF (V0 .GE. V1) GO TO 120
C RAY NOT CRITICAL SO JUMP OUT
    TEM = V0 / V1
    TEM = TEM / SQRT(1 - TEM**2)
    CRIT = ATAN(TEM)
    CRIT = PI2 - CRIT
    IF (ABS(THETA) .LT. CRIT .OR. ABS(THETA - PI) .LT. CRIT)
   1      THEN
      IF (THETA .GT. PI) THEN
        IF (ZPO .LT. ZEND) THEN
          THETAT = THETA0
          ZINTT = ZPO
        ELSE
          THETAB = THETA0
          ZINTB = ZPO
```

```
      END IF
      THETA0 = .5 * (THETAT + THETAB)
    ELSE IF (THETA .GT. PI2) THEN
      IF (ZPO .LT. ZEND) THEN
        THETAT = THETA0
        ZINTT = ZPO
      END IF
      IF (ZPO .GT. ZEND) THEN
        THETAB = THETA0
        ZINTB = ZPO
      END IF
      THETA0 = .5 * (THETAB + THETAT)
    ELSE IF (THETA .GT. 0.0) THEN
      THETA0 = THETA0 + DTHETA
    ELSE IF (THETA .GT. - PI2) THEN
      THETA0 = THETA0 - DTHETA
    END IF
    GO TO 140
C  RAY IS SUPERCRITICAL SO JUMP OUT AND TRY NEXT RAY
      ELSE
C APPLY SNELL'S LAW
 120      NLAYO = NLAY
    NLAY = NLAY + 1
    IF (THETA .LT. 0.0 .OR. THETA .GT. PI) NLAY = NLAY - 2
    V0 = VV(NLAYO)
    V1 = VV(NLAY)
    COST2 = V1 * COS(THETA) / V0
    IF (ABS(COST2) .GT. 1.0) THEN
      WRITE (6,*) NLAYO, NLAY, V0, V1, THETA
      GO TO 140
    END IF
    THETA2 = ABS(ASIN(COST2))
C THETA2 IS NOW ANGLE TO VERTICAL
    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
```

```
 150 CONTINUE
   THETAL = THETA
C DEFINES ANGLE OF RAY AT SECOND POINT
C GO BACK TO MAIN PROG
   RETURN
   END

C------------------------------------------------------------------
c  BOXEL RAYTRACING
C now the second raytracing subroutine this raytracing only applicable
C to tank data set (water in 1st 2,last 1 cell)
c the subroutine is divided into two major sections:
c           1)raytracing of rays from source cells,as these may not
c            be on exact grid positions.
c 00        2)the main raytracing loop for all other cells.
c this second main loop is divided into the following:
c           2i)check for direction of rays...if ray going to left
c            it will try next angle increment.for rays travelling
c            downwards isn is flagged as +1 and for rays travelling
c            upwards isn is flagged -1.
c           2ii)if rays are horizontal ..simple update of path,time etc.
c           2iii)if dynamic source on horizontal mesh line or corner:
c             a)check if arc of circle raytracing required
c             b)routine for rays travelling through the horizontal
c              base of box
c             c)routine for rays travelling through the vertical
c              side of a box
c           2iv)if dynamic source on a vertical mesh line:
c             a)check if arc of circle raytracing required
c             b)routine for rays travelling through the horizontal
c              base of box
c             c)routine for rays travelling through the vertical
c              side of a box
c           3)check where the ray ends and if captured.
c           4)update the take off angle
c              there are two methods of ray search for updating
c            the take off angle.the original method,for which the
c            flag lost is set as 0,is an iterative technique using
c            the previous distance and final angle for calculation
c            of the new angle.this converges very fast (5-10 its')
c            on the required angle but can get caught in loops and
c            generally 5-30% of rays are lost.the second method which
c            is only used when the first method loses a ray(lost is
c            now set to 1)uses a simple step search method.this
c            generally finds an extra 80-90% of the lost rays.
c  ~~~~~~~~~~~~~~~~~~
C  Adapted for use with berryn.f by Peter Rowbotham 91/92
C  Only deals with one source/receiver and image-point pair at one time
```

```
C   same as rayt13.f but with path write to unit 12 09/04/93
C   NB adaptions made to search methods, therefore the above
c   description may be incorrect
    SUBROUTINE RAYT14(COORDS,SL,TK,TOTR,TTOT,
   1  NORAYS,BOXX,BOXZ,ICELLS,INIT, ICELXZ,ICELLX,
   2 ICELLZ,XMAX,ZMAX, ASD,ASTEP,ACC,NTRY,ITRY,
   3  AOUT1,AOUT2,IWRYN,ISTRT)
c COORDS- input coords of start + end of ray       (array)
c TIMES-  input times - not used
c SL-    input cell slowness                    (array)
C TK-    output length of ray in each cell (only nec for tomog) (array)
C TOTR-  output total length of ray
C TTOT-  output total travel time for this ray
C NORAYS- output no of rays passing through each cell (array)
C BOXX-  input cell dimension X
C BOXZ-  input cell dimension Y
c ICELLS- input no of cells
c NIT-   input iteration number
C ICELXZ- input no of cells including a layer below model bottom
c ICELLX- input no of cells in horizontal direction
C ICELLZ- input no of cells in vertical direction
C XMAX-  input max X distance -not used
C ZMAX-  input max Z distance
C ASD-   input angle for shallow ray arc
C ASTEP- input ray search angle increment
C ACC-   input required accuracy to receiver points
C NTRY-  input max trial rays for each search
C ITRY-  output actual number of rays
C AOUT1- output angle at
C AOUT2- output angle at
C AINI-  output take-off angle for ray?
c XTIMES1 is an input file
c
    REAL COORDS(6),TK(1600),SL(1600)
    INTEGER NORAYS(1600),WRITE
    PI=ACOS(-1.)
    PI2=PI/2.
    ISTRT = 0
    TTOT = 0.
    TOTR = 0.
    RLEN = 0.
    TD = 0.
    DIF = 999.
    THETAH = PI/2
    THETAB = -PI/2
    DO 105,I=1,ICELXZ
       NORAYS(I)=0
 105  CONTINUE
```

```
C NOW FOR SEARCH LOOP FOR SOURCE CELLS....AND
C DETERMINING WHICH SEARCH METHOD WILL BE USED
C SET UP FLAGS AND INITIALIZE ANY VARIABLES
C SET PARAMS FOR ITERATIVE SEARCH
   ASTEP1 = 0.009973
C NOW MAIN LOOP .. PER RAYPATH
   IF ( (COORDS(3)-COORDS(6)).EQ.0.0) THEN
    AINI = 0.0
   ELSE
    AINI=ATAN( (COORDS(3)-COORDS(6)) /
  1   (COORDS(1)-COORDS(4)) )
   ENDIF
   WRITE = 0
   XSTART=COORDS(1)
   ZSTART=COORDS(3)
   XEND=COORDS(4)
   ZEND=COORDS(6)
   ITCH = 0
   A2 = AINI
C   (LOST FLAG = 0 FOR ORIG' SEARCH , 1 FOR NEW SEARCH)
   LOST=0
   KFLAG=0
C--------------------------------------------------------------------
C FIRSTLY FOR CELL INCLUDING SOURCE
 55  XPO = XSTART
   ZPO = ZSTART
   IF(WRITE .EQ. 1) THEN
    IF(IWRYN.EQ.1) WRITE(12,166)NPT,XPO,ZPO
 166   FORMAT(I4,2F8.4)
    NPT = 1
   ENDIF
   ITRY=0
   ITRY2=0
   D1=0.0
   PHI1=0.0
   ANGLE=0.0
 120  CONTINUE
   ITRY=ITRY+1
   AIN=AINI
   IF(LOST.EQ.1)ITRY2=ITRY2+1
   CALL ZERO(ICELXZ,TK)
C   NOW SET UP CONDITION IF RAY NOT FOUND after
c   first method
C   TRY NEW RAY SEARCH BASED ON HALVING OF ANGLES
C   LOST = 1 MEANS NEW RAY SEARCH!  AINSAV IS THE
c   VALUE OF AINI ON THE PREVIOUS ITERATION
   IF(ITRY.EQ.NTRY.AND.LOST.EQ.0) THEN
c      PRINT*,'RAY LOST'
      TTOT=0.0
      TOTR=0.0
      AINI = A2
      LOST=1
      AINSAV = PI/2.
      ITRY2 = 1
      D1 = -999
   ENDIF
C   NOW SET UP CONDITION IF RAY STILL NOT FOUND after
c   second method
   IF(ITRY2.EQ.NTRY.AND.LOST.EQ.1)THEN
      TTOT=0.0
      TOTR=0.0
      CALL ZERO(ICELXZ,TK)
      NPT = 0
      LOST = 0
      ITCH = ITCH + 1
      IF(ITCH.EQ.1) THEN
       ISTRT = 0
       AINI = BAA1 -  0.334*SIGN(1.0,BAA1)
      ELSEIF(ITCH.EQ.2) THEN
C   USE A STRAIGHT RAY FOR THIS RAYPATH
       ISTRT = 1
       WRITE = 0
       AINI=A2
      ENDIF
      GOTO 55
   ENDIF
C ------------------------------------------------------------
c    IF(AINI.LT.0.0.AND.ZSTART.EQ.0.0) AINI=0.005
   IF(ABS(AINI).GE.PI2) THEN
127   FORMAT(/1X,'SHOOTING OUT OF FRAME - ABORT!'/1X,
  1    'AINI= ',E10.4,' IRAY= ',I5/)
c Where the hell did these lines come from?
    IF(AINI.GE.PI2)AINI=PI2-.4
    IF(AINI.LE.-PI2)AINI=-PI2+.4
   ENDIF
   XPO=XSTART
   ZPO=ZSTART
   IX=INT(XPO/BOXX+0.0001)+1
   IZ=INT(ZPO/BOXZ+0.0001)+1
   IF(ZPO-FLOAT(IZ-1)*BOXZ.LT.0.0001.AND.AINI.LT.
  1          -0.0005) IZ=IZ-1
   ICELL=(IZ-1)*ICELLX+IX
   DZ=ZSTART-(FLOAT(IZ-1))*BOXZ
   DX=FLOAT(IX)*BOXX-XSTART
   K=0
   IF(AINI.LT.0.0) K=1
   ALPHA=ATAN(DZ/DX)
   BETA=ATAN((BOXZ-DZ)/DX)
   IEXIT=1
c ray out bottom
   IF(AINI.LE.BETA) IEXIT=0
c ray out top
   IF(AINI.LT.-ALPHA) IEXIT=-1
   IF(IEXIT) 130,140,130
c ray out top (IEXIT=-1) or bottom (IEXIT=+1)
130  TOTR=ABS((DZ-BOXZ*((IEXIT+1)/2))/SIN(AINI))
   TTOT=TOTR*SL(ICELL)
c next cell (NCELL):- IZ+1 for out bottom/ IZ-1 for out top
   IZ=IZ+IEXIT
   NCELL=(IZ-1)*ICELLX+IX
c Out top of model
   IF(NCELL.LT.1) THEN
    AINI=0.0
    GOTO 120
   ENDIF
   VL=1.0/SL(ICELL)
   VELI=1.0/SL(NCELL)
c if loop to test for critical ray
   IF(VL.LT.VELI) THEN
    CRIT=ACOS(VL/VELI)
    IF(ABS(AINI).LE.ABS(CRIT).AND.ISTRT.EQ.0) THEN
     IF ((XPO+TOTR*COS(AINI)).LT.XEND) THEN
      GO TO 1800
     ELSE
      AIN = AINI
      GO TO 131
     ENDIF
    ENDIF
   ENDIF
c refract ray or straight ray
   IF(ISTRT.EQ.0) THEN
    AIN=ACOS(VELI*COS(AINI)/VL)*SIGN(1.0,AINI)
   ELSE
    AIN = AINI
   ENDIF
c new Z position
 131  IF (IEXIT.EQ.1) THEN
    ZPO=FLOAT(IZ-1)*BOXZ
   ELSE
    ZPO=FLOAT(IZ)*BOXZ
   ENDIF
   XPO=XPO+TOTR*COS(AINI)
   GOTO 150
c ray out side
```

```
140  TOTR=ABS(DX/COS(AINI))
   TTOT=TOTR*SL(ICELL)
   IX=IX+1
   NCELL=(IZ-1)*ICELLX+IX
   VL=1.0/SL(ICELL)
   VELI=1.0/SL(NCELL)
c if loop to test for critical ray
   IF(VL.LT.VELI) THEN
    CRIT=ASIN(VL/VELI)
    IF(ABS(AINI).GE.ABS(CRIT).AND.ISTRT.EQ.0) THEN
     IF ((FLOAT(IX-1)*BOXX).LT.XEND) THEN
      GO TO 1800
     ELSE
      AIN = AINI
      GO TO 141
     ENDIF
    ENDIF
   ENDIF
c refract ray or straight ray
   IF(ISTRT.EQ.0) THEN
    AIN=ASIN(VELI*SIN(AINI)/VL)
   ELSE
    AIN = AINI
   ENDIF
141  XPO=FLOAT(IX-1)*BOXX
   ZPO=ZPO+TOTR*SIN(AINI)
150  TK(ICELL)=TOTR
   ICELL=NCELL
   IF(ICELL.LE.0) THEN
     AINI=AINI+ASTEP
     GOTO 120
   ENDIF
C END OF SOURCE CELL CALCULATION
C CHECK TO SEE IF RAY ALREADY PAST X POSITION
   IF (XPO.GE.XEND) GOTO 1220
C NOW FOR MAIN LOOP - I.E. CELLS EXCLUDING SOURCE
c rays start and finish on a gridline
C ISN = SIGN OF AIN ie: +VE = RAY TRAVELLING DOWN
155  ISN=1
   K=0
   ICELL=(IZ-1)*ICELLX+IX
C RAY GOING TO LEFT TRY NEXT ANGLE INCREMENT
   IF(ABS(AIN).GE.PI2) THEN
    DIF=ZPO-ZEND+(XEND-XPO)*TAN(AIN)
    GOTO 1699
   ENDIF
C RAY GOING UP
   IF(AIN.LT.0.0) THEN
    ISN=-1
    K=1
   ENDIF
   VL=1.0/SL(ICELL)
   AIN1 = ABS(AIN)
C IF-ROUTINE FOR STRAIGHT THROUGH RAYS ie horizontal
   IF(AIN1.LE.0.001)THEN
    RLEN=BOXX
    TD=BOXX/VL
    XPO=XPO+RLEN
    IX=IX+1
    GOTO 1220
   ENDIF
   AAA = AIN
c big if for start points near a horizontal line or not (ends before 1220)
   IF (ABS(FLOAT(IZ - 1 + K)*BOXZ - ZPO) .LE. 0.001 ) THEN
C     DYN. SOURCE EITHER ON HOR. MESH LINE OR CORNER
C     RAY SEGMENT AS AN ARC OF A CIRCLE?
c     IF (ABS(AAA) .LE. ASD) THEN
c      NSUB = NSUB + 1
c      ISW = 0
c      CALL CIRCLE(XPO, ZPO, ISW, ISN, AIN, IX, IZ, SL, RLEN,
c    1 TD,ICELLX,ICELXZ,BOXX,BOXZ, GRAD, APR, AINI,NIT,1)
C if no velocity gradient in box, ISW returns 1
c      IF (ISW .EQ. 0) GO TO 1220
c      IF (ISW.EQ. 10) GO TO 1800
c     ENDIF
c     GRAD = 0.0
C* DIS IS DISTANCE FROM DYNAMIC SOURCE TO NEXT
C VERTICAL MESH LINE.  BX IS VERTICAL DISTANCE FROM
C DYN.SOURCE TO EXIT POINT ON VERTICAL MESHLINE.
     DIS = FLOAT(IX) * BOXX - XPO
     BX = DIS * TAN(AIN) * FLOAT(ISN)
     IF (BX .EQ. BOXZ) BX = BX + 0.0001
C*    RAY THROUGH HOR. SIDE OF BOX  ---------------
     IF (BX .GE. BOXZ) THEN
      RLEN = ABS(BOXZ/SIN(AIN))
      TD = RLEN / VL
      IF (IZ+ISN.GT.ICELLZ+1.OR.IZ+ISN.LT.1)THEN
       VELI=VL
       DIF=ZPO-ZEND+(XEND-XPO)*TAN(AIN)
       IF(DIF*ISN.GT.0.0) THEN
        IF ((XPO+SQRT((RLEN*RLEN)-(BOXZ*BOXZ))).
    1 LT.XEND) THEN
         GOTO 1800
        ELSE
         VELI = VL
         AIN = AAA
         GOTO 157
        ENDIF
       ENDIF
       KFLAG=1
       GOTO 1699
      ENDIF
      VELI = 1.0/SL((IZ+ISN-1)*ICELLX +IX)
c if loop to test for critical ray
 156   IF(VL.LT.VELI) THEN
       CRIT=ACOS(VL/VELI)
       IF(ABS(AAA).LE.ABS(CRIT)) THEN
        DI = -DIF
        IF (ISTRT.EQ.0) THEN
         IF ((XPO+SQRT((RLEN*RLEN)-
  1    (BOXZ*BOXZ))).LT.XEND) THEN
          GOTO 1800
         ELSE
          AIN = AAA
          GOTO 157
         ENDIF
        ENDIF
       ENDIF
      ENDIF
c refract ray or straight ray
      IF(ISTRT.EQ.0) THEN
       AIN=ACOS(VELI*COS(AAA)/VL)*FLOAT(ISN)
      ELSE
       AIN = AAA
      ENDIF
 157   ZPO = FLOAT(IZ - K) * BOXZ
      IZ = IZ + ISN
      XPO = XPO + (SQRT((RLEN*RLEN) - (BOXZ*BOXZ)))
     ELSE
C*     RAY THROUGH VERT. SIDE OF BOX --------------
      RLEN = ABS(DIS/COS(AIN))
      TD = RLEN / VL
      IF(IX+1.GT.ICELLX)THEN
       VELI=VL
      ELSE
       VELI = 1.0/SL((IZ-1)*ICELLX+IX+1)
      ENDIF
c if loop to test for critical ray
      IF(VL.LT.VELI) THEN
       CRIT=ASIN(VL/VELI)
       IF(ABS(AAA).GE.ABS(CRIT)) THEN
        DI = -DIF
        IF (ISTRT.EQ.0) THEN
         IF ((FLOAT(IX)*BOXX).LT.XEND) THEN
```

```
          GOTO 1800
         ELSE
          AIN = AAA
          GOTO 1208
         ENDIF
        ENDIF
       ENDIF
      ENDIF
      IF(ISTRT.EQ.0) THEN
       AIN=ASIN(VELI*SIN(AAA)/VL)
      ELSE
       AIN = AINI
      ENDIF
1208    XPO = FLOAT(IX) * BOXX
      IX = IX + 1
      ZPO = ZPO + (BX*FLOAT(ISN))
     ENDIF
C end of loop for ray starting on horiz line
    ELSE
C**    DYN. SOURCE ON VERTICAL MESH LINE
C      RAY SEGMENT AS AN ARC OF A CIRCLE ?
c      IF (ABS(AAA) .LE. ASD) THEN
c       NSUB = NSUB + 1
c       ISW = 0
c       CALL CIRCLE(XPO, ZPO, ISW, ISN, AIN, IX, IZ, SL, RLEN,
c     1 TD, ICELLX,ICELXZ,BOXX,BOXZ, GRAD, APR, AINI,NIT,1)
C if no velocity gradient in box, ISW returns 1
c       IF (ISW .EQ. 0) GO TO 1220
c       IF (ISW.EQ. 10) GO TO 1800
c      ENDIF
      GRAD = 0.0
C  DEP IS DEPTH THAT RAY SOURCE IS FROM NEXT
c  HORIZONTAL MESH LINE.  BX IS HORIZONTAL DISTANCE
c  TRAVELLED BY RAY TO NEXT HOR. MESH LINE.
      DEP = FLOAT(IZ) * BOXZ - ZPO
      IF (AIN .LT. 0.0) DEP = BOXZ - DEP
      BX = DEP / TAN(AIN)
      IF (ABS(BX) .EQ. BOXX) BX = ABS(BX + 0.0001)
      IF (ABS(BX) .GE. BOXX) THEN
C*     RAY THROUGH VERT. SIDE OF BOX ---------
       RLEN = ABS(BOXX/COS(AIN))
       TD = RLEN / VL
       IF(IX+1.GT.ICELLX)THEN
        VELI=VL
       ELSE
        VELI = 1.0/SL((IZ-1)*ICELLX+IX+1)
       ENDIF
c if loop to test for critical ray
       IF(VL.LT.VELI) THEN
        CRIT=ASIN(VL/VELI)
        IF(ABS(AAA).GE.ABS(CRIT)) THEN
         DI = -DIF
         IF (ISTRT.EQ.0) THEN
          IF ((FLOAT(IX)*BOXX).LT.XEND) THEN
           GOTO 1800
          ELSE
           AIN = AAA
           GOTO 1209
          ENDIF
         ENDIF
        ENDIF
       ENDIF
       IF(ISTRT.EQ.0) THEN
        AIN=ASIN(VELI*SIN(AAA)/VL)
       ELSE
        AIN = AINI
       ENDIF
1209    XPO = FLOAT(IX) * BOXX
       IX = IX + 1
       ZPO = ZPO + ((SQRT((RLEN*RLEN) -
     1 (BOXX*BOXX)))*FLOAT(ISN))
      ELSE
C*     RAY THROUGH HOR. SIDE OF BOX --------
       RLEN = ABS(DEP/SIN(AIN))
       TD = RLEN / VL
       IF(IZ+ISN.GT.ICELLZ.OR.IZ+ISN.LT.1)THEN
        VELI=VL
        DIF=ZPO-ZEND+(XEND-XPO)*TAN(AIN)
        IF(DIF*ISN.GT.0.0) THEN
         IF ((XPO+ABS(BX)).LT.XEND) THEN
          GOTO 1800
         ELSE
          VELI = VL
          GOTO 1210
         ENDIF
        ENDIF
        KFLAG=1
        GOTO 1699
       ENDIF
       VELI = 1.0/SL((IZ+ISN-1)*ICELLX+IX)
c if loop to test for critical ray
1210    IF(VL.LT.VELI) THEN
        CRIT=ACOS(VL/VELI)
        IF(ABS(AAA).LE.ABS(CRIT)) THEN
         DI = -DIF
         IF (ISTRT.EQ.0) THEN
          IF ((XPO+ABS(BX)).LT.XEND) THEN
           GOTO 1800
          ELSE
           AIN = AAA
           GOTO 1211
          ENDIF
         ENDIF
        ENDIF
       ENDIF
c refract ray or straight ray
       IF(ISTRT.EQ.0) THEN
        AIN=ACOS(VELI*COS(AAA)/VL)*FLOAT(ISN)
       ELSE
        AIN = AAA
       ENDIF
1211    ZPO = FLOAT(IZ - K) * BOXZ
       IZ = IZ + ISN
       XPO = XPO + ABS(BX)
      ENDIF
C end of the loop for ray starting on vert line
     ENDIF
C end of the loop for ray starting on horizontal line or not
C*************************************************
C* CHECK WHERE RAY ENDS
1220 IF(XPO.LT.XEND) THEN
C  ray not entering same column as receiver position
      TTOT=TTOT+TD
      TOTR=TOTR+RLEN
      TK(ICELL)=RLEN
C  RAY EXITING THROUGH SURFACE
      IF(ZPO.LE.0.0.AND.ISN.EQ.-1) THEN
       DIF=ZPO-ZEND+(XEND-XPO)*TAN(AIN)
       GOTO 1800
      ELSE IF (ZPO .GE. ZMAX+BOXZ) THEN
C  RAY LEAVING VELOCITY FIELD THRO' BOTTOM
       DIF=ZPO-ZEND+(XEND-XPO)*TAN(AIN)
       GOTO 1800
      ELSE
c  not last segment, so go back to 155 for next ray
       IF(WRITE.EQ.1)THEN
        IF(IWRYN.EQ.1) WRITE(12,186)NPT,XPO,ZPO
186       FORMAT(I4,2F8.4)
        NPT = NPT + 1
       ENDIF
       GO TO 155
      ENDIF
     ENDIF
c  ray is past image point, so retrace last segment to same x as im pt
```

```
      XPO=XPO-RLEN*COS(AIN)
      ZPO=ZPO-RLEN*SIN(AIN)
      RLEN=ABS((XEND-XPO)/COS(AIN))
      TD=RLEN/VL
      TOTR=TOTR+RLEN
      TTOT=TTOT+TD
      TK(ICELL)=RLEN
      ZPO=(XEND-XPO)*TAN(AIN)+ZPO
      DIF=ZPO-ZEND
      IF(WRITE.EQ.1)THEN
       IF(IWRYN.EQ.1) WRITE(12,186)NPT,XEND,ZPO
       NPT = NPT + 1
      ENDIF
c KFLAG = 1 - ray out of horizontal side of box CHECK THIS
1699  IF(KFLAG.EQ.1.AND.IZ.LE.1)THEN
       RLEN=ABS((ZEND-ZPO)/SIN(AIN))
       TK(ICELL)=RLEN
       TD=RLEN/VL
       TOTR=TOTR+RLEN
       TTOT=TTOT+TD
      ENDIF
C   NOW CHECK IF RAY HAS BEEN FOUND.
C     if ABS(DIF) < ABS(ACC)
c     it looks as though DIF only measures the Z difference
1800  IF(ABS(DIF).GT.ABS(ACC))THEN
C  DO THE RAY SEARCH TO UPDATE THE "TAKE OFF" ANGLE
       ANGLE=AINI
       IF(LOST.NE.1) THEN
C       ORIGINAL ITERATIVE SEARCH ...WHEN LOST = 0
        IF(D1.EQ.0.0) THEN
         AINI= AINI-ASTEP1*SIGN(1.0,DIF)
        ELSEIF(DIF.EQ.D1) THEN
         IF(AINI.EQ.ANGLE) AINI=ANGLE+ASTEP1
        ELSEIF(SIGN(1.0,DIF).EQ.SIGN(1.0,D1)) THEN
         AINI= AINI-ASTEP1*SIGN(1.0,DIF)
        ELSE
         AINI= AINI-(DIF/(DIF-D1))*(AINI-PHI1)
         IF(AINI.EQ.ANGLE) AINI=ANGLE+ASTEP1
        ENDIF
        D1=DIF
        PHI1=ANGLE
        GOTO 120
       ELSE
C       PETER'S NEW SEARCH METHOD 20/04/93
        IF(DIF.GT.0.0)THEN
         THETAH = AINI
        ELSE
         THETAB = AINI
        ENDIF
        AAAI = (THETAH+THETAB)/2
        D1=DIF
        AINSAV = AINI
        AINI = AAAI
c        WRITE = 1
        GOTO 120
       ENDIF
      ELSE
C      RAY FOUND!!!!!   RESET LOST FLAG
       LOST=0
       AOUT1=AINI
       AOUT2=AIN
       IF(WRITE .EQ. 0)THEN
        WRITE = 1
        GO TO 55
       ENDIF
       DO 2005,I=1,ICELXZ
        IF(TK(I).NE.0.0) NORAYS(I)=NORAYS(I)+1
2005    CONTINUE
       XDUM = 999.
       ZDUM = 999.
       IF(IWRYN.EQ.1) WRITE(12,2003)NPT,XDUM,ZDUM
2003   FORMAT(I4,2F8.4)
      ENDIF
      RETURN
      END
```


UNIV DUNELM