

070
Row
Kir
MS

Kirchhoff pre-stack depth migration
AC .H3 no. RM87 15491

APR 13 1987



Rowe, Mary M.
SOEST Library

HAWAII INSTITUTE OF GEOPHYSICS
LIBRARY

KIRCHHOFF PRE-STACK DEPTH MIGRATION

A Thesis Submitted to the Graduate Division of the
University of Hawaii in Partial Fulfillment
of the Requirements for the Degree of

Master of Science

in Geology and Geophysics

May 1987

By

Mary Mannion Rowe

Thesis Committee

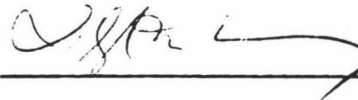
Brian Taylor, Chairman
L. Neil Frazer
Donald M. Hussong
Ralph Moberly

We certify that we have read this thesis and that, in our opinion, it is satisfactory in scope and quality as a thesis for the degree of Master of Science in Geology and Geophysics.

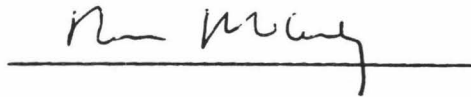
THESIS COMMITTEE

Brian Taylor

Chairman







ABSTRACT

This thesis describes a computer algorithm developed to perform Kirchhoff pre-stack depth migration of seismic multichannel data using travel times obtained by tracing rays. Before migration, a table of travel times $T(x_n, x, z)$ from each depth point (x, z) to each surface point $(x_n, 0)$ is constructed. This table is computed by ray tracing from an array of subsurface points through a velocity model consisting of constant-velocity polygons. Next the double Kirchhoff integral over shots and receivers is evaluated. While migrating, a "coherency depth section" can be computed which provides an objective measure of the goodness of the assumed velocity function. Sampling theory prescribes the shot and receiver apertures for the double Kirchhoff integral as functions of shot spacing and receiver spacing, respectively. These apertures determine a theoretical upper limit on the horizontal resolution of Kirchhoff depth migration.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF FIGURES	v
I. INTRODUCTION	1
II. PURPOSE OF MIGRATION	2
III. MIGRATION TECHNIQUES	8
Finite-difference Migration	8
Frequency Domain Migration	11
Reverse Time Migration	12
Kirchhoff Summation Migration	13
Pre-stack Migration	17
Depth Migration	18
Velocity Analysis	22
IV. RAY TRACING	25
Ray Tracing Through the Test Model	35
V. THE SYNTHETIC DATA SET	37
VI. MIGRATION	40
Migration of the Test Model	45
VII. THE COHERENCY SECTION	49
VIII. SUMMARY	52
APPENDIX A: PROGRAM LISTINGS	
Ray Tracing	54
Migration	76
APPENDIX B: INCREASING RAY TRACING EFFICIENCY	89
LITERATURE CITED	94

LIST OF FIGURES

Figure		Page
1	Hyperbolic diffraction curve produced by signal scattered from a subsurface point	3
2	Iso-travel time curves associated with shot - receiver pairs (x_s, x_{r1}) and (x_s, x_{r2})	5
3	Envelope of iso-travel time curves tangent to a given reflector .	7
4	Common depth point gather including signals reflected from points other than the common depth point	19
5	Image ray, emerging perpendicular to the surface, offset from the scattering point	21
6	Test model and ray trace for a single depth point	27
7	Wavelet sampled adequately, avoiding aliasing, showing stretching with depth	30
8	Maximum angle θ for which wavelet impinging on the surface will not be aliased at trace separation Δx	31
9	Maximum shot - receiver separation (D) without aliasing, dependent on incidence angle, θ , of wavefront at the surface . .	32
10	Horizontal resolution d as a function of aperture D_s (D_r) and wavelet period T_{min}	34
11	Test model used to construct synthetic data set, showing a single shot reflected from the third interface	38
12	First 40 shots of synthetic data	39

13	(a) Single common shot gather	42
	(b) Migrated trace formed by summing over all the shot gathers .	42
14	Construction of depth migrated trace by spreading each data sample over iso-travel time curve $t(x_s, x_r, z, x)$	43
15	Depth section obtained by migrating synthetic data set	47
16	(a) Region of velocity model affecting coherence value $C(x', z')$.	50
	(b) Region of low coherence caused by bad velocity estimate at (x'', z'')	50
17	Rays traced downward from shot S and through the grid of depth points (x', z')	90
18	Tracing rays from point (x_n, z_n) to points $(x_m', z_m' - \Delta z)$ to de- termine shortest travel time path from the surface to (x_n, z_n) . .	92

I. INTRODUCTION

Kirchhoff migration methods use the integral solution to the scalar wave equation to sum scattered energy back to its point of origin (French, 1974, 1975; Gardner, et al., 1974; Schneider, 1978). Kirchhoff migration assumes that each point in the subsurface acts as an independent point scatterer and that any reflector may be accurately represented as a distribution of point scatterers (Gardner, et al., 1974). Many Kirchhoff migration methods are designed to operate on stacked seismic data (for example, French, 1975; Schneider, 1978). Normal moveout (NMO) correction and common depth point (CDP) stacking significantly reduce the amount of computer time required for migration and simplify the migration process but they also degrade the data (Kuhn and Alhilali, 1977; Schultz and Sherwood, 1980; May and Covey, 1983; Hosken and Deregowski, 1985). It has been known for many years that imaging problems caused by stacking and NMO correction can be avoided only by pre-stack migration, not by any post-stack processing. Applied to unstacked data, Kirchhoff summation migration has some advantages over other types of migration. The Kirchhoff integral is robust in the presence of velocity variations, the waveform is preserved, and migration is not limited to regions of shallow dip (Schneider, 1978).

II. PURPOSE OF MIGRATION

Multichannel seismic sections from structurally complex regions are generally difficult to interpret. Structural features are masked by high amplitude hyperbolic curves caused by signals diffracted from scattering points or surfaces within the subsurface. Migration collects this scattered energy and restores it to its source, collapsing the diffraction curves and enhancing the images of the reflectors in the earth. Mufti (1985) gives an excellent description of migration in general and its limitations.

Energy scattered from a point P in a homogeneous medium produces a hyperbolic curve in a zero-offset seismic section, the apex of the hyperbola lying directly on the point scatterer (Figure 1). The image ray, the ray arriving perpendicular to the surface, is recorded at the shot-receiver pair directly above the scatterer. Receivers to either side of the point scatterer receive diffracted energy. Since the energy scattered to either side travelled farther in order to reach the surface, these signals are recorded at later times than the specular reflection, creating the hyperbola observed on the seismic section (Figure 1).

The Kirchhoff-Huygens diffraction principle states that a reflecting surface may be treated as a series of closely spaced point scatterers, each of which scatters energy equally in all directions. Assuming the distance between such point scatterers is infinitely small, the limbs of the diffraction curves from each point scatterer cancel out, leaving only the signal at the apex of the hyperbola (Gardner, et al., 1974, Figure 2). Thus a zero-offset reflection section from a flat reflection surface correctly images the flat reflector except at the edges where the limbs of the hyperbolae do

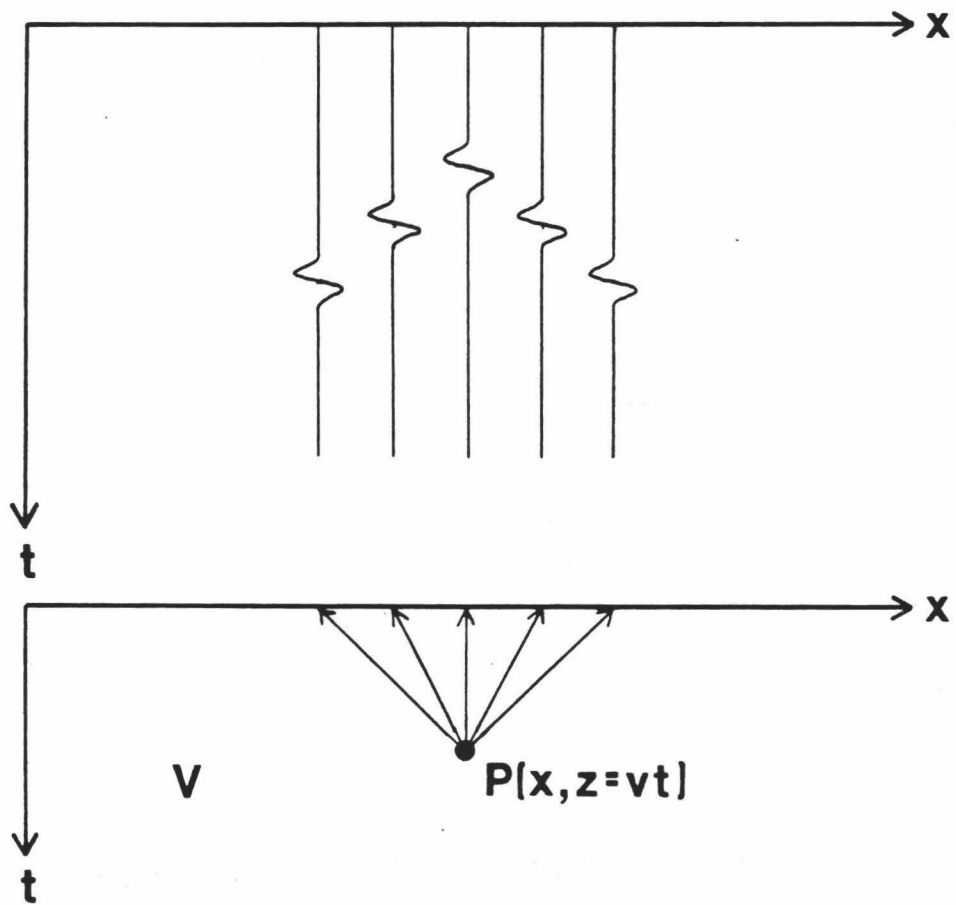


Figure 1. Hyperbolic diffraction curve produced by signal scattered from a subsurface point.

not quite cancel. Reflection sections acquired over regions where the subsurface is made up of flat layers do not need to be migrated.

Where the structure is more complex, collecting this diffracted energy by migration is required to clarify the section. In most cases, the velocity structure will cause diffraction curves to deviate from a hyperbola and will shift the apex off the actual location of the point scatterer. Dipping surfaces may appear to be imaged correctly by a zero offset section. However, as shown by Mufti (1985), the position and dip of the reflector have in fact been distorted. Though the limbs of the diffraction curves cancel, the apexes are not correctly located along the reflector (Mufti, 1985, Figure 3). The shortest distance between a surface point and a reflector is along the path perpendicular to the reflector. In the case of a dipping reflector the first arrival will be recorded at a surface point offset from the point lying directly above the scattering point on the reflector. The image of the reflector will be shifted laterally and the dip of the reflector will be decreased (Mufti, 1985). If the reflectors are curved and discontinuous few diffraction curves will destructively interfere. Instead, they will mask the specular reflections and create a confusing seismic section. Migration must be performed on the section to image the dipping reflectors correctly and to eliminate the diffraction curves.

Each scattered signal pulse from a source at x received at x arrives at a time t . Since, ideally, the shot radiates energy equally in all directions and the receiver records energy arriving from all directions, the amplitude of the signal received at time t is the sum of the signals lying along the iso-travel time curve for time t (Figure 2). Each iso-travel time curve is made up of all points (x,z) for which the time it takes to travel from the shot to

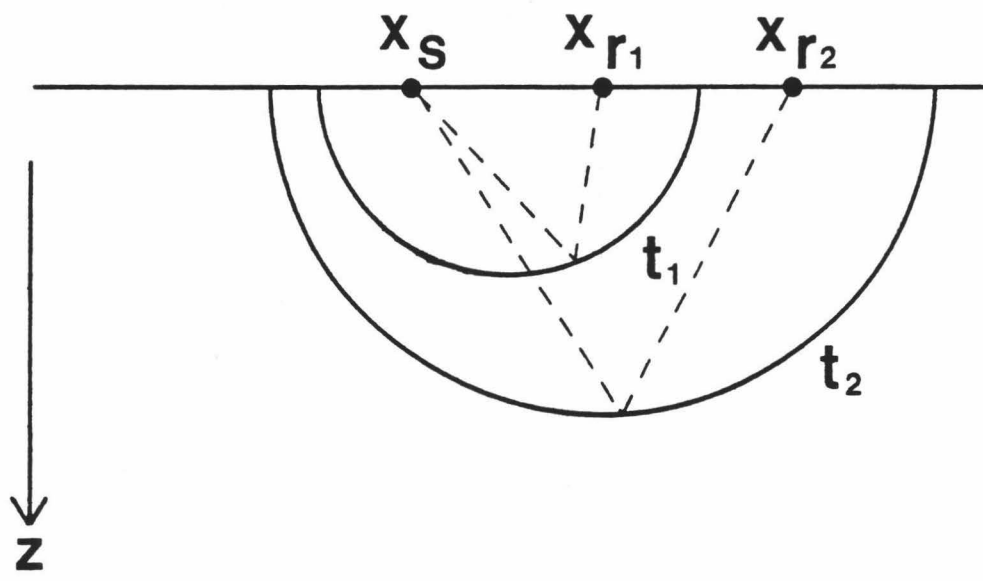


Figure 2. Iso-travel time curves associated with shot - receiver pairs (x_s, x_{r1}) and (x_s, x_{r2}) .

point (x,z) and up to the receiver is equal to time t . Any components of the signal received at that time may have originated at any point (x,z) along the curve for t . Ideally, migration would be able to return each component of the received signal to the exact point from which it originated. Geophone and hydrophone arrays are generally not directional and a single time sample is not made up of discrete components, therefore the exact source of each contribution to the time sample cannot be pinpointed. Migration spreads the pulse received at time t over all points lying on the iso-travel time curve (Figure 2). The envelope of the iso-travel time curves tangent to a reflector determines the position of the reflector. Correct locations of strong scatterers stand out against background noise since a strong scatterer will lie along many travel time curves for which the received amplitude was large. Points not lying on scattering surfaces will lie on a few travel time curves associated with high amplitude signals, but most of the iso-travel time curves passing through these point will be associated with low amplitude signals (Figure 3). In order to minimize background noise and enhance the reflections migration does not include the entire iso-travel time curve. The window over which migration is performed must be carefully chosen and weighted, as discussed below.

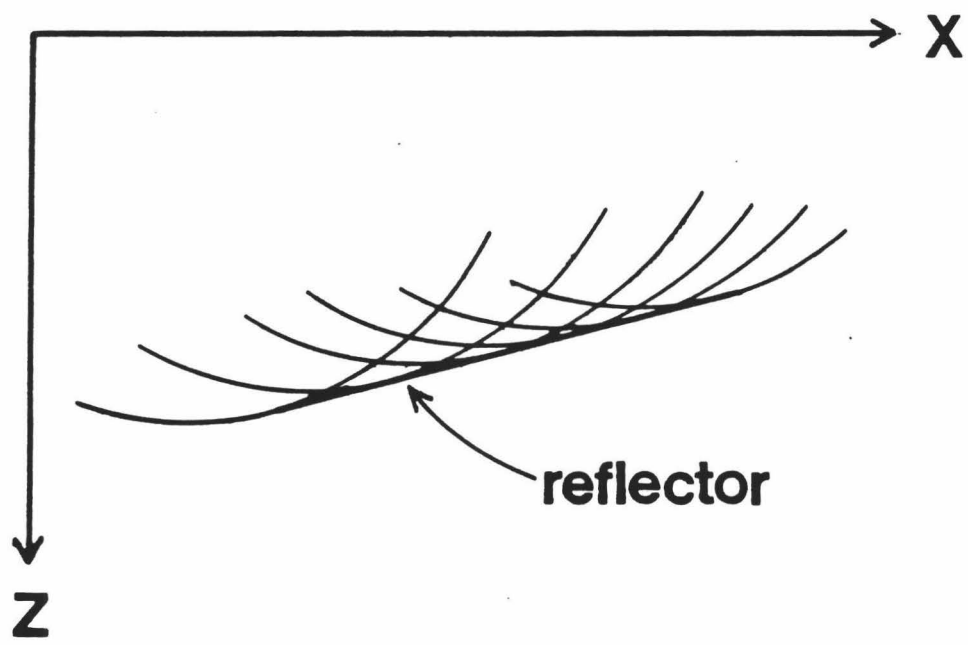


Figure 3. Envelope of iso-travel time curves tangent to a given reflector.

III. MIGRATION TECHNIQUES

Migration sums energy diffracted from subsurface scatterers back to the points from which it was scattered. This process is equivalent to evaluating the wave field at the time it was scattered in the subsurface (Mufti, 1985). To simplify the problem of solving the wave equation to obtain the wavefield at time $T=0$ it is commonly assumed that the earth behaves like a fluid, that density changes may be ignored, that the earth is two dimensional (in the case of two dimensional data), and that multiply reflected energy may be ignored. In general, migration by summation along diffraction curves gives some improvement even when the curves are assumed to be hyperbolic but it must be noted that most wave equation migration schemes do not take into account small phase shifts resulting from diffraction and that significant migration noise may be generated (Mufti, 1985).

Commonly used migration techniques are finite-difference migration, frequency domain (F-K) migration, reverse time migration, and Kirchhoff migration. These methods are based on different methods for evaluating the acoustic wave equation and are generally referred to as wave equation migration. These methods ignore S-waves and P- to S-wave conversions.

Finite-difference Migration

Finite-difference migration is derived by expressing the scalar wave equation as a difference equation. Given the wavefield recorded at depth $z=0$ over some interval of time the wavefield at earlier times can be found by extrapolating backwards. Each downward continuation step produces a migrated

section above the current datum and a partially migrated section below. Finite-difference migration applied to zero-offset data is based on Claerbout and Doherty's (1972) "exploding reflector" model. The zero-offset wavefield at the surface is assumed to be due to signals originating on subsurface reflectors at time $t = 0$. While the exploding reflector model is good for calculating travel times its usefulness is limited to cases in which there are no caustics, no focal points, and no absorption or dispersion (Berkhout, 1984). The theory is also limited to normal moveout (NMO) corrected, stacked data (zero-offset data). In addition, geometrical spreading is not properly accounted for (Berkhout, 1984) and only the minimum travel time arrival from each scatterer is modeled (Lowenthal, et al., 1976; Deregowski and Brown, 1983). Travel times for finite-difference depth migration were obtained by Reshef and Kosloff (1986) by solving the eikonal equation at each depth interval as the solution steps downward through the section. Velocities for each step are determined by trial and error; the best velocity is that which produces the greatest continuity in the reflectors.

Derivation of the difference equation ignores the reflected field and restricts the solution to wavefronts travelling less than 30 degrees from the vertical (Claerbout and Doherty, 1972). A further limitation of the accuracy of the migrated section, true of all migration schemes, is that the velocity must be fairly well known in order to extrapolate the wavefield correctly and focus the diffracted signal correctly. Claerbout and Doherty (1972) were able to get good results using stacking velocities for migration. Unfortunately, in typical cases the velocity structure is poorly known prior to migration. Claerbout and Doherty (1972) also assumed that the velocity function was slowly varying vertically and horizontally. Claerbout (1970), however, showed

that finite-difference methods, unlike ray theory, accurately handle diffractions, enabling dipping and curved reflectors to be handled correctly. DeVries and Berkhout (1984) believe migration by finite-difference downward continuation is less sensitive to velocity error than other migration techniques.

Judson, et al., (1980) and Schultz and Sherwood (1980) extended the principles of finite-difference migration to depth migration by adding a time shifting term to the migration equation. This operator is a velocity dependent term which transforms the time-migrated section at each downward continuation step into a depth section using the local velocity. Incorporation of this time shifting term produces a much improved migrated section but is time consuming (Judson, et al., 1980). Use of this time shifting term also increases the sensitivity of the migrated section to the velocity model (Judson, et al., 1980). Finite-difference migration enables the user to specify new parameters at each downward continuation step, making it particularly well suited to migration in the presence of vertical and lateral velocity inhomogeneities (Judson, et al., 1980; Schultz and Sherwood, 1980).

Hatton, et al., (1981), have developed an alternate depth migration scheme using a version of the finite-difference equation which retains the second order z differential term. Velocity is no longer assumed to be laterally slowly varying. Energy deflected over large horizontal distances can be accurately repositioned. According to Hatton, et al., (1981) neglecting this second order z term will only give satisfactory results for a migrated time section. The corresponding migrated depth section would be inaccurate. During development of this algorithm, Hatton, et al., (1981)

found that the migrated section is more sensitive to the shape of the reflectors than to the values of the velocities.

Frequency Domain Migration

Stolt (1978) and Gazdag (1978) simplified the downward continuation operator for migration by working in the frequency-wavenumber domain. Both their methods result in a more accurate migrated section, especially where features dip steeply, and require significantly less computation time than finite difference migration. In the development of his migration technique Gazdag (1978) uses the second order approximation to the two dimensional scalar wave equation as expressed by Claerbout and Doherty (1972). The downward extrapolation operator is derived from a double Fourier transformed asymptotic form of the solution to the wave equation. Extrapolation is achieved as a phase shift of the data in the frequency-wavenumber domain (Gazdag, 1978).

To apply Stolt's (1978) technique a double Fourier transform is applied to the wave equation, transforming it into frequency-wavenumber space. In this domain, migration of a diffracted signal is equivalent to vertical mapping. Downward continuation is achieved by means of a frequency shift accompanied by a scale change (Stolt, 1978). Events of equal dip are migrated simultaneously. Steep dips (up to 45 degrees) and laterally varying structure may be accurately migrated (Stolt, 1978; Chun and Jacewitz, 1981). Like finite-difference downward continuation methods, this technique does not account for P- to S-wave conversions or transmission loss (Berkhout, 1984). Migration parameters must be chosen such that aliasing does not occur at the

higher frequencies present in the data. Frequencies above the aliasing frequency will be undermigrated (Stolt, 1978; Gazdag, 1978). An advantage of frequency-wavenumber domain migration noted by Berkhout (1984) is that each frequency may be weighted differently to improve the quality of the migrated section. To convert from a time section to a depth section requires a separate step.

Reverse Time Migration

Baysal, et al (1983) developed a migration method using the complete acoustic wave equation. This technique is based on the forward models developed by Gazdag (1981) and Kosloff and Baysal (1982, 1983) which model the velocity or pressure field using solutions to the complete two-dimensional acoustic wave equation. This solution is obtained by Fourier transforming the two dimensional acoustic wave equation with respect to time, converting it to a difference equation and solving it as a boundary value problem. Spatial derivatives are retained enabling structures having dips up to 90 degrees and containing strong horizontal velocity contrasts to be accurately migrated.

Assuming traction and displacement across all interfaces is continuous, the boundary conditions are provided by the seismic data set (the wavefield at the surface) and its time (Gazdag, 1981) or spatial derivatives (Kosloff and Baysal, 1983). Since the solution assumes the exploding reflector model is valid (only upward travelling waves are considered) the migration method derived by inverting this model can only be applied to stacked data (Kosloff and Baysal, 1983; Baysal, et al, 1983). Like the frequency domain migration techniques described above, this method avoids problems caused by truncating

summation over the diffraction curve (Gazdag, 1981) and enables each frequency to be migrated separately (Kosloff and Baysal, 1983). As in finite-difference migration, different migration parameters may be specified at each downward continuation step. In addition, evanescent waves can be specifically damped by eliminating all solutions for which the horizontal wavenumber is greater than $w/c(z)$ where w is frequency and $c(z)$ is velocity at depth z (Kosloff and Baysal, 1983). Error is introduced into the migration equation due to approximation of the second order spatial derivative terms. Since solutions obtained by the reverse time migration algorithm are non-unique, the downgoing waves must be specifically eliminated by discounting all solutions having a negative vertical wavenumber (Kosloff and Baysal, 1983).

Kirchhoff Summation Migration

Kirchhoff migration uses the Kirchhoff integral solution to the wave equation as the downward continuation operator (Schneider, 1978). Using the wavefield recorded at the surface, the scattered signal is summed along the diffraction curve and restored to the scattering point (French, 1974; Schneider, 1978). Implicit in Kirchhoff's solution is the Kirchhoff-Huygens diffraction principle; each point in the subsurface acts as an independent point scatterer. A reflecting interface is a continuum of these point scatterers (Hilterman, 1975; Waters, 1981). This principle also assumes that the incident pulse is scattered equally in all directions and that transmission losses are negligible (French, 1974, 1975). Such losses may be significant at interfaces having high velocity contrast, such as the water-sediment interface (Berkhout, 1980).

The Kirchhoff integral is most accurately applied to acoustic wavefields in a solid, neglecting shear waves. Hilterman (1975) showed that the Kirchhoff integral can be applied to forward modeling of wavefields in an elastic medium with good results. Kuo and Dai (1984) have developed a method of migration using a Kirchhoff-Helmholtz type integral solution to the complete elastic wave equation. Migration of P- and S-waves occurs simultaneously. Noise is reduced since shear wave arrivals are treated as signal rather than noise. Application of this type of migration assumes that shear wave information is available in the data. Commonly seismic surveys are not designed to record shear wave arrivals.

The advantages of Kirchhoff summation migration over finite difference migration include the use of a more exact solution to the scalar wave equation, the ability to weight the migration aperture to reduce noise, and the ability to migrate dips up to and including 90 degrees (Gardner, et al., 1974; Schneider, 1978). Berkhout (1980) shows that the versatility of the Kirchhoff migration technique is improved if it is applied recursively. Another attractive feature of Kirchhoff migration is that it preserves the waveform of the data (Schneider, 1978).

French (1974, 1975) and Schneider (1978) discuss simple, rapid Kirchhoff migration algorithms limited to velocity structure which may vary greatly in the vertical direction but only weakly in the horizontal direction. Overburden is assumed to be homogeneous, implying that diffraction curves will be hyperbolic, centered on the scattering point (Figure 1). Root mean square velocity is adequate for migration velocity. Lateral velocity changes are averaged over the migration aperture. Under these assumptions the uppermost reflector will be migrated adequately, but deeper reflectors will not have the

correct shape (Hubral, 1977). French (1974) showed that this method can easily be extended to three dimensions, an advantage since geologic structures are rarely truly two-dimensional. Reflections from off-line structures (side-swipe) are often focussed rather than suppressed by two-dimensional migration techniques (French, 1974).

Schneider (1978) and Schultz and Sherwood (1980) noted that the Kirchhoff integral, without the simplifying assumptions, such as those made by French (1974) and Schneider (1978), does not break down in the presence of lateral velocity changes. Several authors have taken advantage of this flexibility to develop Kirchhoff type migration techniques which take into account deviation of the diffraction curve from hyperbolic when lateral inhomogeneities are present. Carter and Frazer (1983) accommodated horizontal velocity variations by assuming the velocity is laterally homogeneous and then calculating perturbations caused by the lateral changes. This method retains much of the speed of the simple Kirchhoff integral migration technique but is limited to small, localized velocity variations. Kuhn and Alhilali (1977) noted that lateral velocity changes can be taken into account by dividing the migration aperture into many smaller segments, each with a different set of migration parameters. Hubral (1977) uses image rays (those emerging perpendicular to the surface) to correctly migrate horizontally varying structure. Deregowski and Brown (1983) recommend tracing rays through the velocity structure to obtain correct travel times, accounting for horizontally varying structure, for migration. Ray tracing will accurately predict the shape of the diffraction curve, limited in accuracy only by the velocity model through which the rays are traced and the computer time available. This method is generally considered too time consuming and expensive to implement. As noted

above, Kirchhoff migration as applied by French (1974, 1975), Gardner, et al (1974), and Schneider (1978) to stacked multichannel data is a rapid process. Unfortunately, common depth point (CDP) stacking and NMO correction degrade the data and decrease the resolution of the resulting migrated section (Kuhn and Alhilali, 1977; Schultz and Sherwood, 1980; Hosken and Deregowski, 1985).

Although in theory Kirchhoff migration is exact for an acoustic wavefield its lateral resolution is limited by seismic frequency, migration aperture, weighting factors, accuracy of the velocity model, and the time and spatial sampling intervals (Safar, 1985). Approximating the Kirchhoff integral as a discrete sum produces migration noise which becomes worse as frequency and migration aperture increase and velocity, travel time, and sampling interval decrease (Schneider, 1978). According to Gardner, et al., (1974) migrating signals from outside a small segment bracketing the scattering point will contribute more noise than signal to the migrated section. Signal level falls off rapidly as wavefronts propagate. Truncation of the summation will contribute additional noise to the migrated section. Because of these factors which decrease the signal to noise ratio, contributions to the sum must be weighted according to the distance travelled (Gardner, et al., 1974). Kuhn and Alhilali (1977) successfully applied weighting to the migration aperture in cases in which the data are aliased, producing a more accurately imaged section. Though weighting is necessary to improve the quality of the migrated section it also tends to reduce the resolution of steeply dipping events since signals diffracted from steeply dipping structures may travel large horizontal distances before reaching the surface (Gardner, et al., 1974).

Pre-stack Migration

Migration may be performed before the data are stacked or after NMO corrections and stacking have been applied to the data. Processing time required to migrate stacked data is less than that required to migrate unstacked data by a factor equal to the fold of the data. Claerbout's exploding reflector model only applies to stacked data. Pre-stack migration treats downward and upward travelling signals separately, post-stack migration assumes the same path is travelled by the signal in both directions. Except in the case of flat lying, homogeneous structure, the paths travelled by upward and downward waves may be significantly different. Post-stack migration will fail to collapse this diffracted signal completely or will incorrectly locate the source of the signal, creating a misleading migrated section (Schultz and Sherwood, 1980; Judson, et al., 1980; Reshef and Kosloff, 1986). Where velocity varies laterally or reflectors dip steeply the source of a diffraction will not lie at the apex of the diffraction curve and the curve will not be hyperbolic, assumptions inherent in post-stack migration techniques.

Berkhout (1984) favors pre-stack migration because it produces a true CDP stack as migration is performed, unlike conventional stacking which uses root mean square (rms) velocities to sum the traces and correct moveout. If the structure is complex then summing traces using averaged velocities will distort the data. Hatton, et al., (1981) reject CDP stacking as a step in processing seismic data due to the resulting loss of information from complex structural features which cannot be restored by migration. Others such as Kuhn and Alhilali (1977) and Schultz and Sherwood (1980), have reported that

NMO corrections and stacking smear the diffracted signals, decreasing horizontal and vertical resolution. Figure 4 illustrates how traces within a single CDP gather may actually contain signals scattered from many different portions of a reflector. Events from steeply dipping interfaces are discriminated against by stacking. Such events have significantly lower stacking velocities than events from structures having shallow dip since the diffracted signals may travel large horizontal distances before reaching the surface (May and Covey, 1983). Post-stack migration also does not take into account triplication of the travel time curve due to caustics. Lenses of low velocity material will cause caustics, regions where raypaths from a single depth point are focussed. In these regions several raypaths connect a single surface and depth point pair. Each of these paths is associated with a different travel time. Since stacking assumes there is only one path from a surface point to a depth point only one of these arrivals will be included in the stack, usually the strongest arrival. Hosken and Deregowski (1985) suggest the presence of caustics is a strong indication that pre-stack migration must be performed in order to extract the subsurface structure from the data correctly. Hatton, et al., (1981) believe that if pre-stack migration is necessary it should be performed in depth rather than in time. Pre-stack time migration is little or no improvement over post-stack migration due to distortions caused by imaging the structure in the time domain.

Depth Migration

Schultz and Sherwood (1980) show that migration in the time domain will not accurately image reflectors lying beneath inhomogeneous material.

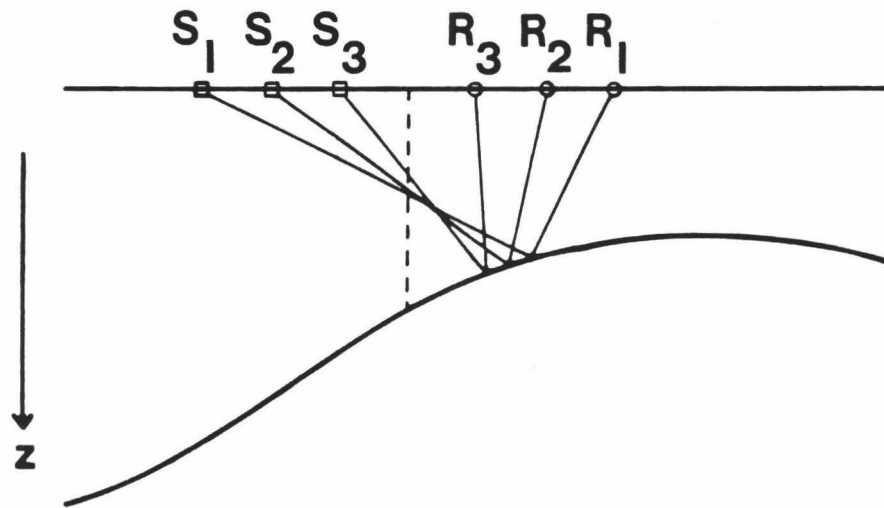


Figure 4. Common depth point gather including signals reflected from points other than the common depth point.

Increases in travel times due to rays bending as they pass through curved interfaces are not taken into account. Conversion to a depth section using correct interval velocities is necessary to correct the shape of deeper reflectors, removing the effect of the overlying velocity structure (French, 1974). Artifacts such as velocity pull-up and pull-down caused by curved interfaces and lateral velocity inhomogeneities are focussed rather than attenuated by Kirchhoff summation migration (French, 1974, 1975; Hatton, et al., 1981). Hubral (1977) and Larner, et al (1981) show that it is impossible to collapse the diffraction curve completely using wave equation time migration if over burden is inhomogeneous, regardless of the migration velocity used. The section must be converted from time to depth. A simple technique for performing this conversion is described by Hubral (1977) and Larner, et al., (1981). Points making up the migrated time section are moved along image rays to their correct position in depth. Image rays are the paths along which the seismic signal emerges perpendicular to the surface (Figure 5). The image raypath is also the minimum travel time raypath. The apex of the diffraction curve, the minimum time diffracted signal, can be mapped to its correct position along the minimum travel time raypath. Using this technique, migration is artificially split into two steps to create a depth section and an accurate velocity model is required for tracing image rays in order to re-position the scattered energy correctly (Schultz and Sherwood, 1980). However, since the conversion to depth takes place after migration, the conversion process may be applied iteratively, refining the velocity model with each iteration (Hubral, 1977; Larner, et al., 1981). This technique does not appear to work well when strong velocity contrasts cause rays to be

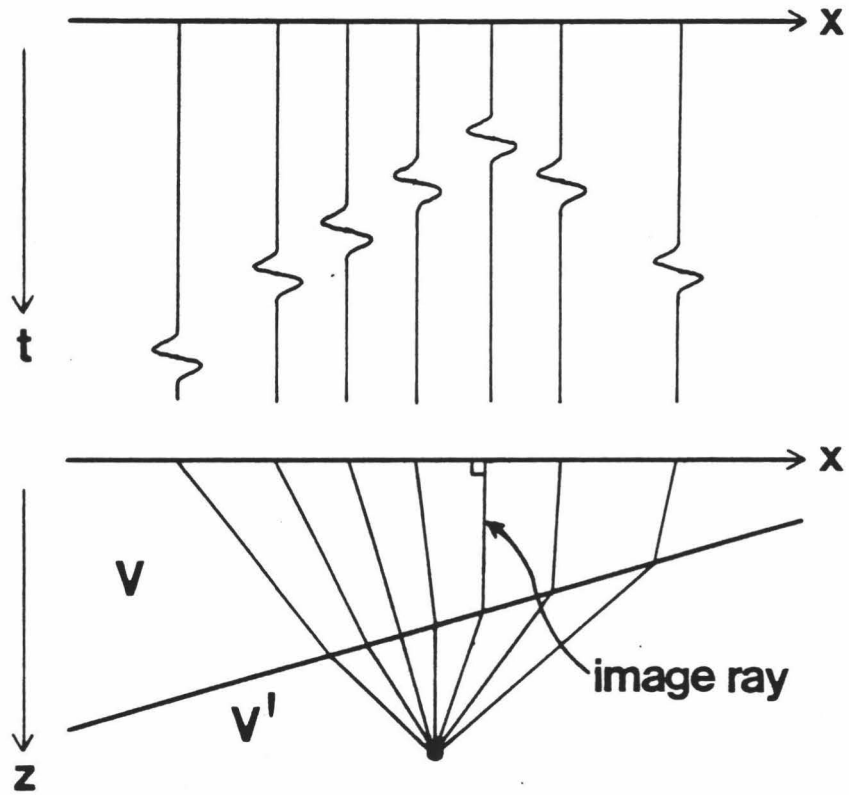


Figure 5. Image ray, emerging perpendicular to the surface, offset from the scattering point.

deflected over large horizontal distances (Judson, et al., 1980; Hatton, et al., 1981).

Schultz and Sherwood (1980) believe that downward continuation migration methods can be used to correct the focussing and defocussing of the signal caused by curved velocity interfaces. By incorporating a time shifting term, a function of interval velocity, into the finite difference algorithm a migrated depth section can be produced in one step (Judson, et al., 1980; Schultz and Sherwood, 1980). Hatton, et al., (1981) recommend depth migration in one step over time migration followed by conversion to depth since the latter may result in the same loss of information as caused by CDP stacking.

Inderwiesen (1985) produces a migrated depth section using ray tracing to obtain accurate travel times to use in a pre-stack Kirchhoff summation migration algorithm. Though the resulting depth section is more accurate, tracing rays from all depth points is time consuming and has previously been dismissed as impractical. Recent advancements in computer technology may make this procedure practical for migrating data from complex structures to resolve detailed structural problems (Inderwiesen, 1985). Reshef and Kosloff (1986) obtain a depth section using wave equation techniques by using the eikonal equation to determine travel times between the source, scatterer, and receiver. Their technique is analagous to tracing rays but requires less computation time at the cost of some loss of resolution.

Velocity Analysis

The accuracy of a migrated section is highly dependent upon the accuracy of the velocities used during migration, regardless of the migration method

employed. Good velocity models may be compiled from prior geophysical surveys and processing which has already been performed on the data set, and from accurate interpretation of the geology. Incorrect migration velocities have different effects on the various types of migration techniques. Kirchhoff summation migration is more likely to under- or over-migrate structure if incorrect velocities are used while finite-difference techniques tend to mislocate scattering centers (DeVries and Berkhout, 1984). Post-migration time to depth conversion techniques such as the method of Larner, et al., (1981) rely heavily on accurate interval velocities to produce a correctly imaged section. Migration or stacking velocities will not be adequate, though they may be used to derive interval velocities.

Gardner, et al., (1974) check the accuracy of the migration velocity model by comparing migrated data taken from the same line but at different offsets. The velocity model for which the semblance of the two migrated sections is highest is the best migration velocity model. These semblance calculations are made using the methods of Neidell and Taner (1971). Such a trial and error approach to obtaining migration velocities can be cumbersome. Berkhout (1984) applies a residual NMO correction to the section after migration to compensate for errors in migration velocity.

Migration appears to be more sensitive to the shape of the velocity interfaces than to the actual values of the velocities (Hatton, et al., 1981). Velocities may be up to ten percent incorrect and still produce sufficiently accurate migrated images (Hatton, et al., 1981). Sattlegger (1975), Schneider (1978), Hatton, et al., (1981) and others advocate refining the velocity model during migration. Migration velocity analysis using unstacked data is preferable to using stacked data since it uses all the velocity information

available in the data rather than an average (Sattlegger, 1975). Migration velocity analysis performed on unstacked data will give better migration velocities than migration velocity analysis performed on stacked data (Sattlegger, 1975). Sattlegger (1975) demonstrates how migration velocity analysis may be carried out using semblance plots, the method of Neidell and Taner (1971). Traces are imaged using a base velocity model. Coherence of these traces is calculated and plotted on a semblance plot. High semblance is obtained for events on the output trace which migrate best using a particular base velocity. Several different models are used to find the best velocities for each portion of the section. Migration velocities are picked directly from the semblance plots.

Owusu, et al., (1983) found Sattlegger's (1975) velocity determination technique effective but prohibitively expensive. Instead, they used the instantaneous power in the migrated section to estimate the accuracy of the velocity model used. At large depths this method will produce abundant migration noise.

Common midpoint stacking velocities were used by Hosken and Deregowski (1985) with good results. To obtain stacking velocities coherence was calculated between two sets of data at their line of intersection. This method gives velocities which are well constrained but is limited to regions with dense seismic coverage and where sideswipe is minimal. In addition, the stacking velocities obtained must be subsequently converted to migration velocities.

IV. RAY TRACING

In order to migrate scattered energy back to its source, without making assumptions about the shape of the diffraction curve, rays must be traced from each possible source point in the migrated section. Previously, ray tracing as a preliminary step to migration was prohibitively time consuming (Carter and Frazer, 1983; Deregowski and Brown, 1983; Inderwiesen, 1985). Instead of using ray tracing to determine the shape of the diffraction curve the curve was assumed to be hyperbolic (French, 1974, 1975; Gardner, et al., 1974; Hubral, 1977; Schneider, 1978). Developments in computer technology have speeded up ray tracing routines and given the user more storage, thus making ray tracing potentially more feasible as a part of seismic multichannel data migration.

Whittall and Clowes (1979) developed a ray tracing method for finding raypaths and travel times in which rays were traced through a velocity model consisting of plane isovelocity interfaces separating layers or blocks having a linear velocity gradient. Head waves and reflected ray arrivals are computed, but multiples, converted phases, and sub-critical incidence reflected rays are ignored.

Inderwiesen (1985) developed a migration routine which uses the ray tracing method developed by Deregowski and Brown (1983). Migration is performed by evaluating the Kirchhoff integral using travel times from the traced rays. Since the diffraction curve is not assumed to be a hyperbola, lateral velocity variations are easily accommodated. Interfaces separating the isovelocity layers of the velocity model were defined using cubic splines. Finding the intersection of a ray with an interface involves solving a cubic

equation and looking at the segments of the spline making up the interface. Snell's law is used to calculate the angle of departure of the ray from the interface.

Reshef and Kosloff (1986) developed a faster ray tracing technique for migration. Rays are traced downward from the shot location and downward from each receiver location, at a variety of takeoff angles. At the points where the rays from a shot and a receiver intersect the travel time is obtained. Rays are continued until they reach the bottom of the section. Gray (1986) improved the efficiency of ray tracing by a similar method (Appendix B). In both cases eliminating redundancy in the ray tracing procedure greatly reduces the computation time required to obtain travel times by ray tracing.

Gebrande (1976) developed a method of ray tracing which is well suited to models made up of continuous layers having smoothly varying velocities. However, the routine is not well suited to laterally varying velocities or discontinuous layers.

In this thesis a simple ray tracing program was developed, similar to that of Whittall and Clowes (1979), which easily accomodates strong lateral velocity variations and discontinuous layers. Rays are traced from each point (x,z) for a range of takeoff angles to the boundary of the polygon containing (x,z) (Figure 6). At the boundary the direction of propagation changes, according to Snell's law. Each ray is traced through successive polygons until it reaches the surface. Multiples and refracted rays are not computed and are assumed to be strongly attenuated. A major limitation of this simple ray tracing method is that it does not directly accept velocity models which contain velocity gradients. Velocity gradients may be simulated by thin isovelocity layers.

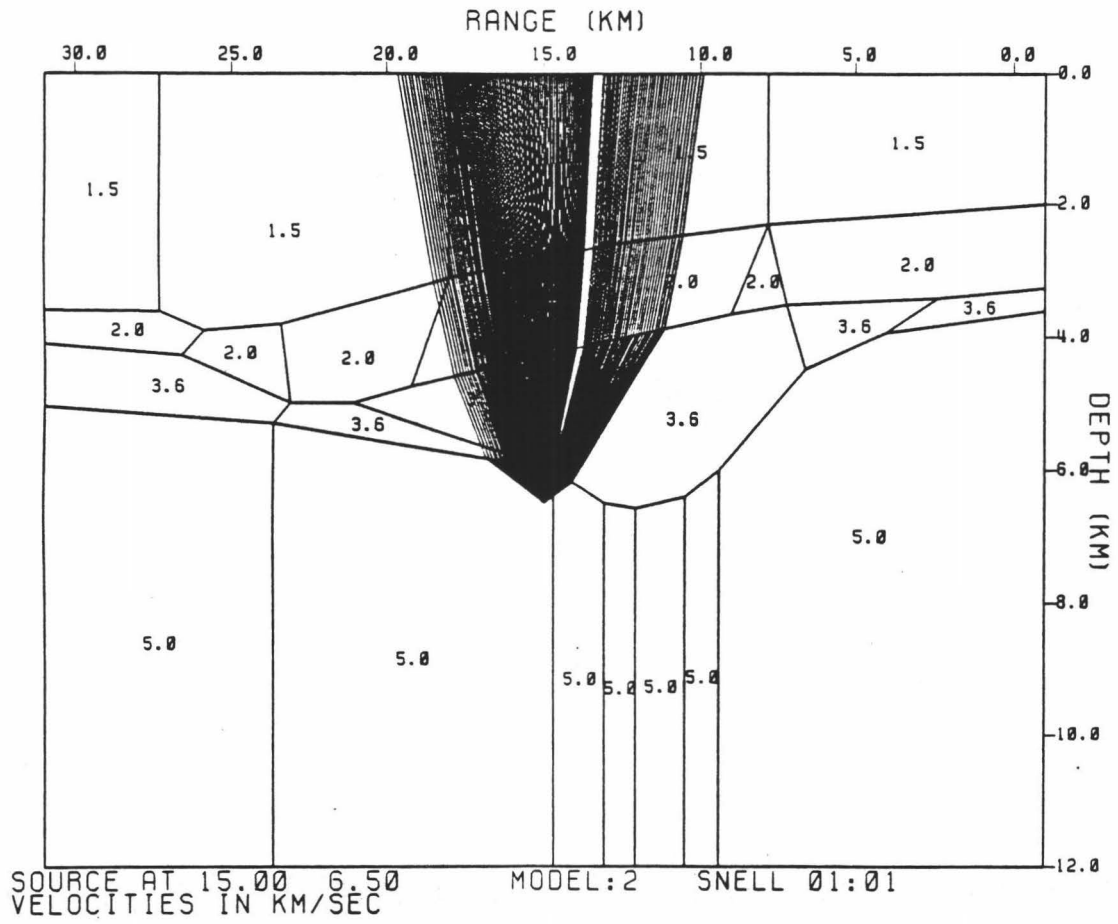


Figure 6. Test model and ray trace for a single depth point.

The velocity model is constructed by dividing the subsurface (the section to be migrated) into polygons of constant velocity (Figure 6). Construction of this model is rapid and locating points of intersection between traced rays and block boundaries is simple. Discontinuities in travel times due to corners on interfaces in the velocity model do not appear to be significant.

The ray tracing routine interpolates the travel times to generate a table of travel times $T(x',z,x)$ where x' is the emergence point of the ray at the surface, z is the z -coordinate of the source point, and x is the x -coordinate of the source point (Figure 6). Since two-way travel times are recorded in the data and one-way ray paths and travel times are calculated by the ray tracing program, this procedure utilizes the reciprocity property of ray paths.

The grid of depth points from which rays were traced is used to construct the migrated section. Vertical separation of depth points must be fine enough to adequately sample the highest frequencies of interest in the data. Since most seismic multichannel data are acquired at a sampling rate high enough to insure adequate sampling of the minimum period to be resolved in the data, the vertical separation between depth points in the migrated section can be chosen based on the data sampling frequency.

Vertical distance between depth points must be small enough to sample the minimum wavelength of interest in the data at least twice. The wavelength at depth z is dependent upon the minimum period, T_{min} , and the velocity, V , at that depth:

$$L(z) = V(z) T_{min} \quad (1)$$

Since a constant distance between depth points simplifies coding, the minimum wavelength of interest depends on the minimum velocity predicted to occur within the migrated section as well as the minimum signal period in the data. To adequately sample the migrated section, the distance between depth points Δz should be at least half this minimum wavelength.

During data acquisition, the sampling rate Δt is chosen to be less than one half the smallest period in the source wavelet to insure that aliasing will not occur. If the wavelet period T_{\min} is constant and $v(z)$ increases, then according to equation (1) the wavelength of the wavelet $L(z)$ must increase, regardless of the sampling rate in time or depth. Decreasing the distance or time between samples will merely sample the same wavelet more often (Figure 7).

Shot separation and receiver separation determine the upper limits of the lengths of shot aperture, D_S , and receiver aperture, D_R , respectively, necessary to avoid aliasing. To calculate this distance D_S (D_R), we must find the maximum angle of incidence, θ , between a wavefront and the surface for which the travel times to adjacent shot or receiver locations differ by less than half the minimum period in the data, T_{\min} (Figure 8). The angle θ is calculated using:

$$\sin \theta = (v T_{\min} / 2) / \Delta x. \quad (2)$$

Wavefronts arriving at angles greater than θ will be aliased.

If velocity can be assumed constant the distance D is related to θ and depth in a straightforward manner (Figure 9):

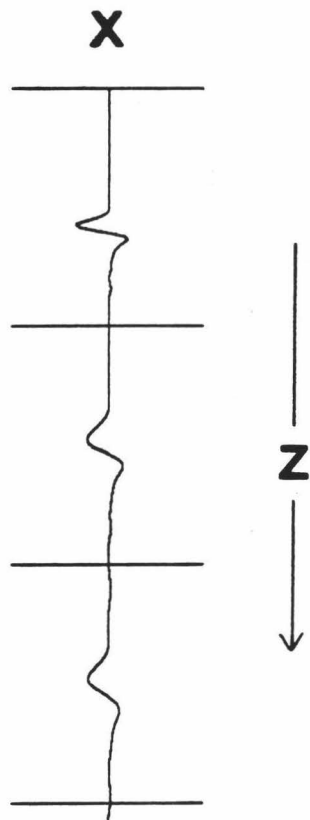


Figure 7. Wavelet sampled adequately, avoiding aliasing, showing stretching with depth.

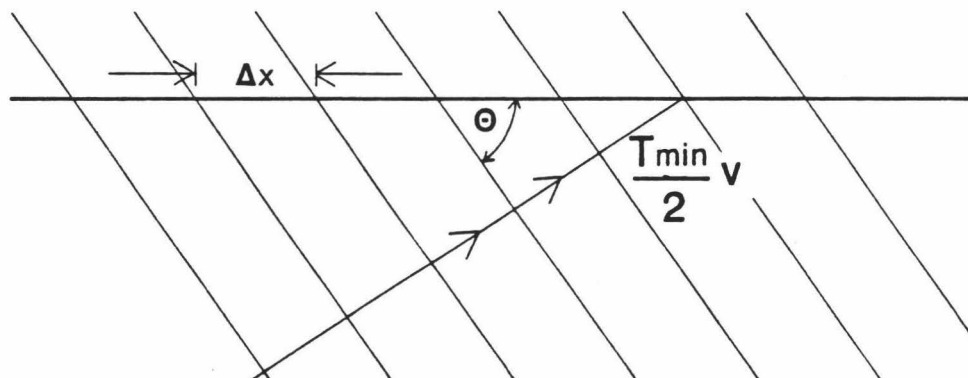


Figure 8. Maximum angle θ for which wavelet impinging on the surface will not be aliased at trace separation Δx .

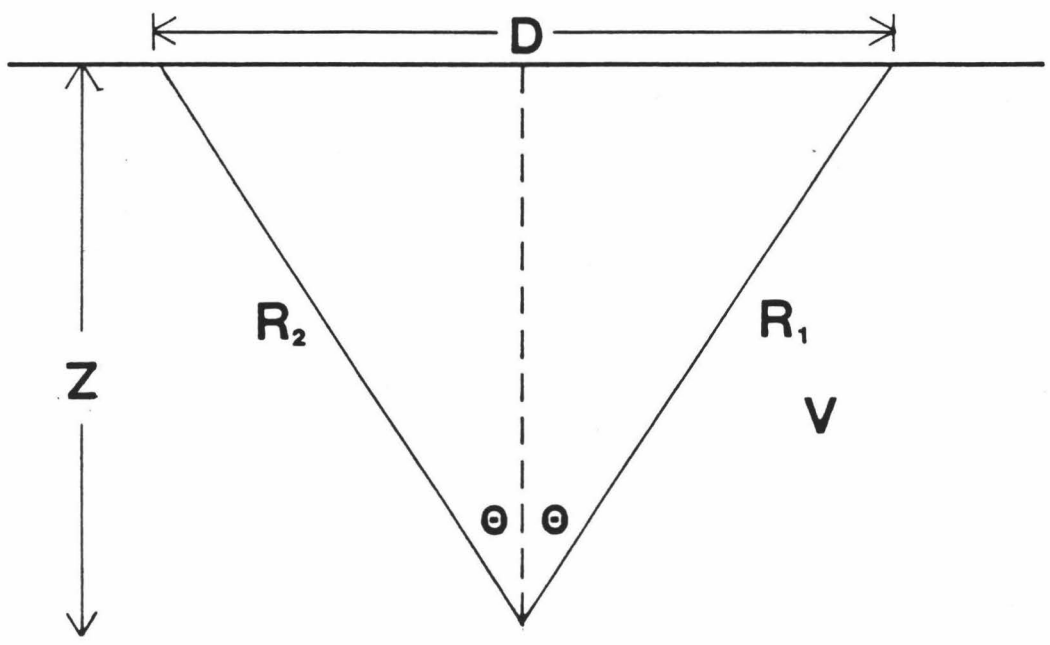


Figure 9. Maximum shot - receiver separation (D) without aliasing, dependent on incidence angle, θ , of wavefront at the surface.

$$D(T_{\min}, / x, z) = (2z)/\tan \theta. \quad (3)$$

For models in which the velocity increases with depth a larger value of D_S (D_R) than that given by equation (3) can be used. Rays are focussed as they travel through a medium with a velocity gradient. A wavefront travelling at an angle θ to the horizontal at depth will arrive at the surface with an angle to the horizontal less than θ due to focussing. In general, velocity increases with depth, therefore the value of D_S (D_R) should also increase with depth to include as much of the non-aliased signal as possible. Since the minimum period of the data, shotpoint (receiver) separation, and velocity structure are all pre-determined, the value of D_S (D_R) at each depth z has a fixed maximum value.

Horizontal depth point separation must be small enough to adequately sample the structure. Decreasing horizontal depth point separation will improve the resolution but below a minimum distance, determined by minimum period, shot separation, depth, and velocity, the horizontal resolution of the migrated section cannot be improved.

To determine horizontal resolution, we use the greatest possible distance between a shot and a receiver within the maximum shot and receiver apertures, $D(T_{\min}, \Delta x, z) = (D_S + D_R)/2$. Horizontal resolution $d = d(D, T_{\min})$ is the distance a depth point (x, z) centered beneath an aperture of width D , can be moved laterally before the travel time from the depth point to a point at the surface on the margin of the aperture increases by more than $T_{\min}/2$ (Figure 10):

$$t_1 - t_2 = T_{\min}/2. \quad (4a)$$

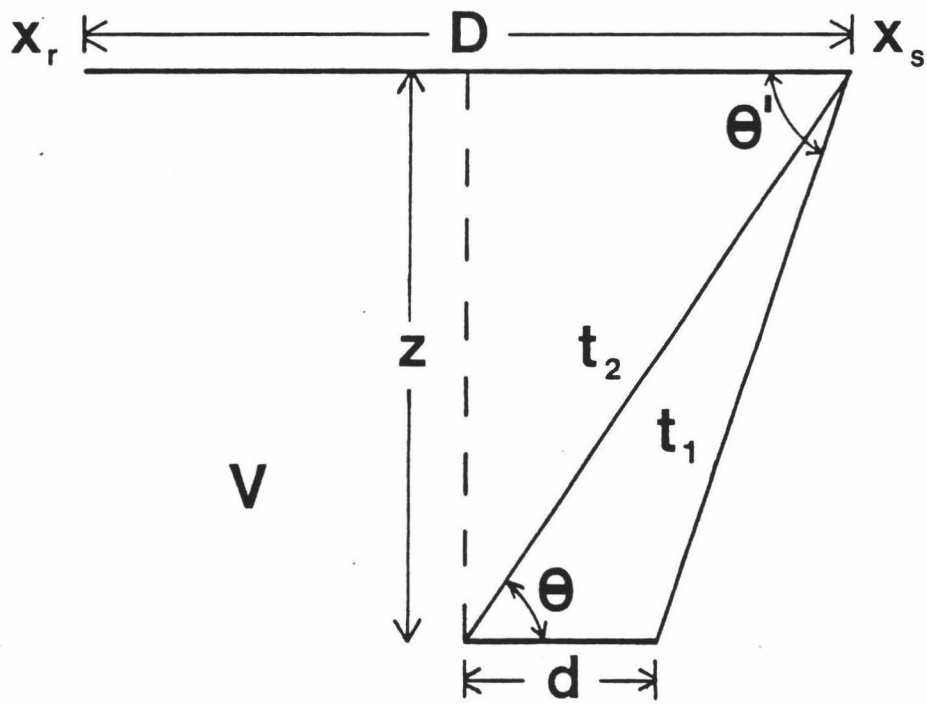


Figure 10. Horizontal resolution d as a function of aperture D_s (D_r) and wavelet period T_{\min} .

For $D \gg d$:

$$d = (t v^2 T_{\min}) / D \quad (4b)$$

where t = travel time from the depth point (x, z) to surface point $x_s (x_r)$.

For a fixed depth, as D increases horizontal resolution decreases since θ more closely approximates θ' . For fixed depth and fixed $D_s (D_r)$ horizontal resolution decreases with decreasing velocity. Therefore, the horizontal depth point separation should be chosen equal to the horizontal resolution in the region of interest having the lowest velocity. This depth point separation will adequately sample depths where velocity is low and will oversample everywhere else.

Complex velocity models, such as that used in this paper, cause focussing of raypaths at caustics. Where caustics occur, the travel time curve has multiple arrivals for a single range. For the calculations shown in this thesis only the first arrivals were used; arrivals along the later branches of the travel time curve were ignored.

Ray Tracing Through the Test Model

The test velocity model used in this thesis is a simplified version of a velocity model which might be used to migrate data collected on the lower forearc of a convergent plate margin such as the Peru margin. The data migration programs developed in this thesis were originally designed to resolve problems in processing and structural interpretation of 24-channel seismic data collected on a dip line from the Peru-Chile trench toward the

shore on the seaward edge of the Peru continental margin. The velocity model generated for this thesis is made up of convex polygons enclosing regions of constant velocity as shown in Figure 6. For this model truncated velocity layers could not be included due to the limitations of another, entirely different, program used to generate the synthetic data. The model is, however, sufficiently complex to determine how well this migration routine will work on a real multichannel data set.

Horizontal depth point separation is 1.0 km, equal to that required to match resolution available in the data at the lowest velocities at the shallowest depth according to our criteria as described above. Vertical depth point separation is much smaller, 0.024 km, determined by an 8.0 msec sampling frequency. At 3.0 km/sec velocity, an 8.0 msec sampling rate will sample every 0.024 km. Greater resolution is actually available at shallower depths and lower velocities but in order to conserve CPU time a larger distance between depth samples was used than that required to match the highest vertical resolution available in the data. This compromise does not appear to have degraded the accuracy of the migrated section.

The entire section for which rays were traced was made up of 8401 depth points, from 1.6 km depth to 8.1 km depth and from 0.0 km range to 30.0 km range, neglecting much of the water column and the uniform 5.0 km/sec portion of the section (Figure 6). For each point rays were traced to points 0.25 km apart at the surface over an aperture 4.0 km wide. Emergence point spacing corresponds to the distance between shots in the synthetic data and the aperture length is equal to the maximum migration window length which would be used during migration of the seismic data set. Tracing rays for this section required 2 hrs and 42 mins of CPU time on a HARRIS H800 computer.

V. THE SYNTHETIC DATA SET

Ray theoretical synthetic data were generated using a program written by Mrinal K. Sen which uses Gebrande's (1976) algorithm to define two dimensional interfaces. Arc tangent functions were used to generate a model as close as possible to the model used to calculate travel times (Figure 11). Unlike the travel time model these interfaces are smooth. Travel times for each shot - receiver pair were calculated for rays reflected from each interface. This data set was then convolved with a 2.5 Hz wavelet to produce the synthetic data set to be migrated (Figure 12). This relatively low frequency was chosen so that a relatively large separation between shots and between depth points could be used without causing serious aliasing problems.

Synthetic traces were calculated for four receivers per shot, 1.0 km apart. This configuration was designed to completely cover the widest anticipated migration window while minimizing computation time. Shots were generated 0.25 km apart. Though this is a smaller separation than used in actual data collection for the given receiver separation, it provides better coverage of the section than would be obtained for 0.5 km shot separation without the added computation time required for added receivers.

The sampling interval for the synthetic data set was 8.0 msec and the trace length was 8.1 sec. This sample rate is sufficient to adequately sample the 2.5 Hz signal and this trace length insures that the deepest reflector will be sampled by the farthest shot - receiver pair. A lower frequency would allow a lower sample rate to be used but would not be able to resolve the model in regions where the interfaces are close together.

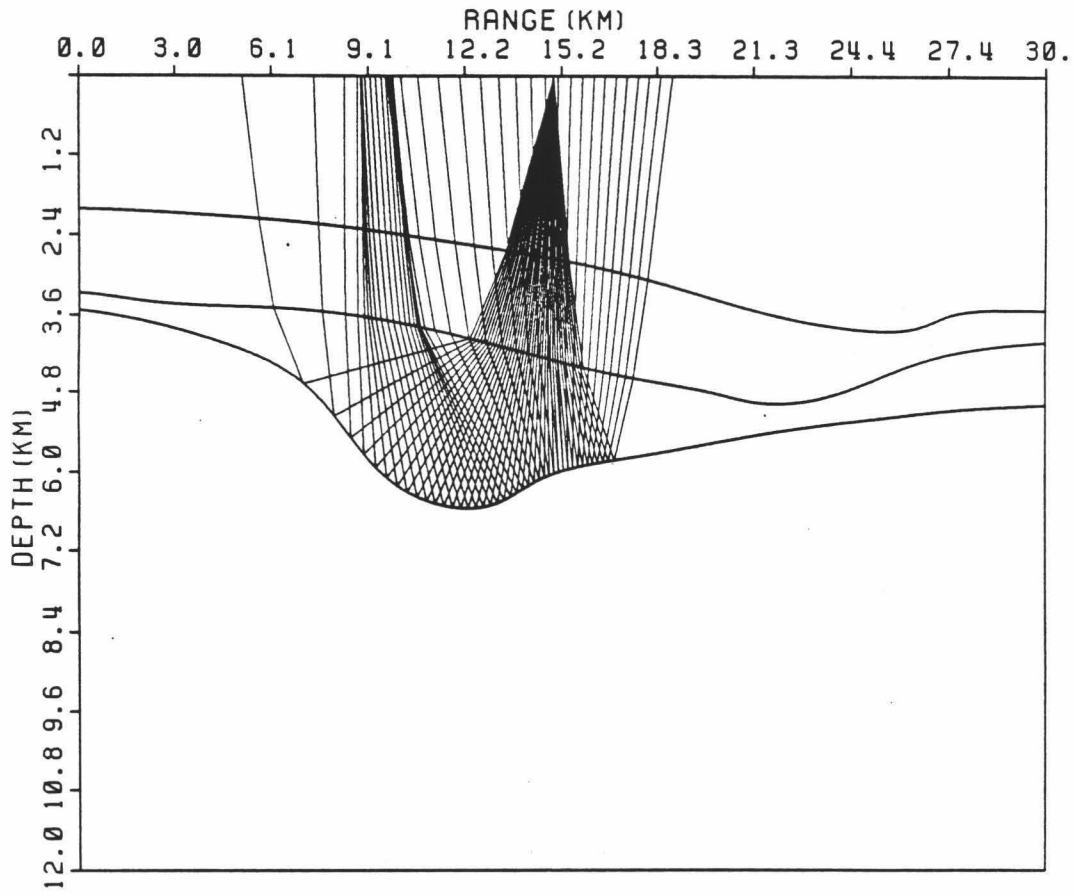


Figure 11. Test model used to construct synthetic data set, showing a single shot reflected from the third interface.

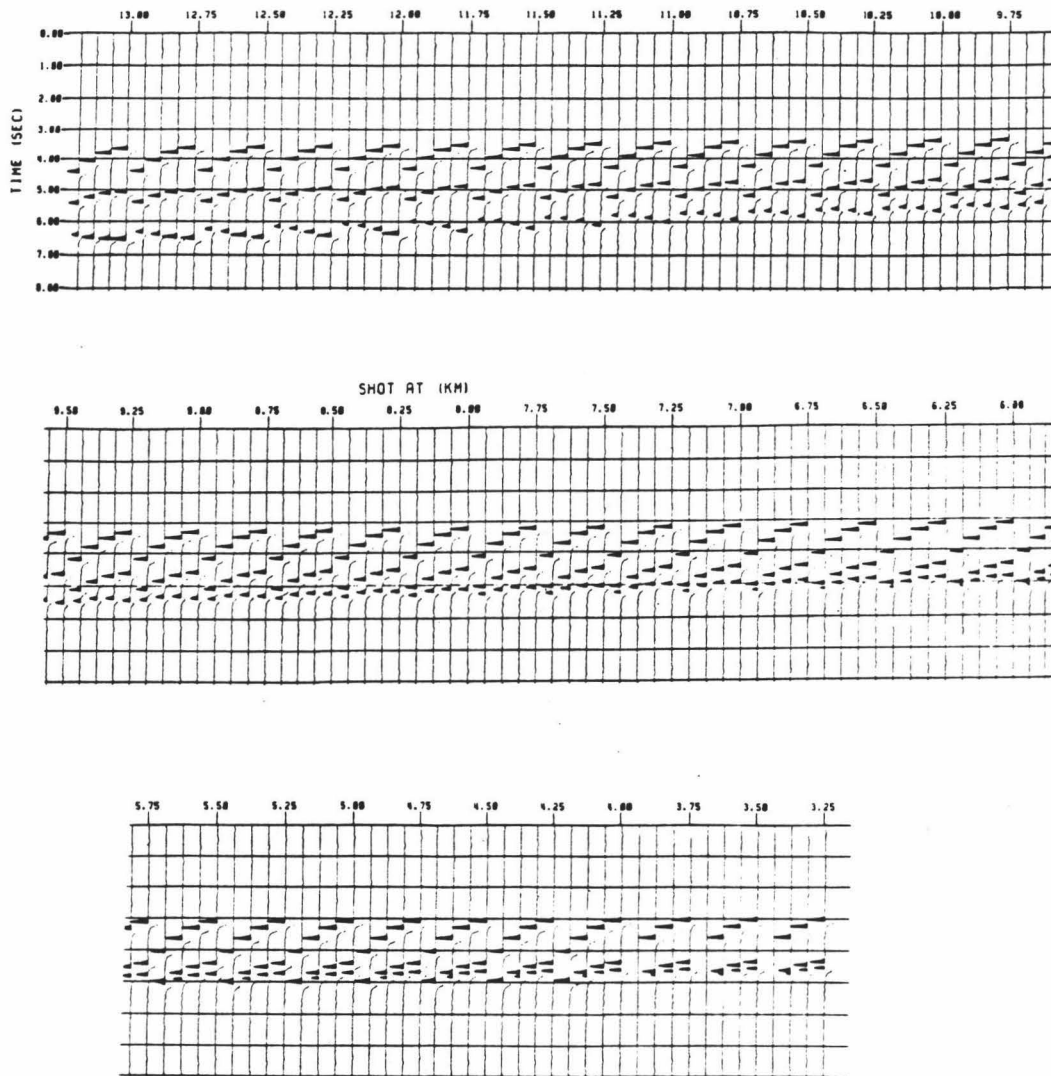


Figure 12. First 40 shots of synthetic data.

VI. MIGRATION

The Kirchhoff migration method used here is mathematically simple. Energy scattered from each depth point is summed and put back at that point. Each point in the subsurface is assumed to act as an independent point scatterer, reflecting incident energy equally in all directions (Trorey, 1971; Waters, 1981). Since this is a two-dimensional migration we must also assume that the geologic structure does not vary in the direction perpendicular to the seismic line. Two-dimensional migration in regions in which the structure is not really two dimensional will focus, rather than eliminate reflections from features lying off the seismic line (French, 1974). Note that we do not assume that the overburden velocity is uniform.

For each shot s and receiver r the travel time from a given subsurface point (x,z) to the shot and receiver is summed to give the two-way travel time:

$$TT(x_S, x_R, z, x) = (T(x_S, z, x) + T(x_R, z, x)). \quad (5)$$

All points for which (5) is true form an "ellipse" in space. The envelope of all ellipses for all shot - receiver pairs which recorded energy from a given reflector (Figure 3) determines the spatial position of that reflector (Reshef and Kosloff, 1986). During migration the amplitude $P(x_S, x_R, t)$ of the time sample in the data trace from the shot at x_S and the receiver at x_R at travel time $TT(x_S, x_R, z, x)$ is summed into the "bin" $M(x,z)$ for point (x,z) :

$$M(x,z) = 1/(N_S N_R) \sum_{x_S} \sum_{x_R} P(x_S, x_R, TT(x_S, x_R, z, x)) \quad (6)$$

where $N_S N_R$ is the total number of samples summed into $M(x,z)$. When the amplitudes from each point (x,z) within range of a given shot - receiver pair have been summed into each bin $M(x,z)$, data from the next shot - receiver pair are considered. Stacking, migration, and time to depth conversion occur simultaneously. The stacking velocity used here is the same as the migration velocity, resulting in a more accurately migrated section (Sattlegger, 1975).

There are two ways of constructing the migrated depth section from a seismic data set. One method is to input the data set and build and output the migrated section trace by trace (Figure 13). Travel time $t(x_S, x_R, z, x)$ associated with depth point (x,z) for each shot - receiver pair (x_S, x_R) is obtained from the travel time table generated by tracing rays. Using these travel times, the diffracted signal spread over the shot gathers is collapsed to a pulse at its point of origin. The amplitude in the data trace $P(x_S, x_R)$ at time $t(x_S, x_R, z, x)$ is summed into the "bin" $M(x,z)$ for the given point (x,z) . When all shot gathers within the migration aperture have been summed into each depth point in the migrated trace (Figure 13) that trace is output and the next trace is constructed from the data set. The migrated section is constructed sequentially trace by trace. Each trace of the migrated section is constructed from all the shots within the migration aperture. All shot gathers within the migration aperture must be contained within the computer core memory.

The second method of constructing a migrated section does the opposite. The migrated section is held in memory and the data are processed trace by trace. Each data sample is spread over a curve of constant travel time defined by the depth points lying at equal travel time from the shot and receiver (Figure 2, Figure 14).

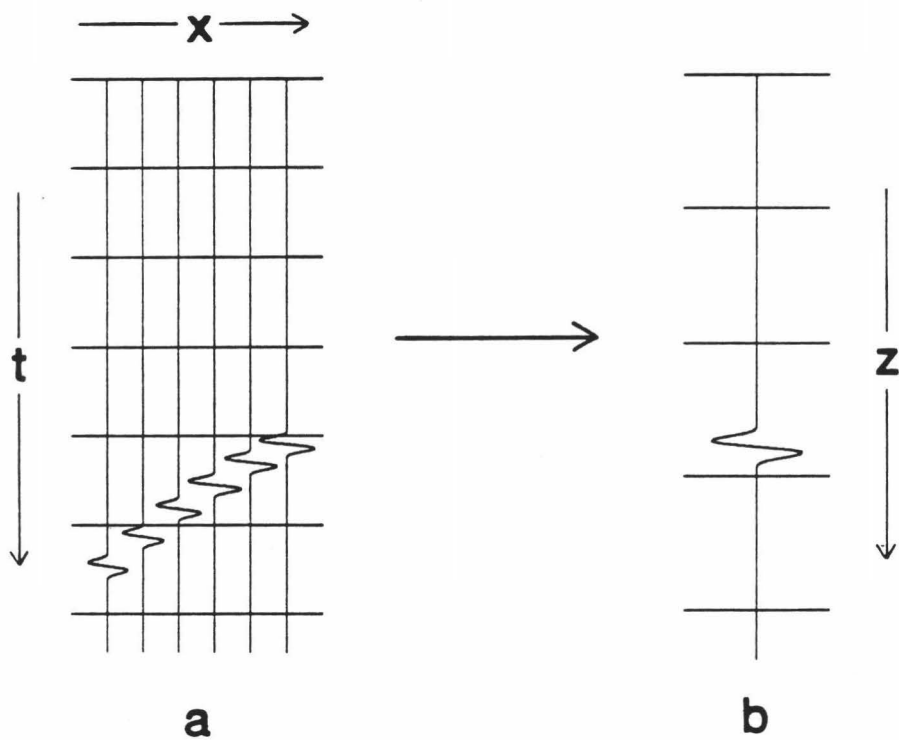


Figure 13. (a) Single common shot gather.

(b) Migrated trace formed by summing over all the shot gathers.

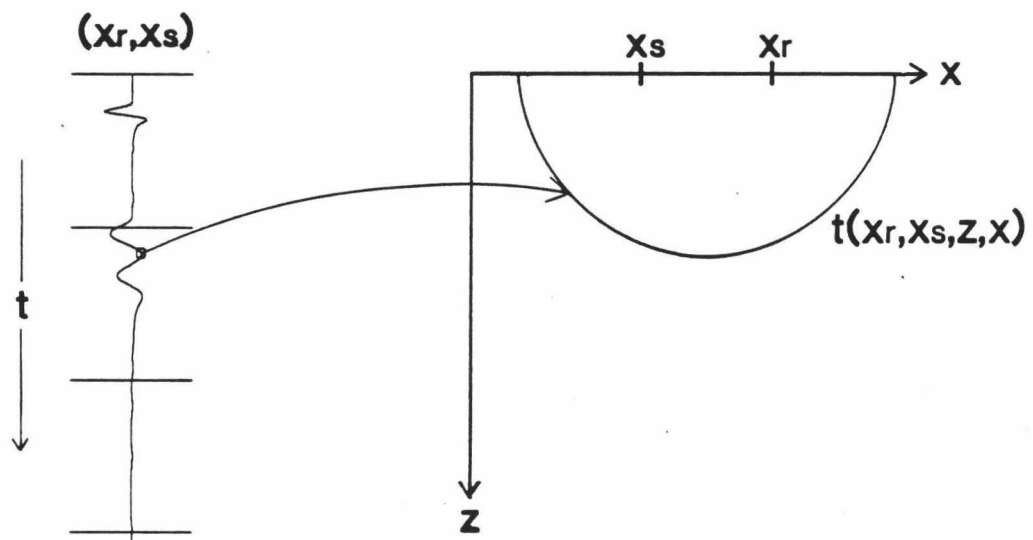


Figure 14. Construction of depth migrated trace by spreading each data sample over iso-travel time curve $t(x_s, x_r, z, x)$.

Data are arranged in common shot gathers. Each data trace from shot - receiver pair (x_s, x_r) is input individually. For each depth point (x, z) in the migrated section within range of the given shot - receiver pair the travel time $t(x_s, x_r, z, x)$ is obtained from the travel time table. The amplitude at travel time $t(x_s, x_r, z, x)$ in the data set is summed into the "bin" $M(x, z)$ for depth point (x, z) (Figure 14). After a data sample from trace (x_s, x_r) has been added to the last depth point within the aperture the next trace is input and spread over the migrated section. Each data trace is looked at only once. Since the volume of data composing the migrated section is much less than the volume of the shot gathers this second method requires less computer memory.

The second method was used to construct the migrated section described in this thesis. Due to the limited computer memory available, not all of the shot gathers within the migration aperture used could not be held in memory.

The amount of migration noise resulting from each data sample being spread over an iso-travel time curve increases with increasing window width, if the anti-aliasing criteria derived above are not adhered to. Following Gardner, et al., (1974), Inderweisen (1985) determined window length such that the difference in travel time between the specular reflection and an arrival from a single point scatterer to the far end of the window would be no greater than the dominant period in the data. Such a window width would minimize migration noise and include the major contributions to the signal from that point. I emphasize here that such a guideline, if used by itself, can result in a spatially aliased depth section if shot and receiver intervals are not sufficiently fine.

The center lobe of a sinc function was found to be a simple but effective weight function $WT(x)$ for each shot or receiver location, x , within the window:

$$WT(x) = \sin(\pi X) / (\pi X). \quad (7)$$

where: $X = \Delta x / (W/2)$

Δx = distance between emergence point and center of window

W = window width

Since the window is centered above the scatterer it does not guarantee summation of all major contributions from the scatterer, especially if a small window is used. However, use of such a tapered window is necessary in order to minimize Gibbs effect in the migrated section. The window (7) is actually applied to both shot and receiver. Thus the window weighting applied to a single time sample $TT(x_S, x_R, x, z)$ is the product $WT(x_S)WT(x_R)$.

Migration of the Test Model

Since the migrated section, travel times, and data set are too large to be stored in the computer, one data trace at a time is processed through a sliding migration aperture. The data trace is distributed over the portion of the depth section within the aperture. After use, it is discarded and the next data trace is read from tape. When the data goes out of range of the migration aperture the aperture slides down the section; a new column of depth points, and associated travel times, are added to one end of the aperture and

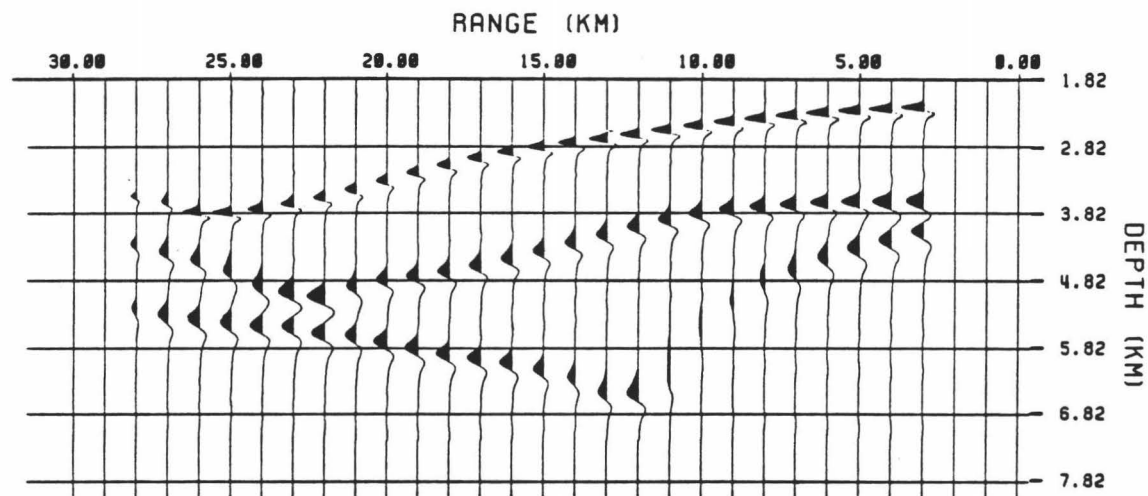
the migrated column of depth points at the opposite end is output to disk. Only the data trace currently being processed and the portions of the travel time table and migrated section within the current aperture are held in memory.

Before output, each trace is normalized by a factor which is a function of the number of samples summed into the "bin" for each depth point. This helps to restore any amplitude lost artificially during the migration process. The zero trace on the margins of the migrated section (Figure 15) are due to inadequate coverage of these traces by the data.

Migration of the model was carried out for 1.6km to 8.1 km depth (Figure 15). The migration window was chosen to range from 2.0 km at 1.6 km depth to 3.0 km at 5.4 to 8.1 km depth. The window near the base of the section is narrower than that determined using Gardner, et al's (1974) methods but this range of lengths produced a better migrated section with minimal migration noise from our limited data set. Migration of this section on a HARRIS H800 computer required 22 minutes of CPU time.

The reflectors are correctly imaged in the migrated depth section but the frequency of the reflected signal appears to decrease as depth increases. Stretching of the migrated signal with increasing depth is due to the conversion from time to depth. A wavelet of period T arriving from a region having velocity $v(z)$ is converted to a wavelet of length $L(z)$ in the migrated depth section using the relation:

$$L(z) = v(z) T. \quad (8)$$



NUMBER OF SHOTS 109
RECEIVERS PER SHOT 4
WINDOW WIDTH: 2.00 KM TO 3.00 KM
SHOT SEPARATION = 0.25 KM
RECEIVER SEPARATION = 1.00 KM
SOURCE FREQUENCY = 2.50 HZ

Figure 15. Depth section obtained by migrating synthetic data set.

Velocity $v(z)$ is controlled by the velocity structure of the model being used. For any fixed period T , invariant laterally and with depth, the wavelength will be longer where velocity is higher, regardless of the values of the other parameters.

Steeply dipping portions of the reflecting interfaces did not migrate well (Figure 15) since much of the reflected energy is scattered away from the region of steep dip, rather than up toward the surface. To recover more energy from these steeply dipping interfaces a larger migration aperture must be used. In order to use a larger aperture without introducing migration noise due to aliasing either the shot and receiver separations must be decreased or the data must be low pass filtered. Inderwiesen (1985) increased the amount of energy migrated to steeply dipping interfaces by centering the migration aperture around the emergence point of the image ray from the depth point being migrated rather than the surface point directly above the depth point. Gardner, et al (1974) noted that tapering the migration window, as is done by this routine, also decreases dip resolution.

VII. THE COHERENCY SECTION

While evaluating the reflectivity function $M(x,z)$ using equation (6) we can simultaneously accumulate the function $A(x,z)$ given by:

$$A(x,z) = 1/(N_S N_R) \sum_{x_S} \sum_{x_R} |P(x_S, x_R, TT(x_S, x_R, z, x))|. \quad (9)$$

We define the "coherency depth section" $C(x,z)$ to be the ratio of M and A :

$$C(x,z) = M(x,z)/A(x,z). \quad (10)$$

The purpose of the coherency section is to provide an objective evaluation of the velocity function in a downward pointing cone with vertex at the point (x,z) and base along the union of the shot - receiver aperture at the surface (Figure 16(a)). Suppose we have a large number of reflecting horizons distributed more or less uniformly with depth. If the velocity function is bad in a small region centered about the depth point (x,z) then the coherency section will reveal an upward pointing cone of low coherence with vertex at the point (x,z) (Figure 16(b)). The function $C(x,z)$ will be a minimum at the vertex of the cone and will gradually increase with depth. Thus the boundaries of the cone will be sharpest near the vertex (a desirable trait) and less distinct at increasing depth. At this point I do not know if $C(x,z)$ has enough value as a diagnostic tool to justify the expense of computing it. When the velocity function is bad everywhere then the cones described above

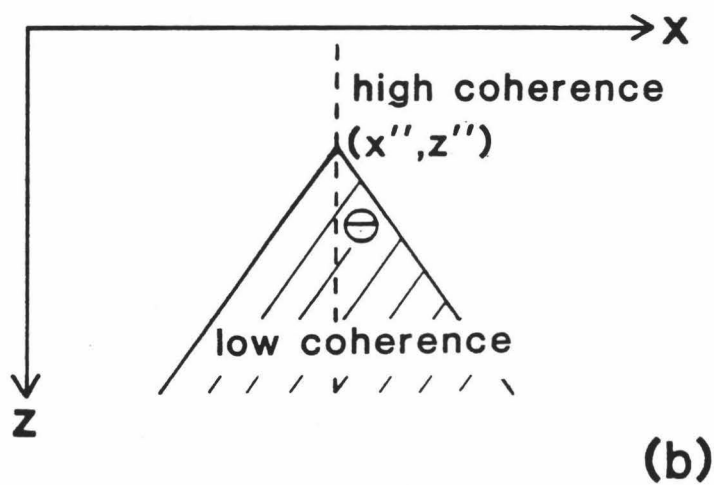
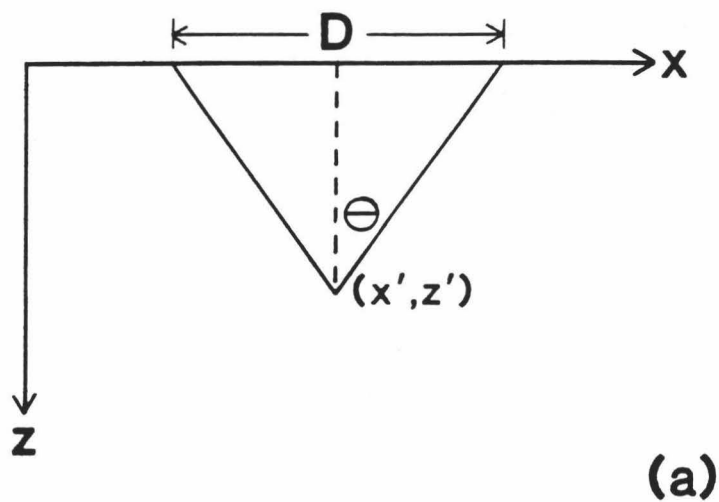


Figure 16. (a) Region of velocity model affecting coherence value $C(x', z')$.
 (b) Region of low coherence caused by bad velocity estimate at (x'', z'') .

may not be apparent. Hence it seems likely that the coherency section may be useful only as an aid to final refinements in velocity.

VIII. SUMMARY

Computer programs were developed to trace rays through a complex velocity model and to migrate seismic multichannel data obtained in regions of complex structure. The traced rays provided accurate travel times for use in the pre-stack Kirchhoff integral depth migration algorithm. Migration performed using such travel times is more accurate than migration schemes which assume that the diffraction curves are hyperbolic or that velocity is horizontally slowly varying. In addition, the section is simultaneously converted from time to depth, eliminating the distortion of the interfaces caused by inhomogeneous overburden. Figure 15 illustrates that highly curved reflecting surfaces are correctly migrated by the Kirchhoff integral migration routine using travel times obtained by tracing rays. Regions of steepest dip have poorer resolution due to rays being strongly deflected by the high angle velocity interface and reaching the surface outside the migration window. To include these arrivals the migration window could be widened but the added noise would exceed the added signal.

In order to determine whether this is in fact a useful data processing tool these routines need to be applied to a real data set obtained in a structurally complex region. Successful migration of the synthetic data indicates this migration technique extracts an accurate depth section from an unstacked time section by correctly shifting the signals. However, the method is time consuming. Rays must be traced for many subsurface points and the complete unstacked data set must be processed. Sensitivity of the migration to the accuracy of the velocity model has not been thoroughly investigated. Testing the technique on a real data set would help determine if the added time

and expense are proportional to the improvement in the seismic section. Incorporation of Gray's (1986) downward ray tracing method, implemented using principles of dynamic programming, as described in Appendix B, would significantly increase the speed of the ray tracing making this data processing scheme more practical.

This Kirchhoff summation migration scheme is flexible, and has potential for improvements not researched in this thesis. For example, caustics could be accurately migrated since travel times associated with all raypaths between a given shot, subsurface point, and receiver are all calculated by the ray tracing method described. Transmission loss across velocity contrasts could easily be included in the ray tracing algorithm, enabling more accurate relative amplitudes to be produced in the migrated section.

APPENDIX A

PROGRAM LISTINGS

Ray Tracing:

NAME SNELL

```

C
C SNELL version 01:01 - character strings rearranged so plot is right side
C up.
C
C
C -----
C
C Program traces rays from a set of subsurface points to the surface
C through blocks of uniform velocity using Snell's Law reflection and
C refraction.
C PHI is the angle between the ray incident on an interface and the
C interface, measured from the perpendicular to the interface to the
C ray.
C THETA is the angle of travel of the ray measured from the +X axis to
C the ray;  $-\text{PI} < \text{THETA} < +\text{PI}$ .
C
C Velocities must be read from input file clockwise, if range of the
C section increases from left to right, or anti-clockwise if range
C increases from right to left.
C
C Special note -- Range of takeoff angles is determined by the width
C of the window for which rays will be traced from the shallowest
C depth. Since rays, in general, converge as they travel upward, due
C to decreasing velocity with decreasing depth, the aperture defined
C by the takeoff angles may be narrower than that specified by sub-
C routine WINDOWS. Travel times for receiver locations outside the
C maximum range of the traced rays are set equal to zero.
C
C -----
C
C Arrays used by the program:
C BLOCK - vertices of each block of uniform velocity
C NVERT - number of vertices in each block
C VEL - velocity of each block
C M - slope of each side of each block
C B - z-intercept of each side of each block
C WINDO - aperture width for each depth z
C TTIME - travel time for each ray to reach surface (NTHETA rays) for one
C RANGE - range at which each ray reaches the surface point
C TTIME and RANGE will be binary files!
C TT - interpolated travel times writtern to external files

```

```

C R - interpolated range coordinates   in binary
C PLOTX - x-coordinates of ray segments (for plotting)           for
C PLOTZ - z-coordinates of ray segments (for plotting)           one
C IP - number of points joined by ray segments in the traced ray point
C
C Indices:  ITH is index for takeoff angle, NTHETA is index for ray reaching
C           the surface.
C
C External files referenced:
C 20 -- INSNELL;  input file containing input parameters and coordinates
C                of uniform velocity blocks, last entry in the file must
C                be 999.
C                The limits of the ray trace section should be
C                greater than the limits of the section to be migrated
C                (in the x-direction) in order for travel times to be
C                interpolated for receivers on the margins of the section.
C                Spacing between x-coordinates should be equal to the
C                CDP spacing for stacking the section to preserve maximum
C                horizontal resolution.
C 30 -- RANGE;  output file, contains range coordinate where ray reached
C                surface, rays from each source point are separated by
C                .9999. File is binary.
C 50 -- TTIME;  output file, contains travel times for each ray to reach
C                surface, rays from each source point are separated by
C                .9999. File is binary.
C 40 -- CRUD;  junk file for program errors, debug statements, program
C                execution info.
C
C ***** SAMPLE INPUT FILE - INSNELL *****
C
C Plot of ray trace                FALSE
C X - T plot                        FALSE
C start source; x-coordinate        0.000
C end source; x-coordinate          6.000
C start source; z-coordinate        1.600
C end source; z-coordinate          6.000
C x-coordinate spacing (km)         0.10
C z-coordinate spacing (km)         0.0175
C step along ray (km)               1.00
C start takeoff angle (deg.)        -155.00
C end takeoff angle (deg.)          -25.000
C takeoff angle incr. (deg)         0.5
C maximum travel time (sec)         11.0
C maximum range in model (km)       6.075
C minimum range in model (km)       -0.075
C maximum depth in model (km)       6.0
C minimum depth in model (km)       0.0
C CDP separation (km)               0.100
C window len. mult. of RECDIST     4.40
C length of x-axis (inches)         30.
C length of z- (or t-) axis         12.

```

```

C ***** COORDINATES AND VELOCITY OF BLOCKS *****
C      4
C  -0.075  0.000
C   6.075  0.000
C   6.075  6.000
C  -0.075  6.00
C   1.5000
C 999
C
C *****
C ***** JOBSTREAM TO RUN SNELL *****
C $JOB JSNELL 1513AC DMP OU=GARBAJ LI=10000 TI=1200 PR=5
C AS 20 = INSNELL
C AS 30 = RANGE
C AS 50 = TTIME
C AS 40 = CRUD
C XSNELL
C VPLOTO7
C $EOJ
C
C ***** JOBSTREAM TO COMPILE AND VULCANIZE *****
C MO EC=ON
C FR ALL
C SAUF77.RPH SNELL
C VU.R XSNELL
C LIB *SAUVPL *LIBERY
C BE
C MO EC=OFF
C *****
C
C written by: Mary M. Rowe - January, 1985
C
C ***** SOURCE PROGRAM SNELL *****
C
COMMON BLOCK(2,10,30),NVERT(30),VEL(30)
COMMON /PARAMS/ M(10,30),B(10,30)
SPECIAL COMMON SURFACE
COMMON /SURFACE/ TTIME(500),RANGE(500),TT(500),R(500)
COMMON /PLOT/ PLOTX(50,360),PLOTZ(50,360),IP(360)
INTEGER VERT,FLAG
REAL M,MRAY,MU
REAL *6 TT,R
LOGICAL RAYTR,TXPLT,RESET
LOGICAL LX1,LZ1,LX2,LZ2
DATA PI,ESPEC /3.141592654,9999.0/
READ (20,1000) RAYTR,TXPLT
READ (20,1010)XSTART,XEND,ZSTART,ZEND,DX,DZ,RAYINC,DTHETA,
&          TMAX,XMAX,XMIN,ZMAX,ZMIN,RECDIST,WMAX,XLEN,
&          ZLEN
1000 FORMAT (T30,L10/,T30,L10)

```

```

1010 FORMAT (17(T30,F10.5/))
      FIRSTZ = ZSTART   ! z-coordinate of the first row of source points
      FIRSTX = XSTART   ! x-coordinate of the first column of source points
C
C Determine range of takeoff angles necessary to bracket window specified
C for tracing rays from uppermost Z.
C
      COTHETA = ATAN(WMAX/(2.*ZSTART))
C
C      COTHETA = 60.*PI/180.           ! constant takeoff angle range
C
      DTHETA = DTHETA*PI/180.         ! takeoff angle increment
      ETHETA = COTHETA-PI/2.+DTHETA*10. ! last takeoff angle
      FTHETA = -ETHETA-PI             ! first takeoff angle
      NRAY = NINT(ABS(ETHETA-FTHETA)/DTHETA)+1
      NOUT = INT(WMAX/RECDIST+1.0001)*2
C
      WRITE (40, '('" DX=" ,F8.4," DZ=" ,F8.4," DTHETA=" ,F8.4," RECDIST=" ,
&          F8.4,)' ) DX,DZ,DTHETA,RECDIST
      NCOL = ABS(XSTART-XEND)/DX+1
      NROW = ABS(ZSTART-ZEND)/DZ+1
C MODEL returns IMAX, the number of uniform velocity blocks in the model.
      CALL MODEL(IMAX)
C SLOPE determines slopes of sides of velocity blocks.
      CALL SLOPE(IMAX)
      RESET = .FALSE.
      NPT = 0           !number of source points in the ray trace section
      I = 0
      LOOP (NCOL)
      XX = XSTART
      I = I+1
      J = 0
      LOOP (NROW)
      ZZ = ZSTART
      NPT = NPT+1
      J = J+1
C FINDPT returns the index of the block the source point is within
30 CALL FINDPT (XSTART,ZSTART,IBSTART,IMAX)
      STHETA = FTHETA !takeoff angle for each traced ray
      THETA = STHETA  ! set initial raypath angle equal to takeoff angle
      ITH = 0        !index for takeoff angle
      NTHETA = 0     !index for rays which reach Z=0
C
C Loop through theta (takeoff angle measured with respect to +x-axis)
C tracing rays to surface from single subsurface point.
C ITH is takeoff angle index, IPLOT is index for points along raypath,
C IP is array containing the number of points along the raypath from each
C takeoff angle THETA (for plotting rays).
C
      LOOP (NRAY)
      XX = XSTART

```

```

ZZ = ZSTART
IB = IBSTART      !index of velocity block ray is currently within
ITH = ITH+1
TIME = 0.0
IPLOT = 1        !index for ray segments in each traced ray
PLOTX(IPLOT, ITH) = XX
PLOTZ(IPLOT, ITH) = ZZ
C
C Set initial values of XX2 and ZZ2.
C
      XX2 = RAYINC*(DCOS(THETA))+XX
      ZZ2 = RAYINC*(DSIN(THETA))+ZZ
C
C Loop through steps along the ray, terminating loop when ray reaches
C the surface.
C
      LOOP
      FLAG = -1
      IPLOT = IPLOT+1
      VERT = 0 !index for vertex of velocity block
C
C Loop through vertices of the block to see if the ray has crossed a
C boundary yet.
C
20      LOOP
      VERT = VERT+1
      IF (VERT.GT.NVERT(IB))
      XX2 = RAYINC*(DCOS(THETA))+XX2
      ZZ2 = RAYINC*(DSIN(THETA))+ZZ2
      VERT = 1
      END IF
      N = VERT+1
      IF (N.GT.NVERT(IB)) N = 1
      XPROD = (BLOCK(1,N,IB)-BLOCK(1,VERT,IB))*(ZZ2-
&          BLOCK(2,VERT,IB))-(BLOCK(2,N,IB)-BLOCK(2,VERT,IB))*
&          (XX2-BLOCK(1,VERT,IB))
C      Exit loop if interface has been crossed.
      EXIT LOOP IF (XPROD.LE.0.0)
      END LOOP
      DIFF = DABS(XX2-XX)
      IF (DIFF.LT.0.000001)
      MRAY = 9999.
      BRAY = 9999.
      ELSE
      MRAY = (ZZ2-ZZ)/(XX2-XX)
      BRAY = ZZ-MRAY*XX
      END IF
      VERT = 0
C
C Loop through sides of the block to see which side is crossed by the
C ray in its direction of travel (not backwards), and locate the point

```



```

C of intersection (x,z) of the ray and the side.
C
      LOOP (10)
        VERT = VERT+1
C
C If starting point of a set of rays lies along the side of a block,
C the point is offset.
C
      IF (VERT.GT.NVERT(IB))
        IF (DABS(XX-XSTART).LE..00001.AND.
          & DABS(ZZ-ZSTART).LE..00001)
          ZSTART = ZZ+.001
          XSTART = XX-.001
          RESET = .TRUE.
          GO TO 30
C
C If the end point of a ray lies on a vertex of a block, the angle of
C propagation is shifted slightly and the ray is re-traced.
C
      ELSE
        THETA = THETA-.0001
        XX2 = XX
        ZZ2 = ZZ
        GO TO 20
      END IF
D      WRITE (40,(' ** ERROR - NO POINT OF INTERSECTION ** '))
D      WRITE (40,('5X," for THETA = ",F8.4," XSTART",F10.4," ZSTART",
D      &F10.4')) THETA,XSTART,ZSTART
D      WRITE (40,('" XX=",F10.4," ZZ=",F10.4," X=",F10.4," Z=",
D      &F10.4')) XX,ZZ,X,Z
D      WRITE (40,('" IB",I4," NVERT(IB)",I4')) IB,NVERT(IB)
D      WRITE (40,('" MRAY=",E10.3E3," BRAY=",E10.3E3')) MRAY,BRAY
D      END IF
C
      IF (DABS(M(VERT,IB)-MRAY).LT.0.000001) GO TO 10
      IF (DABS(MRAY).LT.0.000001.AND.DABS(M(VERT,IB)).EQ.9999.0)
        Z = ZZ
        X = BLOCK(1,VERT,IB)
      ELSE IF (MRAY.EQ.9999.0)
        X = XX
        Z = M(VERT,IB)*X+B(VERT,IB)
      ELSE IF (DABS(MRAY).LT.0.000001)
        Z = ZZ
        X = (Z-B(VERT,IB))/M(VERT,IB)
      ELSE IF (M(VERT,IB).EQ.9999.0)
        X = BLOCK(1,VERT,IB)
        Z = MRAY*X+BRAY
      ELSE
        X = (BRAY-B(VERT,IB))/(M(VERT,IB)-MRAY)
        Z = MRAY*X+BRAY
      END IF

```

```

      N = VERT+1
      IF (N.GT.NVERT(IB)) N = 1
C does ray cross a side of the block?
      DELZ = DABS(BLOCK(2,N,IB)-BLOCK(2,VERT,IB))+.00001
      DELX = DABS(BLOCK(1,N,IB)-BLOCK(1,VERT,IB))+.00001
      IF (DELZ.GE.DABS(BLOCK(2,N,IB)-Z))
        IF (DELZ.GE.DABS(BLOCK(2,VERT,IB)-Z))
          IF (DELX.GE.DABS(BLOCK(1,N,IB)-X))
            IF (DELX.GE.DABS(BLOCK(1,VERT,IB)-X))
C          does ray cross side in direction of travel?
            IF (((X-XX)*(XX2-XX)+(Z-ZZ)*(ZZ2-ZZ)).GT.0.00001)
              &          FLAG = +1
            END IF
            END IF
            END IF
            END IF
            EXIT LOOP IF (FLAG.EQ.+1)
10          CONTINUE
          END LOOP
C
C Write (x,z) to plot file.
C
      PLOTX(IPLOT,ITH) =X
      PLOTZ(IPLOT,ITH) = Z
C
C Calculate travel time to point at which ray intersects the side
C (time will be added to for each increment along the raypath).
C
      TIME = TIME+(((XX-X)**2+(ZZ-Z)**2)**0.5)/VEL(IB)
C Exit loop for incrementing along ray if ray has reached bounds of section.
      EXIT LOOP IF (TIME.GT.TMAX)
      EXIT LOOP IF (Z.LE.ZMIN+.000001)
      EXIT LOOP IF (X.LE.XMIN+.000001)
      EXIT LOOP IF (X.GE.XMAX-.000001)
      EXIT LOOP IF (Z.GE.ZMAX-.000001)
      IBLOCK = 0
      IFLAG = -1
C
C Find which block ray is now entering and its corresponding velocity.
C Loop through the blocks.
C
      LOOP (IMAX)
        IBLOCK = IBLOCK+1
        IF (IBLOCK.EQ.IB) IBLOCK = IBLOCK+1
        ISIDE = 0
        IFLAG = -1
C
C Loop through sides of each block to find the block haveing a side in
C common with the block just exited.
C
      LOOP (NVERT(IBLOCK))

```

```

        LX1 = .FALSE.
        LZ1 = .FALSE.
        LX2 = .FALSE.
        LZ2 = .FALSE.
        ISIDE = ISIDE+1
C
    IF (ISIDE.GT.10)
    WRITE (40, '('" NVERT(IBLOCK) TOO BIG "')')
    STOP
    END IF
C
        NSIDE = ISIDE+1
        IF (NSIDE.GT.NVERT(IBLOCK)) NSIDE = 1
        IF (DABS(BLOCK(1, ISIDE, IBLOCK)-BLOCK(1, N, IB)).LE.
&          .00001) LX1 = .TRUE.
        IF (DABS(BLOCK(2, ISIDE, IBLOCK)-BLOCK(2, N, IB)).LE.
&          .00001) LZ1 = .TRUE.
        IF (DABS(BLOCK(1, NSIDE, IBLOCK)-BLOCK(1, VERT, IB)).LE.
&          .00001) LX2 = .TRUE.
        IF (DABS(BLOCK(2, NSIDE, IBLOCK)-BLOCK(2, VERT, IB)).LE.
&          .00001) LZ2 = .TRUE.
        IF (LX1.AND.LZ1)
            IF (LX2.AND.LZ2)
                LASTIB = IB
                IB = IBLOCK
                IFLAG = +1
            END IF
        END IF
        EXIT LOOP IF (IFLAG.EQ.+1)
    END LOOP
    EXIT LOOP IF (IFLAG.EQ.+1)
    END LOOP
    MU = DATAN(M(ISIDE, IB)) ! MU is the angle of slope.
C    Calculate angle of incidence.
C    Rotate through angle -MU and use SIN function to calculate PHI.
    UM = -1.*MU
C    rotate side to parallel to horizontal axis
    XXR = XX*DCOS(UM)-ZZ*DSIN(UM)
    ZZR = XX*DSIN(UM)+ZZ*DCOS(UM)
    XR = X*DCOS(UM)-Z*DSIN(UM)
    ZR = X*DSIN(UM)+Z*DCOS(UM)
C    re-orient ray so origin is at (xx,zz)
    DXXR = XR-XXR
    DZZR = ZR-ZZR
C    calculate PHI1
    PHI1 = DATAN(DXXR/DZZR)
    SINPHI1 = DSIN(PHI1)
C    calculate PHI2
    SINPHI2 = SINPHI1/VEL(LASTIB)*VEL(IB)
    EXIT LOOP IF(DABS(SINPHI2).GT..99999)
    PHI2 = DASIN(SINPHI2)

```

```

C      calculate next increment along raypath
          DXX2R = RAYINC*SINPHI2
          DZZ2R = RAYINC*DCOS(PHI2)
C      calculate next endpoint along raypath
          DZZ2R = SIGN(DZZ2R,DZZR)
          DXX2R = SIGN(DXX2R,DXXR)
          XX2R = DXX2R+XR
          ZZ2R = DZZ2R+ZR
C      rotate back through angle +MU to original orientation
          XX2 = XX2R*DCOS(MU)-ZZ2R*DSIN(MU)
          ZZ2 = XX2R*DSIN(MU)+ZZ2R*DCOS(MU)
C      calculate angle THETA (using origin at (x,z))
          THETA = DATAN2(ZZ2-Z,XX2-X)
C      reset PHI1
          PHI1 = PHI2
C
C      Reset starting point of ray.
C
          XX = X
          ZZ = Z
C
C      Now trace ray through new block.
C
          END LOOP
          IF (Z.LE.ZMIN+.000001)
            NTHETA = NTHETA+1
            RANGE(NTHETA) = X
            TTIME(NTHETA) =TIME
C
          IF (TXPLT)
            WRITE (42, ) TTIME(NTHETA)
            WRITE (43, ) RANGE(NTHETA)
            END IF
C
          END IF
          IP(ITH)= IPLOT
          STHETA = STHETA+DTHETA
          EXIT LOOP IF (STHETA.GT.ETHETA)
          THETA = STHETA
          END LOOP      ! through takeoff angles
C
D      WRITE (40, ) " NTHETA =",NTHETA
C
          CALL CAUSTIC(NTHETA)
          CALL INTERP(RECDIST,NTHETA,XSTART,WMAX)
          IF (TXPLT) CALL XTPLT(NTHETA,XMAX,XMIN,TMAX,XSTART,ZSTART,
&                               XLEN,ZLEN)
C
C      Write travel time and range information to external files 50 and
C      30, respectively.
C

```

```

BUFFER OUT(30,R(1),B,NOUT,ISTAT30,N30)
CALL STATUS(30)
BUFFER OUT(50,TT(1),B,NOUT,ISTAT50,N50)
CALL STATUS(50)
ZSTART = ZSTART+DZ
IF (RESET)
  ZSTART = ZSTART-.001
  XSTART = XSTART+.001
  RESET = .FALSE.
END IF
END LOOP
ZSTART = FIRSTZ
XSTART = XSTART+DX
END LOOP
IF (RAYTR) CALL TRACE( ITH,XMAX,XMIN,ZMAX,ZMIN,XLEN,ZLEN)
STOP
END

```

```

C
C *****
C
C ***** SUBROUTINE SLOPE. *****
C
C Subroutine calculates slope and z-intercept of the sides of each block.
C

```

```

SUBROUTINE SLOPE(IMAX)
COMMON BLOCK(2,10,30),NVERT(30),VEL(30)
COMMON /PARAMS/ M(10,30),B(10,30)
REAL M
I = 0
LOOP(IMAX)
  I = I+1
  J = 0
  LOOP
    J = J+1
    JJ = J+1
    IF (JJ.GT.NVERT(I)) JJ = 1
    DIFF= DABS(BLOCK(1,J,I)-BLOCK(1,JJ,I))
    IF (DIFF.LT.0.000001)
      M(J,I) = 9999.
      B(J,I) = 9999.
    ELSE
      M(J,I) = (BLOCK(2,J,I)-BLOCK(2,JJ,I))/(BLOCK(1,J,I)-
& BLOCK(1,JJ,I))
      B(J,I) = BLOCK(2,J,I)-M(J,I)*BLOCK(1,J,I)
    END IF
    EXIT LOOP IF (JJ.EQ.1)
  END LOOP
END LOOP
RETURN
END

```

```

C

```

```

C *****
C
C ***** SUBROUTINE MODEL *****
C
C Subroutine inputs coordinates of block vertices; maximum possible
C number of blocks is 29, maximum number of vertices is 10.
C
  SUBROUTINE MODEL(IMAX)
  COMMON BLOCK(2,10,30),NVERT(30),VEL(30)
  I = 0
  LOOP (30)
    I = I+1
    J = 0
    READ (20, ) NVERT(I)
    EXIT LOOP IF (NVERT(I).EQ.999)
    LOOP
      J = J+1
      READ (20, ) (BLOCK(K,J,I),K =1,2)
      EXIT LOOP IF (J.GE.NVERT(I))
    END LOOP
    READ (20, ) VEL(I)
  END LOOP
  IMAX = I-1
  RETURN
  END
C
C *****
C
C ***** SUBROUTINE FINDPT *****
C
C Subroutine finds block that the point lies within.
C
  SUBROUTINE FINDPT(XX,ZZ,IB,IMAX)
  COMMON BLOCK(2,10,30),NVERT(30),VEL(30)
  INTEGER VERT
  I = 0
  LOOP
    I = I+1
C
    IF (I.GT.IMAX)
      WRITE (40, '('" ** ERROR ** POINT NOT WITHIN SECTION"')')
      STOP
    END IF
C
    VERT = 0
    LOOP
      VERT = VERT+1
      X1 = BLOCK(1,VERT,I)
      Z1 = BLOCK(2,VERT,I)
      N = VERT+1
      IF (N.GT.NVERT(I)) N = 1

```

```

X2 = BLOCK(1,N,I)
Z2 = BLOCK(2,N,I)
AX = X2-X1
AZ = Z2-Z1
BX = XX-X1
BZ = ZZ-Z1
C
C Take cross product to see whether point lies within the block.
C
CROSS = AX*BZ-BX*AZ
EXIT LOOP IF (CROSS.LT.0.)
EXIT LOOP IF (VERT.GE.NVERT(I))
END LOOP
EXIT LOOP IF (CROSS.GE.0.)
END LOOP
IB = I
RETURN
END
C
C *****
C
C ***** SUBROUTINE TRACE *****
C
C SUBROUTINE TRACE( ITH,XMAX,XMIN,ZMAX,ZMIN,XLEN,ZLEN)
C
C Subroutine traces rays from a single source point. If a ray trace is
C desired and rays are traced from more than one source point by SNELL
C then only the last ray trace will be plotted.
C
C COMMON BLOCK(2,10,30),NVERT(30),VEL(30)
C COMMON /PLOT/ PLOTX(50,360),PLOTZ(50,360),IP(360)
C DIMENSION LABLX(4),LABLZ(4),ITITL(7)
C DIMENSION MODL(150)
C CHARACTER*150 MLBL
C CHARACTER*24 LBL
C EQUIVALENCE (MODL(1),MLBL)
C CALL VP07MP("MODEL",1200,"UNIT",1.00,"XMAX",16.,"MSGLVL",3,
C & "LYNES",500,"END")
C CALL PLOTS(0,0,0)
C
C Input model name from input parameter file, for plot title.
C
C
C Plan axes with tic marks every 5 km on x-axis, every 2 km on z-axis.
C (Draws tic marks only, no axes drawn.)
C
C CALL NEWPEN(1)
C IDELX = INT((XMAX-XMIN)/5.+0.0001)+1
C DX = XLEN/(XMAX-XMIN)*5.
C First tic mark is at X=0.0.
C X = .75-XMIN*DX/5.

```

```

Z = .75
LOOP (IDELX)
  CALL PLOT(X,Z,3)
  Z = .625
  CALL PLOT(X,Z,2)
  X = X+DX
  Z = .75
END LOOP
IDELZ = INT((ZMAX-ZMIN)/2.)+1
DZ = ZLEN/(ZMAX-ZMIN)*2.
C First tic mark at Z=0.0.
X = .75
Z = .75-ZMIN*DZ/2.
LOOP (IDELZ)
  CALL PLOT(X,Z,3)
  X = .625
  CALL PLOT(X,Z,2)
  Z = Z+DZ
  X = .75
END LOOP
C label axes
CALL NEWPEN(2)
WRITE (LBL, '(' RANGE (KM) ')')
READ (LBL, '(4A3)') LABLX
WRITE (LBL, '(' DEPTH (KM) ')')
READ (LBL, '(4A3)') LABLZ
XTITL = XLEN/2.+1.75
ZTITL = ZLEN/2.-0.25
CALL SYMBOL(XTITL,0.30,0.200,LABLX,180.,12)
CALL SYMBOL(0.25,ZTITL,0.200,LABLZ,90.0,12)
Z = .60
X = .875-XMIN*DX/5.
PT = 0.0
LOOP (IDELX)
  CALL NUMBER(X,Z,.125,PT,180.,1)
  X = X+DX
  PT= PT+5.
END LOOP
Z = .79
X = 0.60
PT = 0.0
LOOP (IDELZ)
  CALL NUMBER(X,Z,0.125,PT,180.,1)
  Z =Z+DZ
  PT = PT+2.
END LOOP
C
C draw velocity model
C
DDX = DX/5.
DDZ = DZ/2.

```



```

XNEW = .75-XMIN*DDX
ZNEW = .75-ZMIN*DDZ
C   Re-position origin at (0.0,0.0) on plot.
CALL PLOT(XNEW,ZNEW,-3)
J = 0
C Loop through blocks.
LOOP
  CALL NEWPEN(3)
  J = J+1
  EXIT LOOP IF (NVERT(J).EQ.999)
  I = 1
  X = BLOCK(1,I,J)*DDX
  Z = BLOCK(2,I,J)*DDZ
  CALL PLOT(X,Z,3)
  IVERT = NVERT(J)-1
  LOOP ( IVERT)
    I = I+1
    X = BLOCK(1,I,J)*DDX
    Z = BLOCK(2,I,J)*DDZ
    CALL PLOT(X,Z,2)
  END LOOP
  X = BLOCK(1,1,J)*DDX
  Z = BLOCK(2,1,J)*DDZ
  CALL PLOT(X,Z,2)
C Write velocity at center of each block.
XAV = BLOCK(1,1,J)
ZAV = BLOCK(2,1,J)
FOR I=2,NVERT(J)
  XAV = XAV+BLOCK(1,I,J)
  ZAV = ZAV+BLOCK(2,I,J)
END FOR
X = XAV/NVERT(J)*DDX+.150
Z = ZAV/NVERT(J)*DDZ+.055
CALL NEWPEN(2)
CALL NUMBER(X,Z,0.125,VEL(J),180.,1)
END LOOP
C
C draw rays (superimposed on velocity model)
C
CALL NEWPEN(2)
I = 0
LOOP ( ITH)
  I = I+1
  K = 1
  X = PLOTX(K,I)*DDX
  Z = PLOTZ(K,I)*DDZ
  CALL PLOT(X,Z,3)
  IP(I) = IP(I)-1
  LOOP ( IP(I))
    K = K+1
    X = PLOTX(K,I)*DDX

```

```

      Z = PLOTZ(K,I)*DDZ
      CALL PLOT(X,Z,2)
      END LOOP
END LOOP
WRITE (LBL,('SOURCE AT ',F5.2," ",F5.2)) PLOTX(1,1),PLOTZ(1,1)
READ (LBL,'(7A3)') ITITL
ZTITL = ZLEN+.25
CALL SYMBOL(XLEN,ZTITL,.200,ITITL,180.,21)
READ (20,'(50A3)') MODL
XMODEL = XLEN/2.
CALL SYMBOL(XMODEL,ZTITL,.200,MODL,180.,LEN(MLBL))
WRITE (LBL,('VELOCITIES IN KM/SEC '))
READ (LBL,'(7A3)') ITITL
ZTITL = ZLEN+.50
CALL SYMBOL(XLEN,ZTITL,.200,ITITL,180.,21)
CALL PLOT(0.,0.,+999)
RETURN
END
C
C *****
C
C ***** SUBROUTINE INTERP *****
C
      SUBROUTINE INTERP(RECDIST,NTHETA,XSTART,WMAX)
C
C Subroutine interpolates travel times to range coordinates corresponding
C to receiver locations.
C
      SPECIAL COMMON SURFACE
      COMMON /SURFACE/ TTIME(500),RANGE(500),TT(500),R(500)
      REAL*6 TT,R
      LOGICAL FLAG
C
C      WRITE (40, )NTHETA,XSTART
C
C Set starting value of CDP to be interpolated, may not be within
C section.
      FIRSTX = NINT(XSTART/RECDIST)*RECDIST-WMAX/2.
      XPRIME = FIRSTX
      N = 0
C
C      WRITE (40,(' XPRIME=',F10.4)) XPRIME
C
      WHILE (XPRIME.LT.RANGE(1))
          N = N+1
          R(N) = XPRIME
          TT(N) = 0.
          XPRIME = XPRIME+RECDIST
C
C      WRITE (40,(' XPRIME=',F10.4)) XPRIME
C

```

```

      END WHILE
C
C Start interpolating travel times for receivers which are within
C window.
C
      K = 1
      KK = 2
C
C Loop through receiver locations within window.
C
      LOOP
      FLAG = .FALSE.
C
C Loop through takeoff angles to find ranges for interpolation.
C
      LOOP
      IF (RANGE(K).LE.XPRIME.AND.RANGE(KK).GE.XPRIME)
C
D      WRITE (40, )N,K,KK
C
C travel time intepolated for receiver location between points K and KK.
      N = N+1
      TT(N) = TTIME(K)+(XPRIME-RANGE(K))/(RANGE(KK)-RANGE(K))*
&      (TTIME(KK)-TTIME(K))
      FLAG = .TRUE.
      R(N) = XPRIME
      END IF
      EXIT LOOP IF (FLAG)
C
      IF (KK.GE.NTHETA)
      WRITE (40,('" ** ERROR - range of traced rays too small for
&window "'))
      WRITE (40,('" XPRIME= ",E12.4E3," KK=",I4," RANGE=",
&      E10.3E3')) XPRIME,KK,RANGE(NTHETA)
      WRITE (40,('" XSTART=",E10.3E3')) XSTART
C
      L=1
      LOOP (NTHETA)
      WRITE (40, ) RANGE(L)
      L = L+1
      END LOOP
C
      STOP
      END IF
C
      K = KK
      KK = K+1
      END LOOP
      XPRIME = XPRIME+RECDIST
C
D      WRITE (40,('" XPRIME=",F10.4')) XPRIME

```

```

C
      EXIT LOOP IF (XPRIME.GT.(FIRSTX+WMAX+.00001))
C
C      WRITE (40, ) "RANGE(NTHETA) TOO SMALL"
C
      EXIT LOOP IF (XPRIME.GT.RANGE(NTHETA))
C
C      WRITE (40, ) "DISREGARD ABOVE"
C
      END LOOP
      WHILE (XPRIME.LE.(FIRSTX+WMAX))
        N = N+1
        TT(N) = 0.
        R(N) = XPRIME
        XPRIME = XPRIME+RECDIST
      END WHILE
      RETURN
      END
C
C *****
C *****
C
      SUBROUTINE XTPLT(NTHETA,XMAX,XMIN,TMAX,XSTART,ZSTART,XLEN,ZLEN)
C
C Subroutine draws an X - T plot for each point from which rays were traced.
C
      SPECIAL COMMON SURFACE
      COMMON /SURFACE/ TTIME(500),RANGE(500),TT(500),R(500)
      DIMENSION LABLX(4),LABLT(4),ITITL(7)
      REAL *6 TT,R
      CHARACTER*24 LBL
      CALL VP07MP("MODEL",1200,"UNIT",1.00,"XMAX",16.,"MSGLVL",3,
&               "LYNES",1000,"END")
      CALL PLOTS(0,0,0)
C
C Draw axes, tic marks every 5 km (horizontal) and every 2 secs(vertical).
C
      CALL NEWPEN(3)
      CALL PLOT(1.25,1.250,3)
      CALL PLOT(XLEN,1.25,2)
      IDELX = INT((XMAX-XMIN)/5.)+1
      DX = XLEN/(XMAX-XMIN)*5.
      CALL NEWPEN(1)
      X = 1.25
      T = 1.25
      LOOP (IDELX)
        CALL PLOT(X,T,3)
        T = 1.125
        CALL PLOT(X,T,2)
        X = X+DX

```

```

      T = 1.25
      END LOOP
      CALL NEWPEN(3)
      CALL PLOT(1.25,1.25,3)
      CALL PLOT(1.25,ZLEN,2)
      CALL NEWPEN(1)
      IDELT = INT(TMAX/2.)+1
      DT = ZLEN/TMAX*2.
      X = 1.25
      T = 1.25
      LOOP ( IDELX)
        CALL PLOT(X,T,3)
        X = 1.125
        CALL PLOT(X,T,2)
        T = T+DT
        X = 1.25
      END LOOP
C   label axes
      CALL NEWPEN(2)
      WRITE (LBL, '(' RANGE (KM) ')')
      READ (LBL, '(4A3)') LABLX
      WRITE (LBL, '(' TIME (SEC) ')')
      READ(LBL, '(4A3)') LABLT
      TTITL = ZLEN-.5
      CALL SYMBOL(3.25,0.1,0.35,LABLX,0.0,12)
      CALL SYMBOL(0.1,TTITL,0.35,LABLT,270.0,12)
      T = .75
      X = 1.25
      PT = 0.0
      LOOP ( IDELX)
        CALL NUMBER(X,T,0.2,PT,0.,1)
        X = X+DX
        PT = PT+5.
      END LOOP
      T = 1.25
      X = .625
      PT = 0.0
      LOOP ( IDELT)
        CALL NUMBER(X,T,0.2,PT,0.,1)
        T = T+DT
        PT = PT+2.
      END LOOP
C
C   rays are traced
C
      CALL PLOT(1.25,1.25,-3)
      DDX = DX/5.
      DDT = DT/2.
      CALL NEWPEN(2)
      I = 1
      X = RANGE(I)*DDX

```

```

T = TTIME(I)*DDT
CALL PLOT(X,T,3)
NTHETA = NTHETA - 1
LOOP(NTHETA)
  I = I+1
  X = RANGE(I)*DDX
  T = TTIME(I)*DDT
  CALL PLOT(X,T,2)
END LOOP
WRITE (LBL, '(" SOURCE ",F5.2," ",F5.2)') XSTART,ZSTART
READ (LBL, '(7A3)') ITITL
ZTITL = ZLEN-.3
CALL NEWPEN(4)
CALL SYMBOL(0.,ZTITL,0.3,ITITL,0.,20)
CALL PLOT(0.,0.,+999)
RETURN
END

C
C
C *****
C
C
C
C      SUBROUTINE CAUSTIC(NTHETA)
C
C Subroutine smooths the travel time curve in the presence of caustics
C by including only ranges with the shortest travel times, omitting
C other branches of the caustic.
C
C      SPECIAL COMMON SURFACE
C      COMMON /SURFACE/ TTIME(500),RANGE(500),TT(500),R(500)
C      REAL*6 TT,R
C      REAL M1,M2
C
C Determine endpoints of branches and rays with shortest travel times.
C
C      ISTART = 1
C      FOR I=ISTART,NTHETA-1
C          II = I+1
C          EXIT FOR IF(RANGE(II).LT.RANGE(I))
C      END FOR
C
C      WRITE (40, ) "LOOP 1 EXECUTED"
C
C      IF (I.GE.NTHETA-1) RETURN
C      IIMAX = I
C
C      WRITE (40, ) IIMAX
C
C      FOR I=IIMAX+1,NTHETA-1
C          II = I+1
C          EXIT FOR IF (RANGE(II).GT.RANGE(I))

```

```
END FOR
C
D WRITE (40, ) "LOOP 2 EXECUTED"
C
  IMIN = I
C
D WRITE (40, ) IMIN
C
  IF ( IIMAX.LE.1)
    J=0
    FOR K=IMIN,NTHETA
      J = J+1
      RANGE(J) = RANGE(K)
      TTIME(J) = TTIME(K)
    END FOR
    NTHETA = J
    FOR JJ=J+1,NTHETA
      RANGE(JJ) = 0.
      TTIME(JJ) = 0.
    END FOR
    ISTART = 1
    GO TO 90
  END IF
  FOR I=IMIN+1,NTHETA
    EXIT FOR IF(RANGE(I).GT.RANGE(IIMAX))
  END FOR
C
D WRITE (40, ) "LOOP 3 EXECUTED"
C
  MAX = I
C
D WRITE (40, ) MAX
C
  IF ( RANGE(MAX).LE. RANGE(IIMAX))
    NTHETA = IIMAX
    RETURN
  END IF
C
  FOR I=IIMAX,1,-1
    EXIT FOR IF(RANGE(I).LT. RANGE(IMIN))
  END FOR
C
D WRITE(40, ) "LOOP 4 EXECUTED"
C
  MIN = I
C
D WRITE (40, ) MIN
C
  IF ( RANGE(MIN).GT. RANGE(IMIN))
```

```

      J=0
      FOR K = IMIN,NTHETA
        J = J+1
        RANGE(J) = RANGE(K)
        TTIME(J) = TTIME(K)
      END FOR
      FOR JJ=J+1,NTHETA
        RANGE(JJ) = 0.
        TTIME(JJ) = 0.
      END FOR
      NTHETA = J
      ISTART = 1
      GO TO 90
    END IF
  C
  C
  C Find intersection of branches 1 and 3.
  C
      M1 = (TTIME(IIMAX)-TTIME(MIN))/(RANGE(IIMAX)-RANGE(MIN))
      B1 = TTIME(IIMAX)-M1*RANGE(IIMAX)
      M2 = (TTIME(MAX)-TTIME(IMIN))/(RANGE(MAX)-RANGE(IMIN))
      B2 = TTIME(IMIN)-M2*RANGE(IMIN)
      XMID =(B1-B2)/(M2-M1)
      FOR J=MIN,IIMAX
        EXIT FOR IF (RANGE(J).GE.XMID)
      END FOR
      MID1 = J-1
      FOR J=IMIN,NTHETA
        EXIT FOR IF(RANGE(J).GT.XMID)
      END FOR
      MID2 = J
  C MID1 and MID2 are indices for ranges with smallest travel times
  C bracketing point of intersection of branches 1 and 3.
  C
  C Re-index set of ranges, including only branch points with shortest
  C travel times.
  C
      NDIFF = MID2-MID1
      L = MID2
      FOR K=MID1+1,NTHETA-NDIFF+1
        RANGE(K) = RANGE(L)
        TTIME(K) = TTIME(L)
        EXIT FOR IF(L.GE.NTHETA)
        L = L+1
      END FOR
      NTHETA = K
      ISTART = MID1+1
  C
  D WRITE (40, "(' NTHETA=',I4)") NTHETA
  C
  90 IF(MAX.LT.NTHETA) GO TO 100

```


RETURN
END

Migration:

NAME MIGRAT

C 02/05/860075C Version 02:01; Corrected special common ARRAYS,
 C previously it did not include array NORM. This may correct problem we
 C are having with the coherence calculation.

C

C

 C Program migrates unstacked seismic data using Kirchhoff pre-stack
 C migration. Scattered energy is summed into the scatterer source
 C by matching travel times from the source to the shot and receiver
 C locations found by tracing rays.
 C -----

C external files required:

C 4 - tape

C 20 - input parameters (INMIG)

C 25 - input, travel times (TIME)

C 40 - output file containing program error messages

C 50 - input, one trace from one shot (note that the external source
 C file is variable)

C 60 - output file, migrated section

C 70 - output file, coherence

C 65 - temporary output file, for re-indexing

C 90 - input file containing data to be migrated

C

C

C LFN 50 is assigned by the program, no external ASSIGN is needed. BUT
 C files TRACE1, TRACE2, TRACE3, ..., TRACE_n must be generated for n
 C receivers before the program is run. The program uses these as
 C temporary storage for all traces from one shot while that shot is
 C being migrated.

C

C When running data advance file to first data block, skipping header
 C blocks. Program assumes trace header is not a separate block.

C Trace lengths will probably vary.

C Program assumes 4 msec sample frequency.

C

C Data and output (migrated section and coherence) are type integer
 C (ie. one word each).

C

C Be sure the first shot to be processed is within the specified
 C migration aperture.

C Coverage of the first few CDPs will not be full fold, if full fold
 C coverage is desired for all CDPs then the limits of the migrated
 C section should be enlarged to provide the coverage.

C If the last shot is at the margin of the section there will be full-
 C fold coverage of all points near that margin.

C
 C The aperture over which the actual migration is performed increases
 C with depth, the same depth-variable aperture set up by the ray C tracing.
 C Due to the limited memort of the Harris 800 computer, only a portion C of
 C the migrated section can be held in memory. The width of the section
 C in memory is equal to the width of the migration aperture at the
 C greatest depth in the section. When all traces within the aperture
 C have been migrated, the aperture slides over to the next shot and C those
 C traces are migrated.

C
 C ***** SAMPLE INPUT FILE - INMIG *****
 C
 C 31 number of x-coordinates
 C 51 number of z-coordinates
 C 11 number of shots
 C 0 number of words in trace header
 C 8 number of receivers
 C 11 CDP of first shot
 C 6.0 maximum range of migrated section
 C 0.0 minimum range of migrated section
 C 0.00 delay time before first data sample comes in
 C 0.20 distance between x-coordinates (km) (= CDP separation)
 C 0.40 shot separation (km)
 C 11.0 maximum expected trace length (sec) (take header into account)
 C 4.0 aperture widths at greatest depth (km) (from file WINDOWS)
 C .0080 sample interval (sec) of data trace

C*****

C
 C program arrays:
 C TIME - array containing travel times for each ray reaching the
 C surface from each starting point, dimensioned according to (from
 C left to right) number of rays reaching surface, number of rows,
 C and number of columns
 C MI - array of responses at subsurface points making up the
 C migrated section.
 C CO - array of subsurface responses for generating coherence
 C plots
 C Dimension of MI and CO depends on ray trace window length
 C and depth of section.
 C TRACE - array containing data from one trace from one shot (ie.
 C one receiver location)
 C TMAX - length of each trace (sec)
 C input parameters:
 C NX - number of x-coordinates of points in migrated section
 C NZ - number of z-coordinates " " " " "
 C NSHOT - number of shots to be processed
 C NHEADER - number of words in trace header in one input trace
 C NREC - number of receiver groups in array
 C DELAY - delay time before first signal is received (check to

```

C          see if same for all shots used)
C XDIST - distance between x-coordinates of subsurface points (= CDP
C          separation for stacked data)
C SSEP - shot separation
C SAMP - data sample rate, in seconds
C TLEN - maximum predicted length of data traces, in seconds
C WT - weight functions for each point within the aperture
C WINDO - # x-coordinates of subsurface points within the aperture,
C          symmetric around point, equal to ray trace window.
C SAMP - data sample interval in seconds
C FSHOT - CDP of first shot
C XMAX - maximum range of migrated section
C XMIN - minimum range of migrated section
C
C ***** JOBSTREAM TO COMPILE AND VULCANIZE MIGRAT *****
C
C MO EC=ON
C FR ALL
C SAUF77.XR MIGRAT
C VU.R XMIG-R
C ALLOCATE,S+10
C LIB *LIBERY
C BE
C MO EC=OFF
C
C ***** JOBSTREAM TO RUN MIGRAT *****
C
C $JOB JMIGRAT 1513AC DMP OU=REFUSE LI=10000 TI=1200 PR=5
C AS 90=PULSE
C AS 20=INMIG
C AS 25=WINDOVS
C AS 25=TTIME
C AS 40=CRUD
C AS 60=MISYNTH
C AS 70=COSYNTH
C XMIG-R
C $EQJ
C
C *****
C
C This version designed to use shot separation less than or equal to
C receiver location and lateral distance between depth points in
C migrated section less than or equal to shot separation (to give
C good lateral resolution).
C
C ***** SOURCE PROGRAM MIGRAT *****
C
C          INTEGER FLAG
C          INTEGER WLEN, FSHOT, TSHOT, TREC, DSHOT, DREC

```

```

INTEGER STAT1, STAT2, STAT3, STAT4, STAT6, STAT7, OUT1, OUTWRD1, OUT2,
&      OUTWRD2
INTEGER TRACE, HEADER, CO
CHARACTER CHR(4)*1, TRACENAME(24)*16
COMMON /PARAMS/ TMAX(24), WT(30, 300)
COMMON /TAPE/ HEADER(2), TRACE(2752)
SPECIAL COMMON TIMES
COMMON /TIMES/ TIME(50, 300, 30)
SPECIAL COMMON ARRAYS
COMMON /ARRAYS/ MI(300, 30), CO(300, 30), NORM(300)
PARAMETER (PI=3.141592654)
DATA TRACENAME/"1900AC TRACE1 ", "1900AC TRACE2 ",
&"1900AC TRACE3 ", "1900AC TRACE4 ", "1900AC TRACE5 ",
&"1900AC TRACE6 ", "1900AC TRACE7 ", "1900AC TRACE8 ",
&"1900AC TRACE9 ", "1900AC TRACE10 ", "1900AC TRACE11 ",
&"1900AC TRACE12 ", "1900AC TRACE13 ", "1900AC TRACE14 ",
&"1900AC TRACE15 ", "1900AC TRACE16 ", "1900AC TRACE17 ",
&"1900AC TRACE18 ", "1900AC TRACE19 ", "1900AC TRACE20 ",
&"1900AC TRACE21 ", "1900AC TRACE22 ", "1900AC TRACE23 ",
&"1900AC TRACE24 "/
READ(20, '(5(I8/), I8)') NX, NZ, NSHOT, NHEADER, NREC, FSHOT
READ(20, '(14(F8.3/), F8.4)') XMAX, XMIN, XDIST, ZMAX, ZMIN, ZDIST, DELAY,
&      SSEP, RSEP, TLEN, TTWMAX, WMAX, WMIN, SAMP, STREAM
C
WRITE (40, ) "XMAX=", XMAX, "XMIN=", XMIN, "DELAY=", DELAY, "XDIST=",
&XDIST
WRITE (40, ) "SSEP=", SSEP, "RSEP=", RSEP, "TLEN=", TLEN
WRITE (40, ) "TTWMAX=", TTWMAX, "WMAX=", WMAX, "WMIN=", WMIN
WRITE (40, ) "SAMP=", SAMP, "STREAM=", STREAM
C
NCO = NZ*2
NARR = NINT(TTWMAX/SSEP)+1 ! number of shots within aperture
NARR2 = (NARR+1)/2
NDEAD = NINT(STREAM/SSEP)-NINT((RSEP/SSEP)*(NREC-1))
C Number of columns of depth points within migration aperture - WLEN.
WLEN = NINT(TTWMAX/XDIST)+1-NDEAD*NINT(SSEP/XDIST)
NTIME = INT(TLEN/SAMP) ! maximum number of data points per trace
DREC = NINT(RSEP/SSEP) ! ratio of receiver sep. to shot sep.
NTRAVEL = (NINT(TTWMAX/SSEP)+1)*2
C
C Calculate migration window lengths and weights for each window length.
C Store in array WT.
C
C
WRITE (40, ) "CALCULATED WINDOW LENGTHS"
C
WMAX = WMAX+2.*SSEP ! want last interval = maximum window
DELW = WMAX-WMIN ! range of aperture user specified
DELZ = ZMAX-ZMIN ! range of migrated depth samples

```

```

FOR IZ=1,NZ
  Z = ZMIN+(IZ-1)*ZDIST
  DZ = Z-ZMIN      ! depth of point currently under consideration
  W = INT((WMIN+DELW/DELZ*DZ)/SSEP+.001)*SSEP !window width at C Window
width at depth z
  IF (INT(W/(SSEP*2.+.001)*SSEP*2.0.LT.W) W = W-SSEP
C   Accomodate window specified window lengths greater than aperture
C   of traced rays.
  IF (W.GT.TTWMAX)
    WRITE (40, ) "*** ERROR ** DESIRED WINDOW TOO BIG - DEFAULT TO
& TTWMAX"
    W = TTWMAX
  END IF
C   WRITE (40, ) W
C
  NARRW = NINT(W/SSEP)+1
C   Calculate normalization factor for each depth.
C   NWINDO = NINT(W/SSEP)+1-NDEAD
C   NFACT = 1
C   FOR I=2,NWINDO
C     NFACT = NFACT+I           ! Y I C K ! ! !
C   END FOR
C   NORM(IZ) = NFACT
C
C   WRITE (40, ) NORM(IZ)
C   N = NARR2-(NARRW+1)/2
  FOR I=1,N
    WT(I, IZ) = 0.0
  END FOR
  J = 0
  FOR I=N+1, NARR-N
    J = J+1
    X = ABS(W/2.-(J-1)*SSEP)
    FACTOR = X/W*2.
    IF (FACTOR.LE.0.000001)
      WT(I, IZ) = 1.
    ELSE
      WT(I, IZ) = SIN(PI*FACTOR)/(PI*FACTOR)
    END IF
  END FOR
  FOR I=NARR-N+1, NARR
    WT(I, IZ) = 0.0
  END FOR
END FOR

C
C Initialize travel time table.
C
  FOR INX=1,WLEN
    FOR INZ=1,NZ

```

```

        BUFFER IN(25, TIME(1, INZ, INX), B, NTRAVEL, ISTAT, NWRDS)
        CALL STATUS(25)
    END FOR
END FOR
DELSHOT = SSEP/XDIST
DSHOT = NINT(DELSHOT)      ! number of cols. between shots
IF (SSEP.LT.XDIST) DSHOT = 1
DS = 0.0                  ! initialize for columns to be output
NLOOP = WLEN-DSHOT        ! no. cols. to be migrated
ISHOT = FSHOT             ! initialize location of first shot (CDP)
SRANGE = XMIN             ! initialize range of first column of migrated
RANGE = SRANGE            ! section
IXS = 1                   !
C initialize index of first column of migrated
IX = IXS                   ! section
NSHIFT = 0
C
C Zero arrays MI and CO.
C
    FOR L=1, WLEN
        FOR N=1, NZ
            MI(N, L) = 0
            CO(N, L) = 0
        END FOR
    END FOR
C
C Loop through total number of shots to be migrated.
C
    LOOP (NSHOT)
C     WRITE (40, '(' ISHOT=', I4)') ISHOT
C
C Transfer one shot from tape to disc. Shot range fixed.
C
        I=0
        IREC = ISHOT-NDEAD      !initialize CDP index of receiver #1
        LOOP (NREC)
            I=I+1
            BUFFER IN(90, TRACE(1), B, NTIME, STAT2, NWORD2)
            CALL STATUS(90)
C
C If end-of-tape is read, suspend execution so new tape may be mounted.
C
C     IF (STAT2.GT.2) CALL NEWTAPE(NWORD2, NTIME)
C
        NWNH = NWORD2-NHEADER ! No. words in data trace, excluding
        TMAX(I) = REAL(NWNH*SAMP) ! Trace length in seconds. header.
        OPEN (UNIT=50, FILE=TRACENAME(I), STATUS="OLD")
        BUFFER OUT(50, TRACE(1), B, NWNH, STAT3, NWORD3)

```

```

      CALL STATUS(50)
C
      IF (STAT3.GT.?)
        WRITE (40, '(' ** ERROR IN SHOT INPUT, STATUS =",I3)') STAT3
      STOP
      END IF
C
      CLOSE(UNIT=50)
      END LOOP
C
C Now perform migration, using the shot just read.
C Loop through all traces (shot-receiver pairs) for one shot.
C K is trace index, from 1 to NREC.
C
      DS = DS+DELSHOT
      IF (ISHOT.GE.NSHOT) DS =1.0
      K = 0
      LOOP (NREC)
        K=K+1
C
        WRITE (40, '(' IREC=",I4)') IREC
C
C
C Input one trace (shot-receiver pair) to memory.
C
      NPTS = TMAX(K)/SAMP      ! total number data samples in trace K
      OPEN (UNIT=50, FILE=TRACENAME(K), STATUS="OLD")
      BUFFER IN(50, TRACE(1), B, NPTS, STAT4, NWORD4)
      CALL STATUS(50)
C
      IF (STAT4.GT.?)
        WRITE (40, '(' ** ERROR IN TRACE INPUT, STATUS =",I3)') STAT4
      STOP
      END IF
C
      REWIND 50
      CLOSE (UNIT=50)
C
C Response at a given subsurface point is found in the data by looping
C through the data, finding the point whose two-way travel time cor-C responds
C to that of the point (x,z) for the shot - receiver pair
C under consideration. Loop through data for each point (x,z) within
C the migration aperture.
C
C ISHOT and IREC are shot and receiver indices for locating correct
C ray in input file TIME.
C
C Migration is done with a sliding aperture of length equal to
C the window length minus distance between shot and first re-

```



```

C ceiver. When a range coordinate is out of range of the trace
C under consideration, that "trace" is output and another "trace"
C is input to the migrated section at the opposite end.
C
C Pass each trace through migration aperture. Aperture length used
C is the length of the aperture at the greatest depth.
C
C     Loop through migration aperture (ie. through X).
C     LX is the location of the column being migrated, relative to
C     the moving aperture.
C
      FOR L = 1, WLEN
        TSHOT = ISHOT - INT((IX-1)/DELSHOT+.001) + NARR2 - 1
        IF (TSHOT.LE.NARR)
C     If shot is out of range, go to next column.
          TREC = TSHOT + IREC - ISHOT
C     If receiver is out of range, go to next column.
          IF (TREC.GT.0)
            CALL SUM(K, L, NZ, NPTS, ISHOT, IREC, DELAY, SAMP, TSHOT,
&TREC)
          END IF
        END IF
        RANGE = RANGE + XDIST
        IX = IX + 1
C     If next column is out of range of the receiver, go to next data
C     trace.
        EXIT FOR IF (RANGE.GT.XMAX)
      END FOR      ! end loop through window
C     increment receiver index within window
      IREC = IREC - DREC ! start at receiver closest to shot
      EXIT LOOP IF (IREC.LE.0)
      RANGE = SRANGE
      IX = IXS
    END LOOP

C
C Have passed all traces from one shot through aperture.
C Output the columns of the migrated section which are out of range
C of the next shot. Skip this loop if no traces are out of_range of
C shots yet.
C
      IF (DS.LT.0.9999) GO TO 30
      IF (ISHOT - NARR2.LT.0) GO TO 30
      NSHIFT = NSHIFT + 1 ! number of times migration aperture shifts
      FOR ICOL = 1, DSHOT
        FOR NN = 1, NZ
C     Multiply coherence by factor of 1000 to prevent the value of the co-C
C     herence from being truncated to zero.
          IF (CO(NN, ICOL).GT.0)
            &      CO(NN, ICOL) = (ABS(MI(NN, ICOL)) * 1000) / CO(NN, ICOL)

```

```

C      MI(NN,ICOL) = MI(NN,ICOL)/NORM(NN)
      END FOR
C
C      WRITE (40, ) ' FOR MIGRATED TRACE ASSOCIATED WITH SHOT ',NSHOT
      WRITE (40, ) (CO(ITEST,ICOL),ITEST=1,NZ)
C
      BUFFER OUT(60,MI(1,ICOL),B,NZ,ISTAT,MWORD)
      CALL STATUS(60)
      BUFFER OUT(70,CO(1,ICOL),B,NZ,ISTAT)
      CALL STATUS (70)
      END FOR
C
C Add new columns to travel time table TIME, ouput columns no longer
C within range of shot, re-index table.
C
      FOR ICOL=1,NLOOP
        NEWCOL = ICOL+DSHOT
        FOR INZ = 1,NZ
          BUFFER OUT(65,TIME(1,INZ,NEWCOL),B,NTRAVEL,ISTAT)
          CALL STATUS (65)
          REWIND 65
          BUFFER IN(65,TIME(1,INZ,ICOL),B,NTRAVEL,ISTAT)
          CALL STATUS (65)
          REWIND 65
        END FOR
      END FOR
      INX = NLOOP
      LOOP (DSHOT)
        INX = INX+1
        FOR INZ=1,NZ
          BUFFER IN(25,TIME(1,INZ,INX),B,NTRAVEL,ISTAT,NWRD)
          CALL STATUS (25)
        END FOR
      END LOOP
C
C Reset indices so that next columns are added to the opposite end in
C the correct order.
C
      FOR ICOL=1,NLOOP
        IC = ICOL+DSHOT
        BUFFER OUT(65,MI(1,IC),B,NZ,NSTAT)
        CALL STATUS (65)
        REWIND 65
        BUFFER IN (65,MI(1,ICOL),B,NZ,NSTAT)
        CALL STATUS (65)
        REWIND 65
      END FOR
      FOR ICOL=1,NLOOP
        IC = ICOL+DSHOT

```

```

        BUFFER OUT(65,CO(1,IC),B,NZ,NSTAT)
        CALL STATUS (65)
        REWIND 65
        BUFFER IN (65,CO(1,ICOL),B,NZ,NSTAT)
        CALL STATUS (65)
        REWIND 65
    END FOR
C   Initialize last columns in migrated section within aperture.
    FOR ICOL = NLOOP+1,WLEN
        FOR INZ = 1,NZ
            MI(INZ,ICOL) = 0
            CO(INZ,ICOL) = 0
        END FOR
    END FOR
C   increment shot index
    DS = 0.0
    IXS = IXS+DSHOT
    SRANGE = SRANGE+SSEP
30  RANGE = SRANGE
    ISHOT = ISHOT+1
    IX = IXS
    END LOOP

C
C   Output traces left in section in memory after last shot has been
C   migrated.  Have already shuffled traces back into first columns.
C
    NOUT = NINT((XMAX-SRANGE)/XDIST)+1
    FOR ICOL = 1,NOUT
        FOR NN = 1,NZ
            IF (CO(NN,ICOL).GT.0.)
                & CO(NN,ICOL) =ABS((MI(NN,ICOL))*1000)/CO(NN,ICOL)
C           MI(NN,ICOL) = MI(NN,ICOL)/NORM(NN)
        END FOR
        BUFFER OUT(60,MI(1,ICOL),B,NZ,NSTAT,MWORD)
        CALL STATUS (60)
        BUFFER OUT(70,CO(1,ICOL),B,NZ,NSTAT)
        CALL STATUS(70)
    END FOR
    STOP
    END

C
C
C
C *****
C
C
C
C   Subroutine calculates the sum for each point within the migration
C   aperture, except those points which have just been added to the

```

```

C aperture.
C
C
SUBROUTINE SUM(K,L,NZ,NPTS,ISHOT,IREC,DELAY,SAMP,TSHOT,TREC)
COMMON /PARAMS/ TMAX(24),WT(30,300)
COMMON /TAPE/ HEADER(2),TRACE(2752)
SPECIAL COMMON TIMES
COMMON /TIMES/ TIME(50,300,30)
SPECIAL COMMON ARRAYS
COMMON /ARRAYS/ MI(300,30),CO(300,30),NORM(300)
INTEGER TSHOT,TREC,PSUM
INTEGER TRACE,HEADER,CO
LOGICAL FLAG
N = 0
LOOP (NZ) ! fix z-coordinate, N is z-index.
10 N = N+1
C
C WRITE (40, ) "N =",N
C
C If entire column is out of range of shot being migrated
C proceed to next column.
C IF (N.GT.NZ) RETURN
C
C Translate CDP indices into indices relative to position of shot and
C receiver within aperture. TSHOT and TREC are locations of shot and
C receiver, respectively, within ray trace aperture, centered around
C the subsurface point. This eliminates need for travel time array to
C be four dimensional.
C L is position of subsurface point relative to the sliding migration
C aperture.
C
C IF (TIME(TSHOT,N,L).LE.0.) GO TO 10
C IF (TIME(TREC,N,L).LE.0.) GO TO 10
C FLAG = .FALSE.
C
C WRITE (40, ) "S ",TSHOT,TIME(TSHOT,N,L)
C WRITE (40, ) "R ",TREC,TIME(TREC,N,L)
C
C TT = TIME(TSHOT,N,L)+TIME(TREC,N,L) !travel time from ray trace
C
C Go to next column of subsurface points if this one is too far
C away to contribute significant response to this data trace.
C
C IF (TT.GT.TMAX(K)+DELAY) GO TO 10
C
C TT is travel time from shot to receiver through sub-
C surface point under consideration
C
C Travel time less than trip delay means no data for point under

```

```

C      consideration - go to next point.
C
      IF (DELAY.GT.TT+.00001)
        WRITE (40, '('" **ERROR - TRAVEL TIME LESS THAN DELAY FOR SHOT",
          &      I3, " TRACE", I3)') ISHOT, IREC
        GO TO 10
      END IF
C
      WTSHOT = WT(TSHOT, N)
      WTREC = WT(TREC, N)
      TTRI = DELAY          ! initialize data sample time
      J = 1                 ! initialize data sample index
C
C Loop through data trace to find travel times closest to TT. J is
C index for data sample.
C
      LOOP (NPTS-1)
        JJ=J+1
        TTR2=TTRI+SAMP
        IF (TTR2.GT.TT)
          IF(TTRI.LE.TT)
C
C          interpolate to get response at subsurface point(x,z)
C          under consideration
C
C
C          WRITE (40, ) K, " TIME 1 =", TTRI, " TIME 2 =", TTR2
C          WRITE (40, ) "      TRACE 1 =", TRACE(J), "TRACE 2 = ", TRACE(JJ)
C
          P = TRACE(J)+(TT-TTRI)/(TTR2-TTRI)*(TRACE(JJ)-
          &      TRACE(J))
          FLAG = .TRUE.
C
C Response at given shot-receiver pair is assigned to all subsurface
C points from which it originated.
C
          PSUM = INT(P*WTSHOT*WTREC)
          MI(N, L) = MI(N, L)+PSUM
          CO(N, L) = CO(N, L)+ABS(PSUM)
          END IF
          END IF
          EXIT LOOP IF(FLAG)
          J = JJ
          TTRI = TTR2
        END LOOP      ! end loop through data trace
      END LOOP      ! end loop through depth
      RETURN
      END
C

```

```
C
C
C *****
C
C ***** SUBROUTINE NEWTAPE *****
C
C Subroutine halts program execution when end-of-tape is read during
C BUFFER IN. Program waits for re-start command after new data tape
C is loaded. Execution resumes where it left off.
C
C SUBROUTINE NEWTAPE(NWORD2,NTIME)
C COMMON /TAPE/ HEADER(2),TRACE(2752)
C INTEGER JUNK(100)
C INTEGER STAT2
C REWIND 4 ! rewind tape just processed
C
C PAUSE ! load new tape then type RP cp
C
C LOOP(4) ! skip tape headers and EOF mark
C BUFFER IN (4,JUNK(1),B,100,ISTAT)
C CALL STATUS(4)
C END LOOP
C BUFFER IN(4,HEADER(1),B,NTIME,STAT2,NWORD2) ! read first trace
C CALL STATUS(4)
C IF (STAT2.NE.?)
C WRITE (40, ) "ERROR - TAPE INPUT STATUS = ",STAT2
C STOP
C END IF
C RETURN
C END
```

APPENDIX B

INCREASING RAY TRACING EFFICIENCY

Tracing rays to obtain travel times for migration requires significantly more computation time than the Kirchhoff depth migration process itself. Ray tracing performed for the model migrated above required more than five times the CPU time used to migrate the model. Decreasing the CPU time required for tracing rays would significantly increase the efficiency of this migration process.

Travel times for migrating the velocity model were obtained by tracing rays upward from each depth point. Gray (1986) developed a ray tracing method which is faster by a factor of N_z , where N_z is the number of depth values needed in the travel time table to be constructed. Shorter computation time is achieved by tracing rays downward from each shot point at the surface (Figure 17). Travel times from the surface to each grid point are accumulated as each ray is propagated by interpolating between travel times along rays bracketing each depth point. In the process of tracing rays upward from each depth point rays are re-traced for each depth point lying near a given raypath. By tracing rays downward from the surface a single raypath is used to obtain travel times for all depth points lying along it. Reciprocity is still assumed since rays are only traced downward.

Kalaba (1961) showed that the solution to the eikonal equation (the ray tracing equation) is the shortest travel time path, or optimal raypath, between two points. Bellman's (1957) principle of optimality defines the

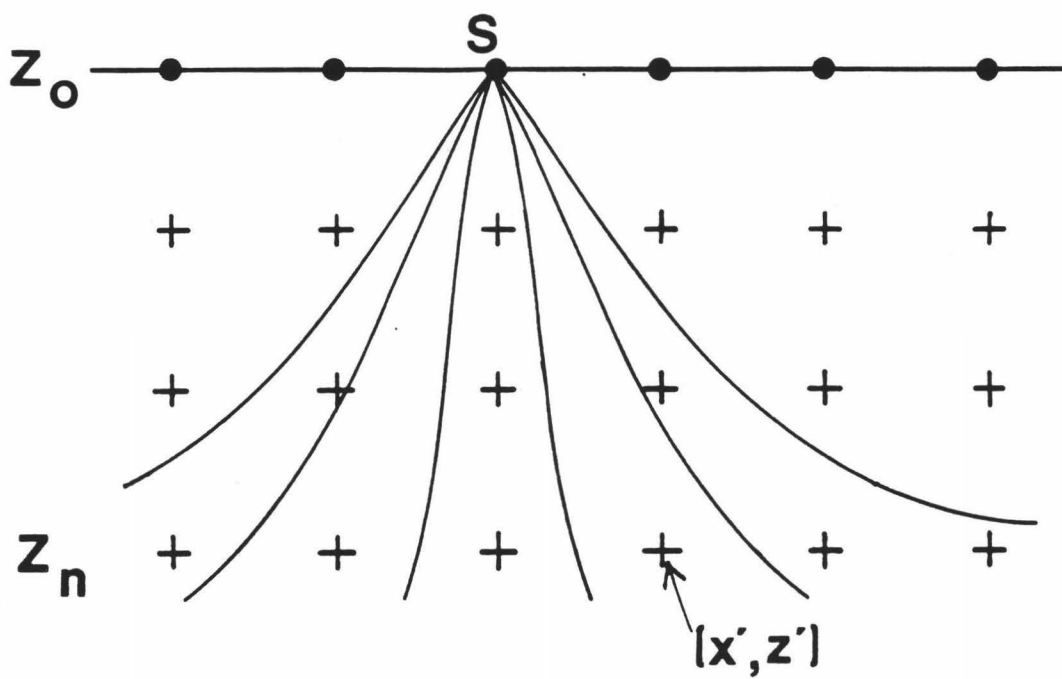


Figure 17. Rays traced downward from shot S and through the grid of depth points (x', z') .

optimal raypath as the raypath composed of optimal ray segments. Dynamic programming (Kalaba, 1961) employs optimality to construct fast, efficient computer codes and can be easily applied to Gray's (1986) ray tracing technique. Ray tracing incorporating dynamic programming can be used to calculate travel times for the same grid of points used to migrate the model as described above. As before, the velocity model is represented by convex polygons of uniform velocity. Departing from the method described by Gray (1986) rays are traced directly to the depth points making up the grid (Figure 18). A fan of rays is traced upward from each point (x_n, z) to the row of points $(x_n', z - \Delta z)$ above, where the distance between ranges x_n' is less than the horizontal distance between grid points x_n (Figure 18). For each point (x_n, z) the shortest travel time from depth $z - \Delta z$ to depth z is summed with the minimum travel time previously calculated from the surface to point $(x_n, z - \Delta z)$ to obtain the minimum travel time from a surface point $(x_n, 0)$ to depth z . Travel times associated with points coincident with grid points are saved as travel times along the optimal ray path from a given shot location at the surface to each grid point. This procedure is repeated for each pair of rows of depth points z_n and $z_n + \Delta z$, moving downward in the section.

Vertical and horizontal distances between grid points, Δz and Δx , are found according to the same criteria described for the upward ray tracing method. Separation between the points x_n' at which travel times are actually calculated should be chosen such that the cumulative difference between travel times from the surface to adjacent points at the greatest depth within the section to be migrated is no more than a few time samples. Differences in travel times greater than two or three times the sampling rate will not

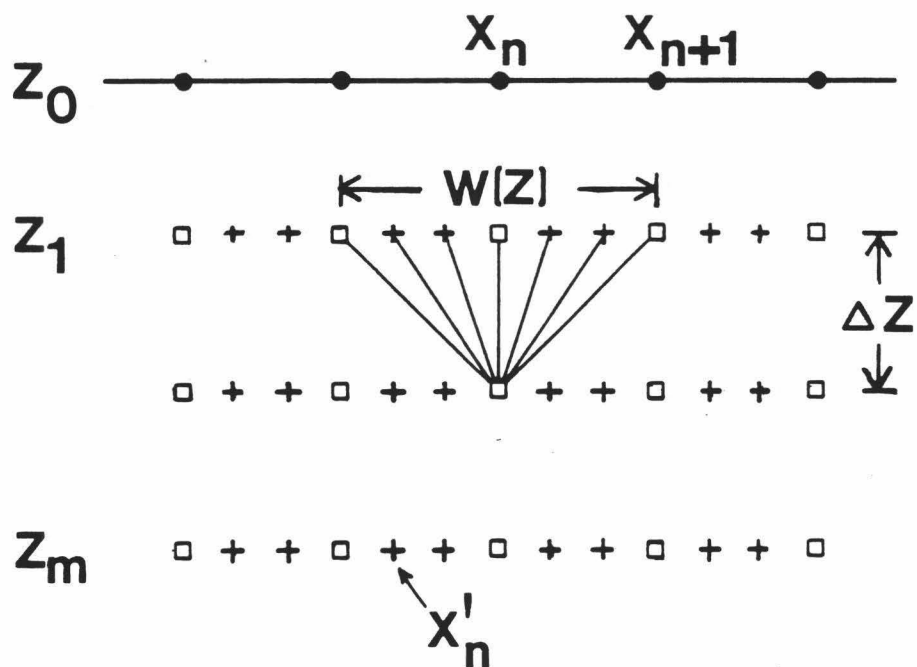


Figure 18. Tracing rays from point (x_n, z_m) to points $(x'_n, z_m - \Delta z)$ to determine shortest travel time path from the surface to (x_n, z_m) .

rigorously locate the minimum travel time path. If this distance is constant throughout the section, shallow depths will be oversampled. Tracing rays through a more closely spaced grid of points than that used in migration increases the accuracy of the calculated travel times without an increase in the amount of storage required for ray tracing and without interpolation.

Since only the minimum travel time from each shot point to each depth point is saved the travel time table generated using dynamic programming contains only first arrivals. Arrivals on the later branches of the travel time curve are not modelled. This top down ray tracing technique efficiently models caustics only if principles of dynamic programming are not used in development of the ray tracing code.

The ray trace aperture appropriate to this ray tracing technique is equal to the aperture calculated for tracing rays upward from each depth point. The same criteria apply. However, this entire aperture need not be searched to find the minimum travel time between depths, z and $z+\Delta z$. This search window length is dependent upon the critical angle where the velocity contrast is highest. Rays will diverge most widely in this region.

This alternate method of generating the table of travel times was not implemented due to lack of time and funding.

LITERATURE CITED

- Baysal, E., Kosloff, D. D., and Sherwood, J. W. C., 1983, Reverse time migration: *Geophysics*, 48, 1514-1524.
- Bellman, R., 1957, *Dynamic Programming*, Princeton University Press: New Jersey.
- Berkhout, A. J., 1980, *Seismic Migration: Imaging Acoustic Energy by Wave Field Extrapolation, Practical Aspects: Developments in Solid Earth Geophysics 14B*, Elsevier Scientific Publishing Company: Amsterdam, 274 p.
- Berkhout, A. J., 1984, Multidimensional linearized inversion and seismic migration: *Geophysics*, 49, 1881-1895.
- Carter, J. A. and Frazer, L. N., 1983, A method for modelling reflection data from media with lateral velocity changes: *Journ. Geophys. Research*, 88, 6469-6476.
- Chun, J. H., and Jacewitz, C. A., 1981, Fundamentals of frequency domain migration: *Geophysics*, 46, 717-733.
- Claerbout, J. F. and Doherty, S. H., 1972, Downward continuation of moveout corrected seismograms: *Geophysics*, 37, 741-768.
- Deregowski, S. M. and Brown, S. M., 1983, A theory of acoustic diffractors applied to 2-D models: *Geophys. Prosp.*, 31, 293-333.
- DeVries, D. and Berkhout, A. J., 1984, Influence of velocity errors on the focussing aspects of migration: *Geophys. Prosp.*, 32, 629-648.
- French, W. S., 1974, Two dimensional and three-dimensional migration of model-experiment reflection profiles: *Geophysics*, 39, 265-277.

- French, W. S., 1975, Computer migration of oblique seismic reflection profiles: *Geophysics*, 40, 961-980.
- Gardner, G. H. F., French, W. S., and Matzuk, T., 1974, Elements of migration and velocity analysis: *Geophysics*, 39, 811-825.
- Gazdag, J., 1978, Wave equation migration with the phase shift method: *Geophysics*, 43, 1342-1351.
- Gazdag, J., 1981, Modeling of the acoustic wave equation with transform methods: *Geophysics*, 46, 854-859.
- Gebrande, H., 1976, A seismic ray tracing method for two-dimensional inhomogeneous media, in *Explosion Seismology in Central Europe*, P. Geise and others eds., New York: Springer - Verlag.
- Gray, S. H., 1986, Efficient travelttime calculations for Kirchhoff migration: *Geophysics*, 51, 1685-1688.
- Hatton, L., Lerner, L., and Gibson, B. S., 1981, Migration of seismic data from inhomogeneous media: *Geophysics*, 46, 751-767.
- Hilterman, F. J., 1975, Amplitudes of seismic waves -- A quick look: *Geophysics*, 40, 745-762.
- Hosken, J. W. J. and Deregowski, S. M., 1985, Tutorial migration strategy: *Geophys. Prosp.*, 33, 1-33.
- Hubral, P., 1977, Time migration - some ray-theoretical aspects: *Geophys. Prosp.*, 25, 738-745.
- Inderwiesen, P. L., 1985, Exact Kirchhoff migration of unstacked seismic data, University of Houston, (unpublished dissertation).
- Judson, D. R., Lin, J., Schultz, P. S., and Sherwood, J. W. C., 1980, Depth migration after stack: *Geophysics*, 45, 361-375.

- Kalaba, R., 1961, Dynamic programming, Fermat's principle and the eikonal equation: Journ. Optical Soc. Am., 5, 1150-1151.
- Khun, M. J., and Alhilali, K. A., 1977, Weighting factors in the construction and reconstruction of acoustical wavefields: Geophysics, 42, 1183-1198.
- Kosloff, D. D., and Baysal, E., 1982, Foreward modeling by a Fourier method: Geophysics, 47, 1402-1412.
- Kosloff, D. D. and Baysal, E., 1983, Migration with the full acoustic wave equation: Geophysics, 48, 677-687.
- Kuo, J. T. and Dai, T. F., 1984, Kirchhoff elastic wave migration for the case of noncoincident source and receiver: Geophysics, 49, 1223-1238.
- Larner, K. L., Hatton, L., Gibson, B. S., and Hsu, I-C, 1981, Depth migration of imaged time sections: Geophysics, 46, 734-750.
- Lowenthal, D., Lu, L., Roberson, R., and Sherwood, J., 1976, The wave equation applied to migration: Geophys. Prosp., 24, 380-399.
- May, B. T., and Covey, J. D., 1983, Structural inversion of salt dome flanks: Geophysics, 48, 1039-1050.
- Mufti, I. R., 1985, Seismic migration: basic concepts and popular methods, Part 1: Geophysics: The Leading Edge of Exploration, 4, no. 8, 24-28.
- Neidell, N. S., and Taner, M. T., 1971, Semblance and other coherency measures for multichannel data: Geophysics, 36, 482,497.
- Owusu, K., Gardner, G. H. F., and Massell, W. F., 1983, Velocity estimates derived from three-dimensional seismic data: Geophysics, 48, 1486-1497.
- Reshef, M. and Kosloff, D., 1986, Migration of common - shot gathers: Geophysics, 51, 324-331.

- Safar, M. H., 1985, On the lateral resolution achieved by Kirchhoff migration: Geophysics, 50, 1091-1099.
- Sattlegger, J. W., 1975, Migration velocity determination: Part 1, philosophy: Geophysics, 40, 1-5.
- Schneider, W., 1978, Integral formulation for migration in two and three dimensions: Geophysics, 43, 49-76.
- Schultz, P. and Sherwood, J., 1980, Depth migration before stack: Geophysics, 45, 376-393.
- Stolt, R., 1978, Migration by Fourier transform: Geophysics, 43, 23-48.
- Trorey, A. W., 1970, A simple theory for seismic diffractions: Geophysics, 35, 762-784.
- Waters, K. H., 1981, Reflection Seismology: A Tool for Energy Resource Exploration, 2nd ed., New York: John Wiley and Sons, 453 p.
- Whittall, K. P. and Clowes, R. M., 1979, A simple, efficient method for the calculation of travel times and raypaths in laterally inhomogeneous media: Journ. Canadian Society of Exploration Geophysicists, 15, 21-29.