UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

A comprehensive system for interpreting seismic-refraction arrival-time data using interactive computer methods

by

Hans D. Ackermann

Leroy W. Pankratz

Danny A. Dansereau

Open File Report 82-1065

1982

ABSTRACT

A set of interactive computer programs are described whose purpose is to calculate velocity and depth models from refraction arrival-time data. The intent is to arrive at one or more such models which, if reinverted into travel times will satisfy the initial data to within the limits of reading accuracy. Three main programs, in the form of subroutines, have been written to accomplish this. One program, named RECIP, calculates depths and velocities of a refracting horizon from reversed arrival-time data given a velocity distribution for the overlying layers. Another program, named CRIT, calculates depths of a refracting horizon from unreversed arrival-time data given a velocity distribution in the overlying layers and an assumed velocity distribution of the refracting horizon. The third program, called RAYTRACE, calculates arrival-time data points for the refracting horizons of a velocity model from assumed shotpoint locations and depths. Numerous other subordinate subroutines are included for manipulating and extending data, inserting hidden layers and otherwise varying the velocity distribution so that any legitimate arrival-time data set can be analyzed and will yield valid results. In addition, a section on manipulating input data to provide an internally consistent set of plots, and also a section on theory are included. Numerous example problems are provided.

INTRODUCTION

These programs were written to satisfy the need to calculate, on a routine basis, velocity and depth sections which satisfy refraction arrival-time plots. Any generalized scheme for doing this must permit velocities to vary both laterally and vertically. The application of reciprocal methods to reversed plots provides the means to uniquely determine both changes in depth and lateral changes in velocity of a refracting horizon (assuming a known overlying velocity distribution) in essentially the same operation. In the case of one-way (unreversed) data, depths to the refracting horizon are uniquely determined by assuming its lateral velocity function obtained from the average slope of the arrival-time curves. The importance of determining lateral changes in velocity as well as variations in depth can hardly be overemphasized. Because rock properties and subsurface conditions are estimated from velocities, lateral changes may infer zones of weakness, fractured zones, or changes in lithology.

The interactive methods presented here require that a velocity model be constructed one layer at a time and permit the user to vary the overlying velocity distribution as seen fit to arrive at any number of solutions of the underlying refracting horizon. Solutions obtained should generally satisfy the initial data to within the accuracy of the arrival-times picked from the seismogram.

Although graphical-reciprocal methods have long been available to interpret both changes in depth and velocity of the refracting horizon (Thornburgh, 1930; Rockwell, 1967, Schenck, 1967), the tedium of these processes render their general application impractical. More rapid approximate methods exist, many of which we have used with generally satisfactory results, particularly the methods described by Hawkins (1961), Slotnick (1950 and 1959) and Scott (1972). However, upon arriving at a

solution through these methods, application of the inverse process of computing the original arrival-time data from the calculated model frequently resulted in significant differences which could only be reconciled by adjusting the model through trial and error. Scott's method employs interactive computer techniques and is rapid. It arrives at a "best" solution through successive iterations which vary the depth of the refracting horizon. However, in general a solution which satisfies the data cannot be obtained unless both depths and velocities of the refracting horizon are allowed to vary. Instead of building on his method by permitting velocities to also vary laterally, we decided to forego an iterative process completely and attack the problem analytically.

The reciprocal methods of Thornburgh (1930), Rockwell (1967) and Schenck (1967) require the graphical reconstruction of emerging wave fronts through the use of tangents to circles obtained from the arrival-time curve and the overlying velocity distribution. Instead of following this method for reversed plots, we adopted a scheme for reconstructing rays perpendicular to the wave fronts, using the derivative of the travel-time curve (Slotnick, 1959, p. 9) and the overlying velocity distribution. New travel-time curves are constructed successively on all intervening horizons between the surface and the one being calculated which in essence reduces a problem of m layers into a two-layer problem. These ray-tracing methods have also been adapted to unreversed data, through a modification of Slotnick's (1950) method, for which the velocity variation along the refracting horizon is assumed. In both cases, the program will accept up to ten discrete layers each of which may vary in depth and laterally in velocity.

The programs are organized into a main driver program named SEISMC from which three main subroutines and ten auxillary subroutines may be called. The auxiliary subroutines are used for reading and writing data files, plotting, and manipulating data. The three main subroutines, RECIP, CRIT, and RAYTR, actually make depth and velocity calculations. RECIP is a program invoking reciprocal methods for reversed data. CRIT calculates depths from unreversed data and an assumed velocity. RAYTR does the inverse process, of calculating refracted arrival-times from a velocity model and a shotpoint location. The total package contains twenty additional subroutines which are called automatically during program execution.

The main text in this report is in five sections, titled Manipulating Arrival-Time Curves, Theory, The Computer Programs and Sample Problems, Entry Forms and Conditions (Appendix A) and Program Listings (Appendix B).

The section, Manipulating Arrival-Time Curves, is a discussion of the nature of refraction arrival-time plots. It demonstrates how a single data arrival-time set may mathematically fit a variety of models, and discusses the conditions which must be satisfied, and the way interrelated data may be manipulated to yield an internally consistent set of plots. Clearly, no computational scheme can result in a good interpretation if the input data violate basic physical laws. The success in using these programs hinges in large part on the care given to the initial data handling.

The section, Theory, discusses the equations and graphically illustrates the algorithms executed by subroutines RECIP, CRIT, and RAYTR.

The section, The Computer Programs and Sample Problems, discusses and provides examples of the execution for each of the main and auxillary subroutines called from the driver program SEISMC. The functions of the remaining subroutines are also discussed. The subroutines are described in an order of increasing complexity. The auxiliary subroutines are described first, followed by the main subroutines RAYTR, CRIT, and RECIP. The numerous examples illustrate the different options and also some possible pitfalls.

Appendix A serves as a manual for the execution of the programs. It defines the input variables and coded messages which may be output during execution. It also contains sample forms for listing input parameters. Logically, the information in this appendix also belongs in the section The Computer Programs and Sample Problems and is frequently referred to by it.

The individual programs are listed in Appendix B.

The initial version of these programs, completed in 1975, worked well for refraction data derived from theoretical earth models. However, problems related to the scatter of real field data, and errors in constructing the model of the layers overlying the one being calculated sometimes resulted in intolerable scatter of computed subsurface depth points. Thus the following programs evolved as different problems were encountered with each new batch of field data. The programs are still being modified and appended. For example, a subroutine to analyze computed lateral velocity variations would be helpful. Turn-around time can be improved through the use of computer graphics. The system for automatic digitization of seismograms, briefly touched on in this report, using punched cards, is cumbersome and has recently been changed. Our representation of the overlying velocity distribution, in terms of discrete cells or compartments, is not entirely realistic and could be modified to allow both lateral and vertical linear velocity changes. Finally, the utilization of state-of-the-art digital recording field equipment with large dynamic range offers a wide range of application, particularly to the use of secondary arrivals and the study of signal attenuation.

MANIPULATING ARRIVAL-TIME CURVES

Seismic refraction data are interpreted in terms of successively deeper horizons of increasing velocity. According to ray theory, the shallowest horizon (horizon 1) is defined by a direct ray propagating from the energy source (shotpoint) along the ground surface. Each successive underlying horizon corresponds to the path of a critically refracted (or diffracted) ray, each propagating with successively higher velocity (fig. 1). Geologic inferences result through correlating variations in the depth and velocity of the critically refracted rays to changes in depth and physical properties of geologic layers.

Each such critically refracted ray, according to Huygen's principle, returns energy to the surface along its entire travel path. Thus, in principal at least, these returning arrivals can be detected, and for each critically refracted ray, a plot prepared of arrival time versus distance from the shotpoint. Thus for the four-layer problem of figure 2, four individual arrival-time curves may, in principal, be plotted for every

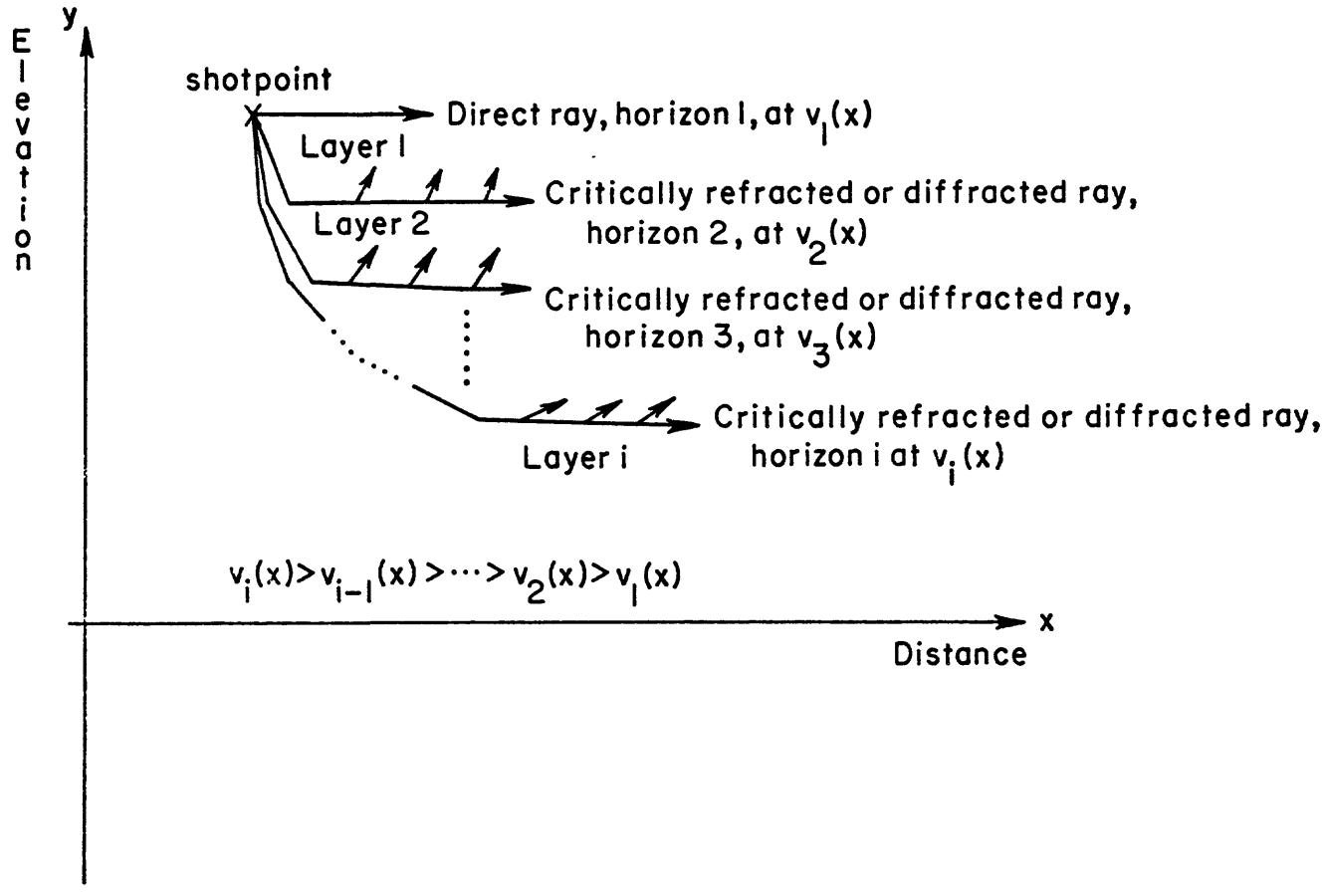

Figure 1.--Hypothetical refracted ray paths for successive horizons of increasing velocity.....

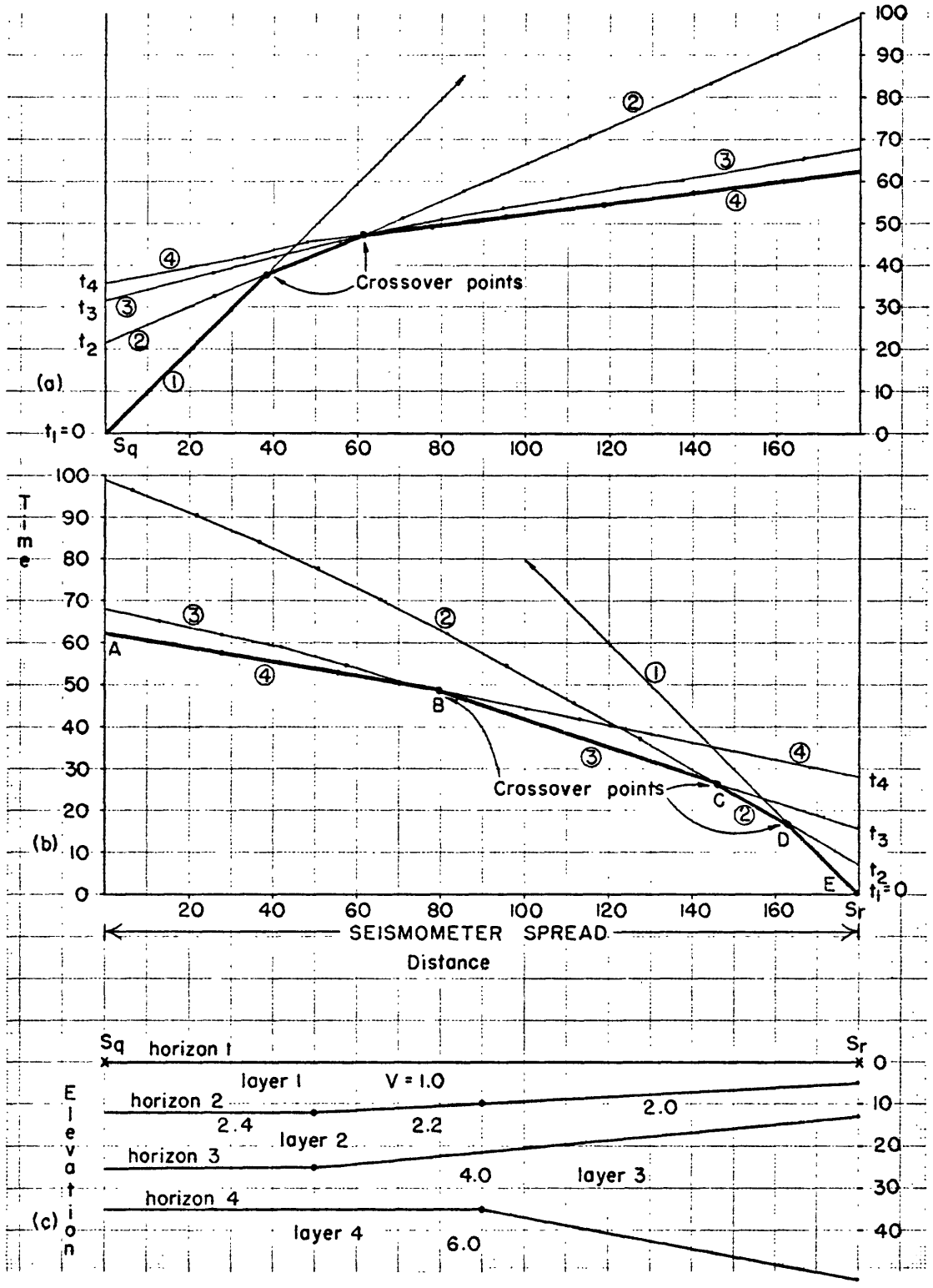

Figure 2.--Arrival-time curves (a) and (b) of a four-layer velocity model (c) for shotpoint locations Sq and Sr. Heavy line segments in (a) and (b) represent first arrivals.....

shotpoint. In a mathematical sense, the arrival time curve for each critically refracted ray may be thought of as passing through the time axis at the shotpoint, and the intersection of the curve for the ith horizon with the time axis is called its intercept time (t_i).

Unfortunately, from the typical refraction seismogram, critically refracted arrivals from each horizon can not be identified across the entire length of the seismometer spread. In fact, in most cases, only initial or first arrivals can be read, resulting in a single valued function of arrival-times versus distance (the t-x plot) as shown by the heavy lines in figures 2a and 2b, for shotpoints labeled S_q and S_r. Thus instead of a complete curve corresponding to each horizon along the entire length of the seismometer spread, only segments are obtained, as seen by AB, BC, CD and DE in figure 2b. Frequently even segments of arrival time curves for particular horizons fail to appear as first arrivals (the hidden layer problem) as seen for horizon 3 in figure 2a.

Inflections of arrival-time curves

The points of intersection between segments representing arrivals from successive horizons are called crossover points (fig. 2). Crossover points are usually marked by a distinct inflection or change in slope of the first arrival-time curve.

Inflections of first-arrival-time curves are also caused by lateral variations, either in velocity or dip of the refracting horizon and/or overlying beds.

Probably the three most difficult, yet extremely important set of tasks for the interpreter are: 1--to relate inflection point on arrival-time curves as due either to crossovers, or lateral changes in velocity or dip of the refracting layer or variations in velocity or dip of the overlying layers, 2-- associate the segments between crossover points with their corresponding refracting horizon, and 3-- from the limited first-arrival data on hand, either implicitly or explicitly reconstruct the hidden remainder of the arrival-time curve for each horizon. Thus in figure 2, this would entail identifying and reconstructing, from the first-arrival data only, each of the eight complete arrival-time curves (four derived from shotpoint Sq and from Sr). The accuracy of the final computations hinges to a large degree on how well horizon identifications are made and the degree to which the extensions of individual first-arrival line segments agree with the true unobserved arrivals. Multiple models (or solutions) result from different reconstructions of the unobserved part of arrival-time curves, each of which may be theoretically valid.

A simple example of multiple solutions for a single unreversed arrival-time curve, obtained by identifying segments with refracting horizons in different ways, is illustrated by the curve of figure 3a. Figure 3b shows one of an infinite number of three-layer solutions. Each inflection represents a crossover point and different solutions are obtained by arbitrarily varying the velocity of the two lower layers. For this particular solution, velocities of 2.5 and 5.0 were used for layers two and three.

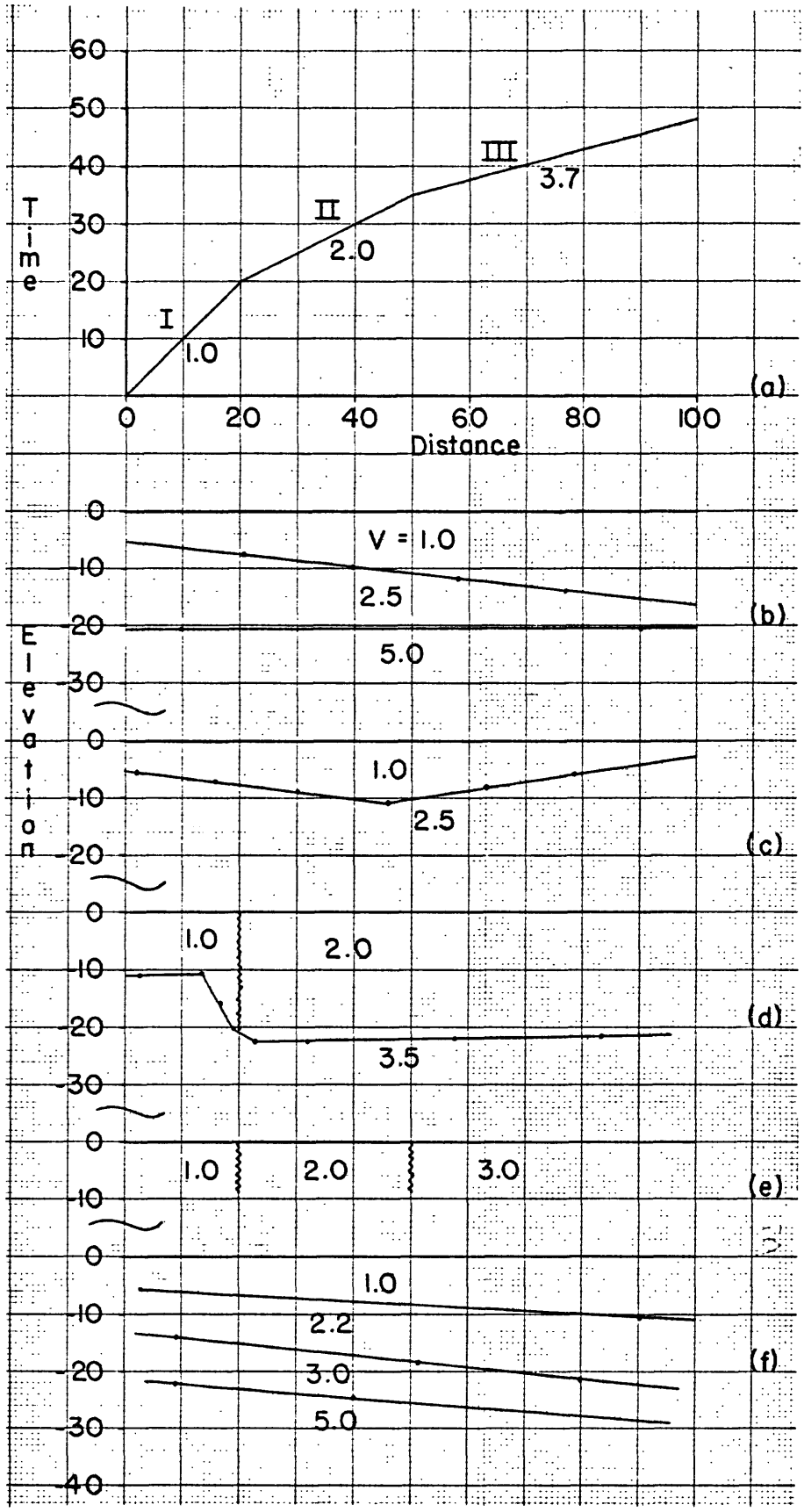

Figure 3.--Five valid solutions (b) through (f), computed from the unreversed arrival-time curve of (a).....

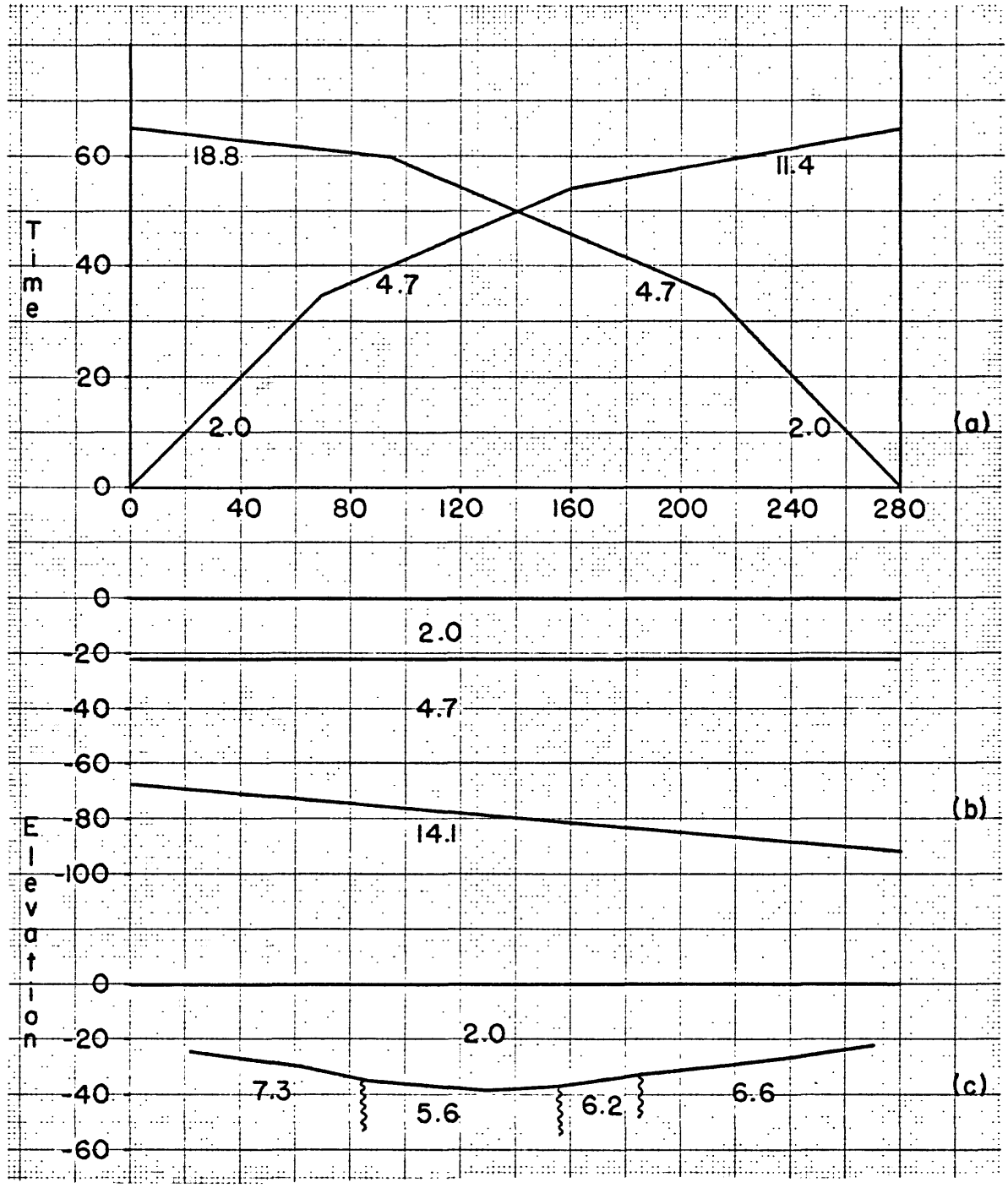

Figure 4.--Two valid solutions (b) and (c), computed from the reversed arrival-time curve of (a).....

Figures 3c and 3d show solutions involving two layers. Figure 3c represents a case for which the change in slope between segments II and III is due to a change in dip of horizon 2. Figure 3d shows a solution for which the change in slope between segments I and II is due to a lateral change in velocity of layer 1. Figure 3e shows a solution involving a single layer with three abrupt lateral changes in velocity. Finally, figure 3f shows a four-layer solution which includes a hidden layer. All of these solutions satisfy the data of figure 3a.

Reversed profiles also have multiple solutions as shown in figure 4. One possible solution is that a different layer corresponds to each line segment of the t-x graph (fig. 4b). Another solution that might be encountered when shooting across an alluvial valley with steeply sloping bedrock along the flanks is shown in figure 4c.

Inflections corresponding to crossover points are nearly always manifested by a decrease in slope of the t-x graph (an increase in apparent velocity) with increased distance from the shotpoint. This has been amply illustrated in the reversed arrival-time curves of figure 2. Under certain conditions involving changes in dip or velocity of overlying layers, the crossover point will not involve a change in slope at all and may even be marked by an increase in slope (a decrease in apparent velocity) with increased distance from the shot point.

Inflections resulting from lateral changes in velocity or dip are illustrated in figures 5, 6 and 7 using 4-layer problems for illustration. Reversed arrival-time curves from horizon 4 are plotted to show the effects in opposite directions. Arrival-times from the overlying three horizons are not plotted.

Figure 5a shows the effect of a change in velocity of horizon 4 with no change in dip. We note approximate equal intercept times for both shotpoints because depth is constant, and a higher apparent velocity on the arrival-time curves over the part of layer 4 with higher velocity. For the case of a gradual change in dip or curved horizon with constant velocity (fig. 5b), the intercept times at opposite shotpoints are significantly different and the apparent velocity corresponding to the reversed curves increase in opposite directions, similar to the effect of an increase in velocity with depth.

The effects of changes in velocity of overlying beds are shown in figure 6a. If the velocity transitions were linear instead of discontinuous, the abrupt changes in slope shown on the graphs would be removed, and the t-x graphs would consist of two straight lines of different slope.

Diffraction patterns from a point source can cause marked inflections on arrival-time curves (fig. 6 b). They are usually associated with faults but can arise from an abrupt change in dip. Segment C of the time-distance curve recorded from a shotpoint at the left end of figure 6b results from a diffraction pattern originating at point c and, likewise, segment D on the reverse curve arrises from point d.

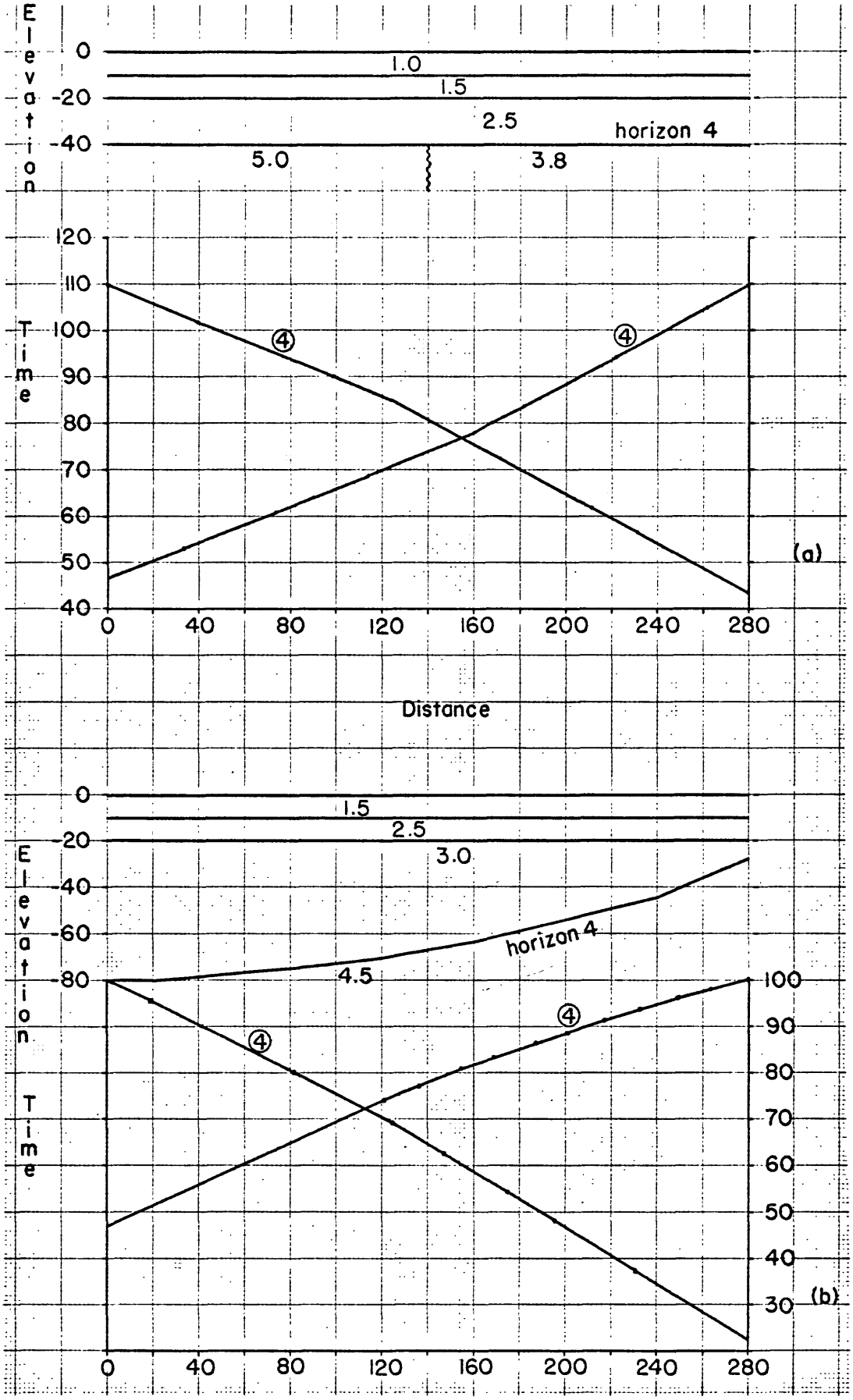

Figure 5.--(a) Inflections on reversed arrival-time curves due to a lateral change in velocity of the refracting horizon.
(b) Inflections on reversed arrival-time curves due to a gradual change in dip of the refracting horizon.....

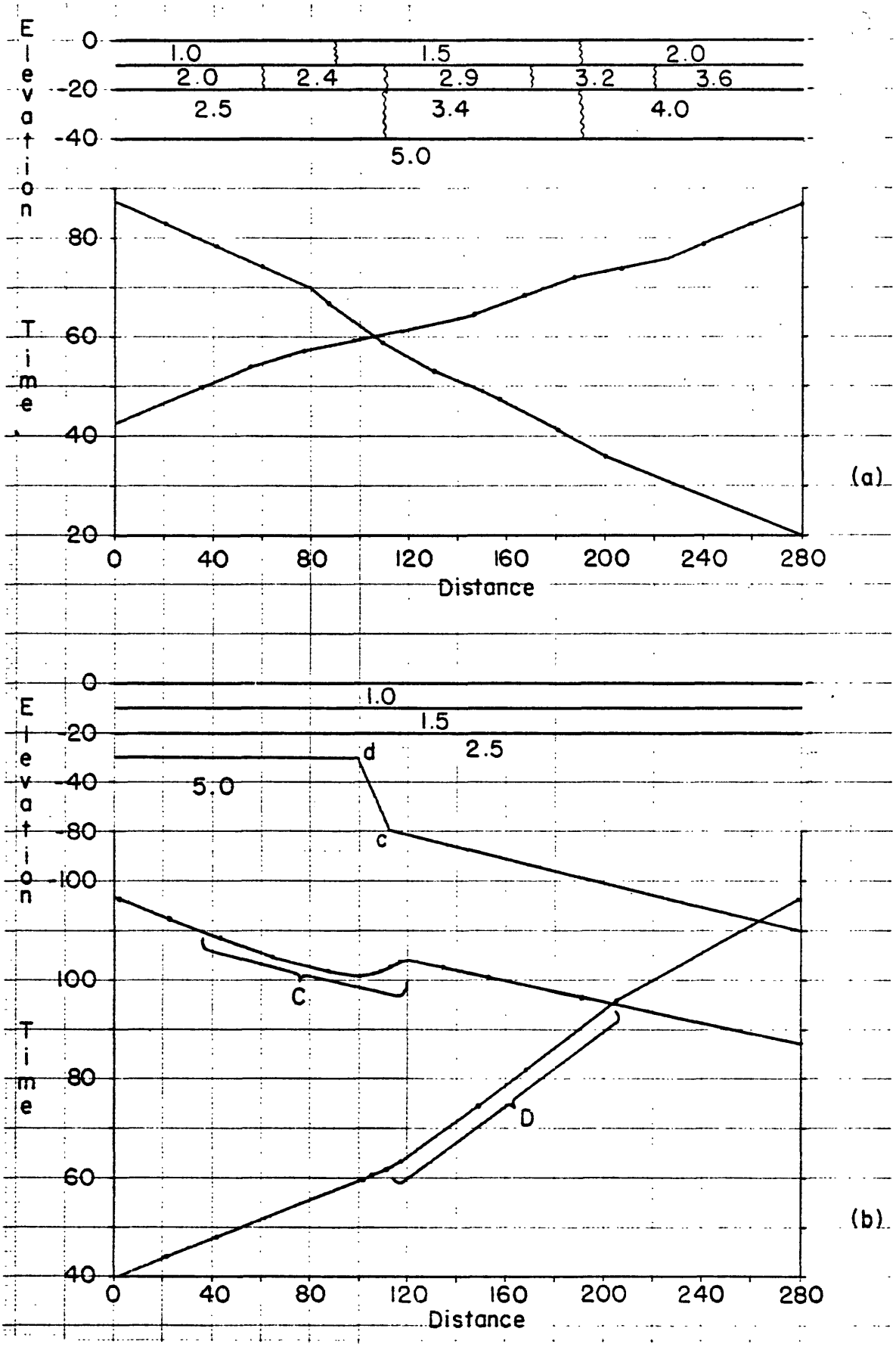

Figure 6.--(a) Inflections on arrival-time curves due to lateral changes in velocity of beds overlying the refracting horizon.
(b) Inflections on arrival-time curves due to an abrupt change in depth of the refracting horizon.....

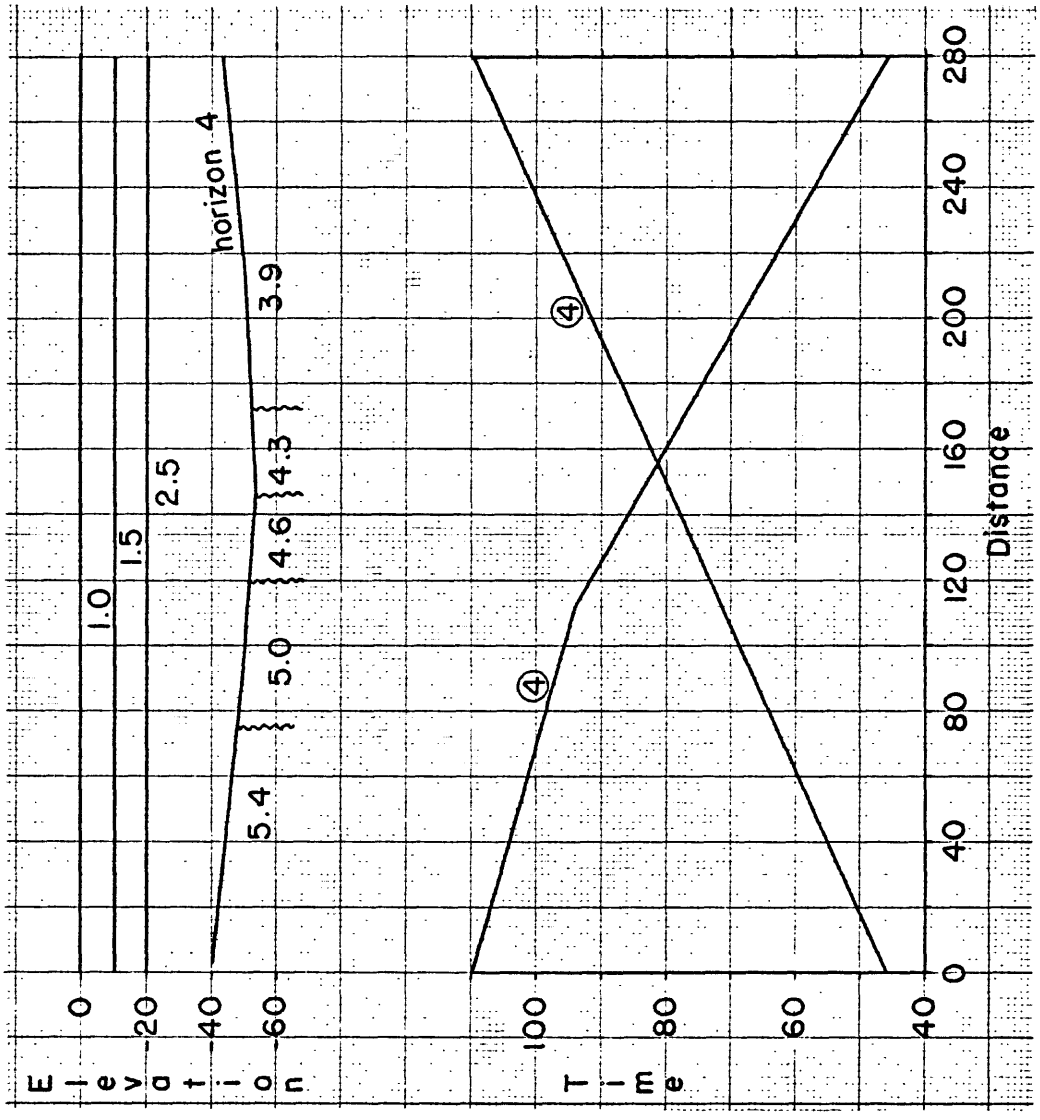

Figure 7.--Trough-like velocity feature derived from reversed arrival-time curves, one of which has no inflections and the other just a single abrupt break.....

These examples illustrate the diversity of shapes of arrival-time curves that can result from relatively simple subsurface changes. As geology becomes complex, the resulting complexity of arrival-time curves increase to the point where simple identification of cause and effect is impossible. Figure 7b, for example, illustrates a t-x graph with a marked inflection in one direction but none in the reverse direction. One possible solution is a trough-like feature with laterally varying velocity in the basal layer. These examples show that the interpreter must distinguish between breaks in slope of arrival-time data due to crossovers from those due to lateral geologic changes. Then the data can be manipulated in a manner consistent with the laws of refraction into a form which can be handled by the computer programs. Once done, the computer programs themselves calculate the lateral changes in velocity and depth for the individual horizons which satisfy the perturbations of arrival-time curves.

Identification of crossover points

The process of unambiguously identifying crossover points and relating segments of arrival-time curves between them with specific refracting horizons requires multiple shotpoints. The first step, however, usually consists of making a preliminary identification (guess) using individual plots. This is illustrated in figure 8, which shows possible horizon identifications represented as straight line segments for a variety of hypothetical arrival time curves. In general, crossover points are identified with the major decreases in slope with distance shown by the dots in figure 8.

Once such preliminary identifications have been made, related pairs of plots may be used for verification or making changes. Such pairs may be classified as reversed, opposed, or complementary plots and each serve in their own way to facilitate horizon identification. Given a sufficient number of related plots, unambiguous identification can usually be achieved.

Reversed plots

Reversed plots have common seismometer locations with shotpoints recorded in opposite (reversed) directions (fig. 9a). In these illustrations the shotpoints are located at the ends of the spread. However, in general, this condition need not be satisfied. Thus reversed shotpoints may be internal to, or off the ends of a spread. Because minimum time paths are independent of direction, a necessary condition for reversed plots is that reciprocal times corrected for shot depth must be equal; namely, the arrival-time TR from S_q to S_r (fig. 9a) must equal that in the reverse direction from S_r to S_q. Furthermore, the same number of refracting horizons must be identifiable on the reversed plots and, at the reciprocal shotpoints refractions must arise from the same horizon. Hence at the reciprocal points the respective horizon number identifiers must be the same.

An example is shown in figure 9. A possible 3-layer solution is shown in figure 9b; segments BC and CD from shotpoint S_q might represent horizon 2 and segment DE might represent horizon 3. Then segment GH of the reversed plot must also represent horizon 3 because, at reciprocal points, refractions must arise from the same horizon. Horizon 2 for shotpoint S_r is

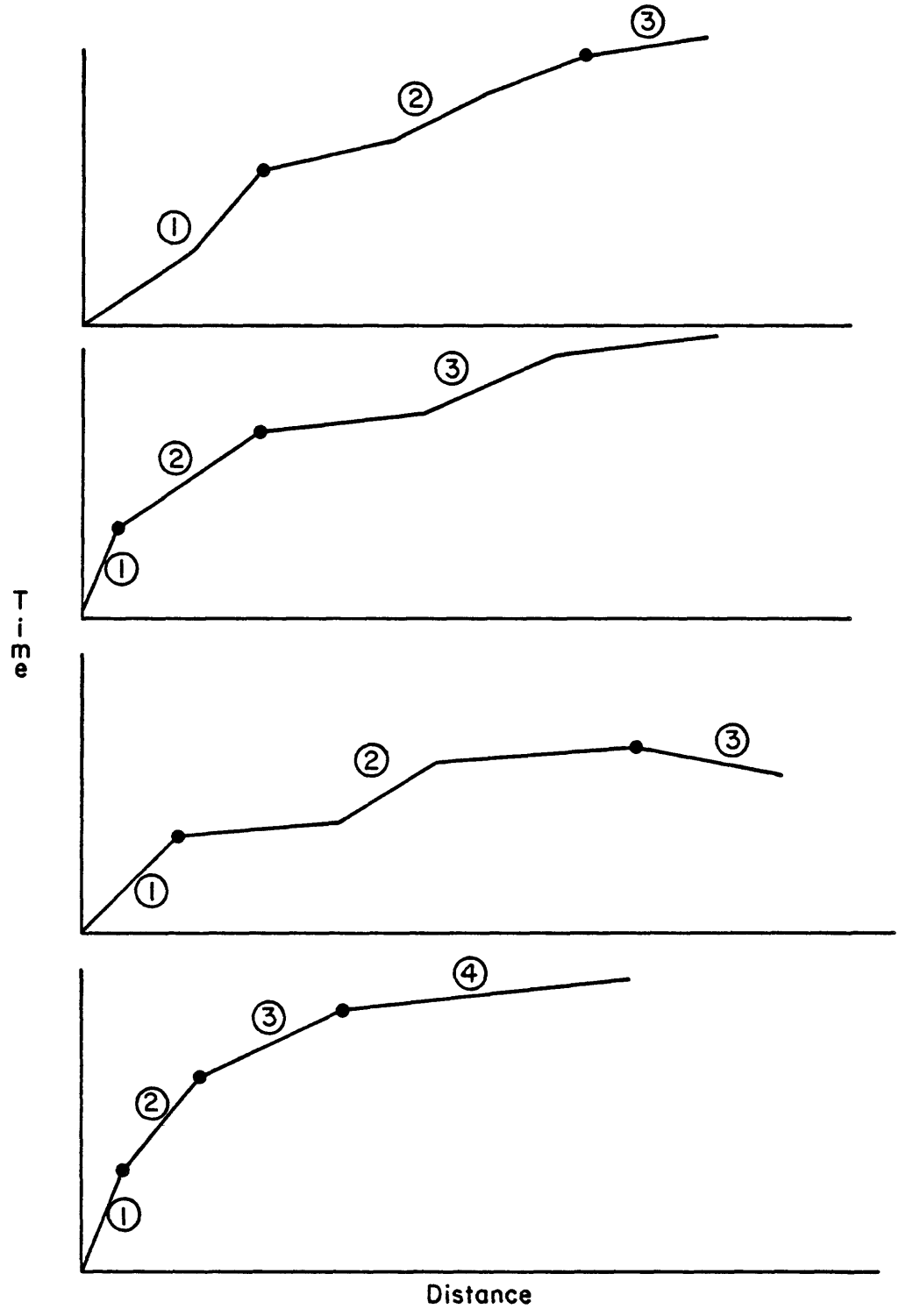

Figure 8.--Possible preliminary horizon identifications for a variety of hypothetical arrival-time curves represented by straight line segments.....

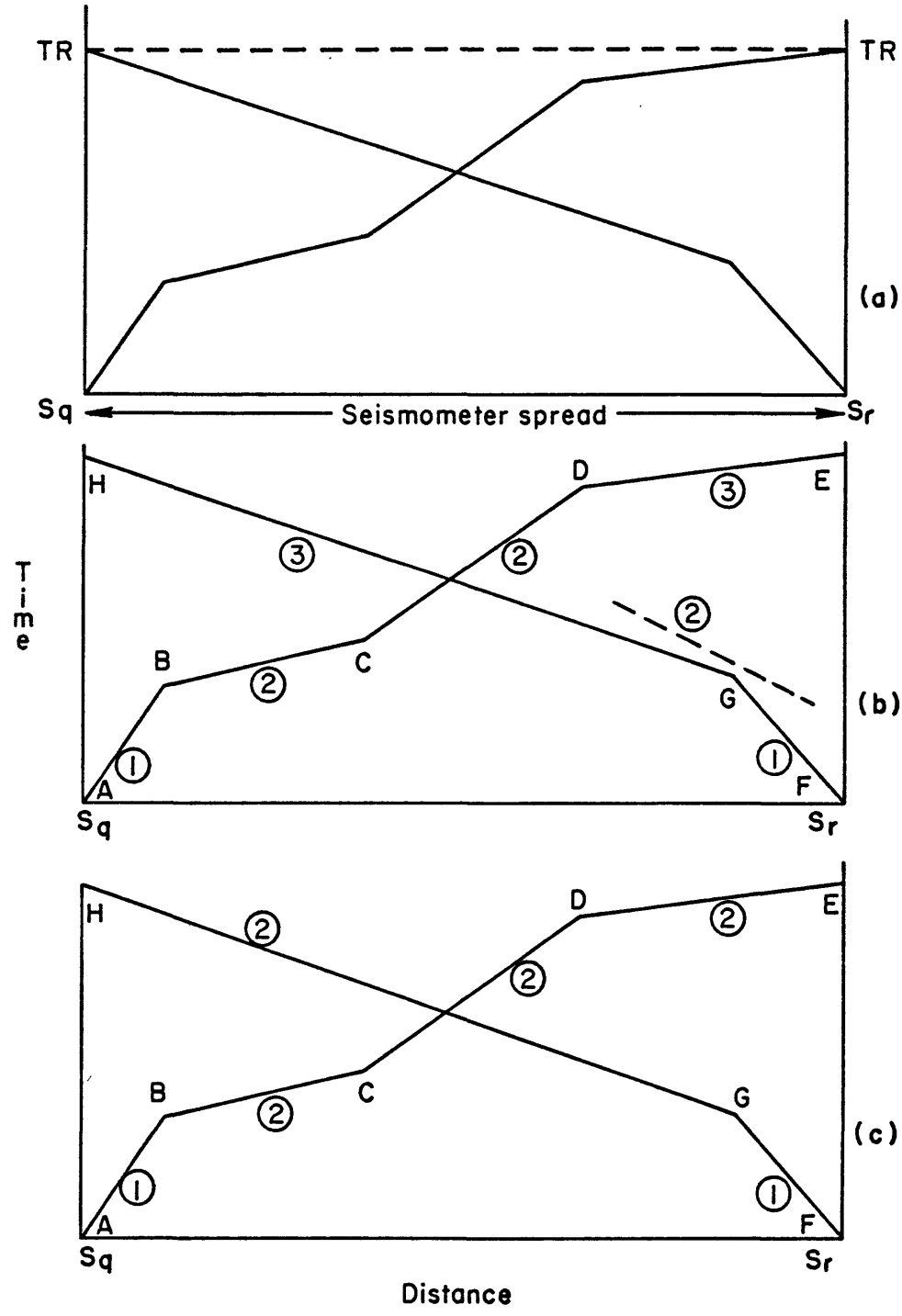

Figure 9.--Two possible horizon identifications (b) and (c), of the reversed arrival-time graph of (a). For the 3-layer identification (b), arrivals representing a hidden layer are sketched in as a dashed line segment.....

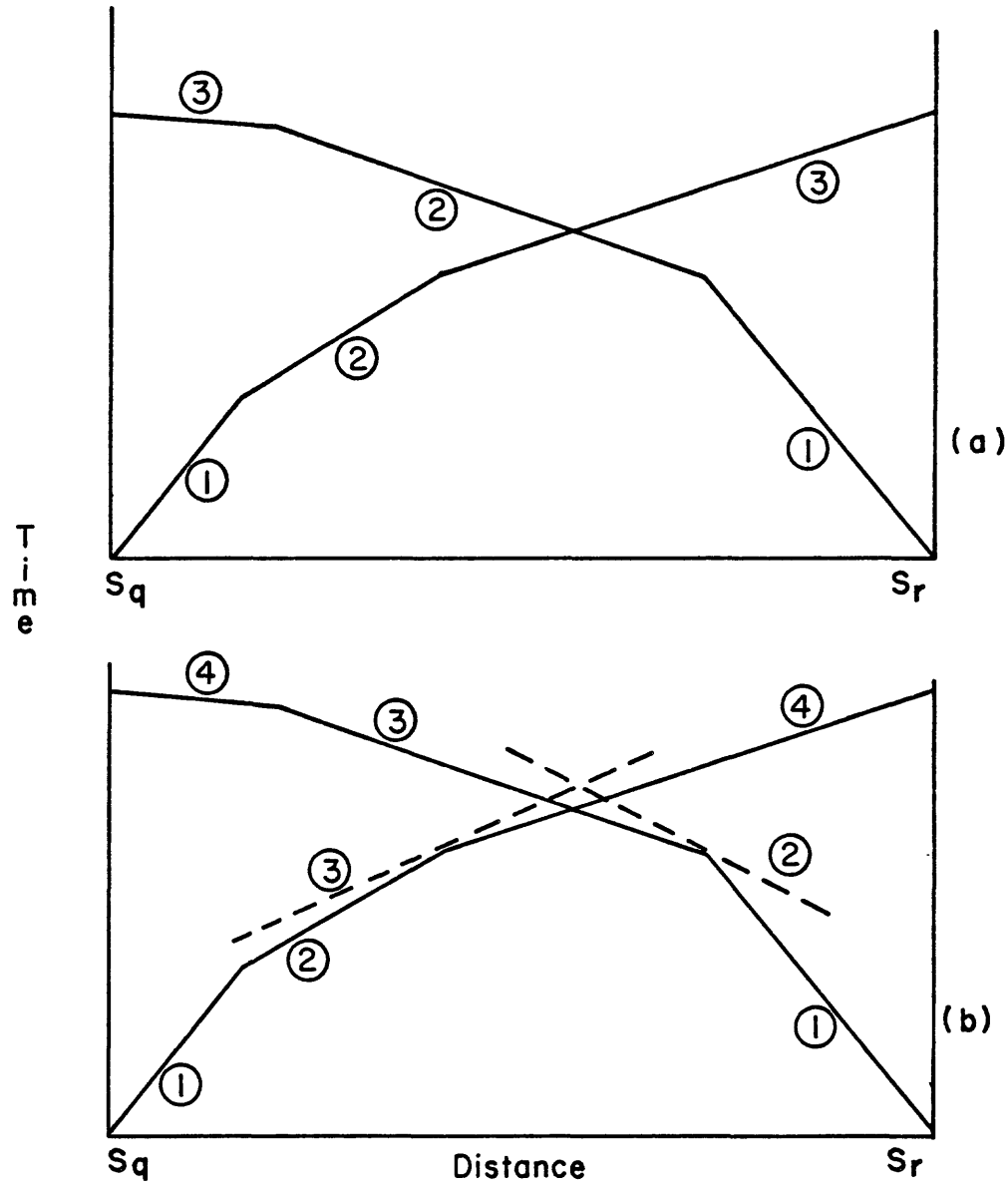

Figure 10.--Two possible horizon identifications of the same reversed arrival-time graph, one of which (b) contains hidden layers.....

assumed to be hidden and may be tentatively sketched in, as shown by the dashed line segment in figure 9b. Another solution could involve two layers (fig. 9c). Segments BC, CD and DE could all represent horizon 2; in which case segment GH in the reversed plot must also represent horizon 2. A one-layer solution is impossible because this would demand that apparent velocities for the reversed plots be equal at corresponding locations along the distance axis which clearly is not the case.

Another example is shown in figure 10. Although the graph may suggest a simple 3-layer solution, such a solution due to the lackof symmetry, would require a highly complex velocity section. (Discussion of such complexities is in the section on extending plots). If however we assume a 4-layer case (figure 10b), the solution would be less complex. In this case, segment 3 is assumed to be hidden for shotpoint S_q and segment 2 for shotpoint S_r.

Opposed plots

Opposed plots have common shotpoint locations but are recorded in opposite directions by different (opposed) spreads (fig. 11a). It can be demonstrated that for opposed plots intercept times (t_i) from common horizons must agree. Intercept times can only be obtained by projecting or extending arrival-time segments back to the time axis at the shotpoint.

The examples of figure 11 show opposed plots, one member consisting of three line segments and the other of four. Identification in terms of four horizons (fig. 11b) demands that horizon 2 on the left hand plot be hidden. Its arrival-time curve has been constructed to meet at the intercept (t_2) of horizon 2 from the right hand plot. Another possible, though perhaps less probable identification in terms of three horizons, is shown in figure 11c. In this case, it is assumed that the velocity of the surface layer changes at dashed line A.

Projections to the intercept point, of course, need not be straight lines, just as first-arrival-time segments from a given horizon need not be straight. Figure 12a is an example of opposed plots consisting of curved line segments. A 3-layer identification is shown in figure 12b, in which corresponding segments are projected as curved lines to intersect at the time axis. A 4-layer identification, using hidden layers, is shown in figure 12c.

Complementary plots

Complementary plots have common seismometer locations and in-line shotpoints recorded from different distances on the same side of the spread (fig. 13).

A property of complementary plots is that the slope of the arrival-time curve from the shotpoint more distant from the seismometer spread (S_r in figure 13) may be either less than (higher apparent velocity), or equal to, but according to ray theory, never greater than (lower apparent velocity) that for the nearer shotpoint. (Cases of higher apparent velocity for the nearer shotpoint do at times occur in cases of near-surface thin high-velocity beds, such as frost, pavement, or thin volcanic flows.) Equal slopes indicate that arrivals for both plots originate from the same

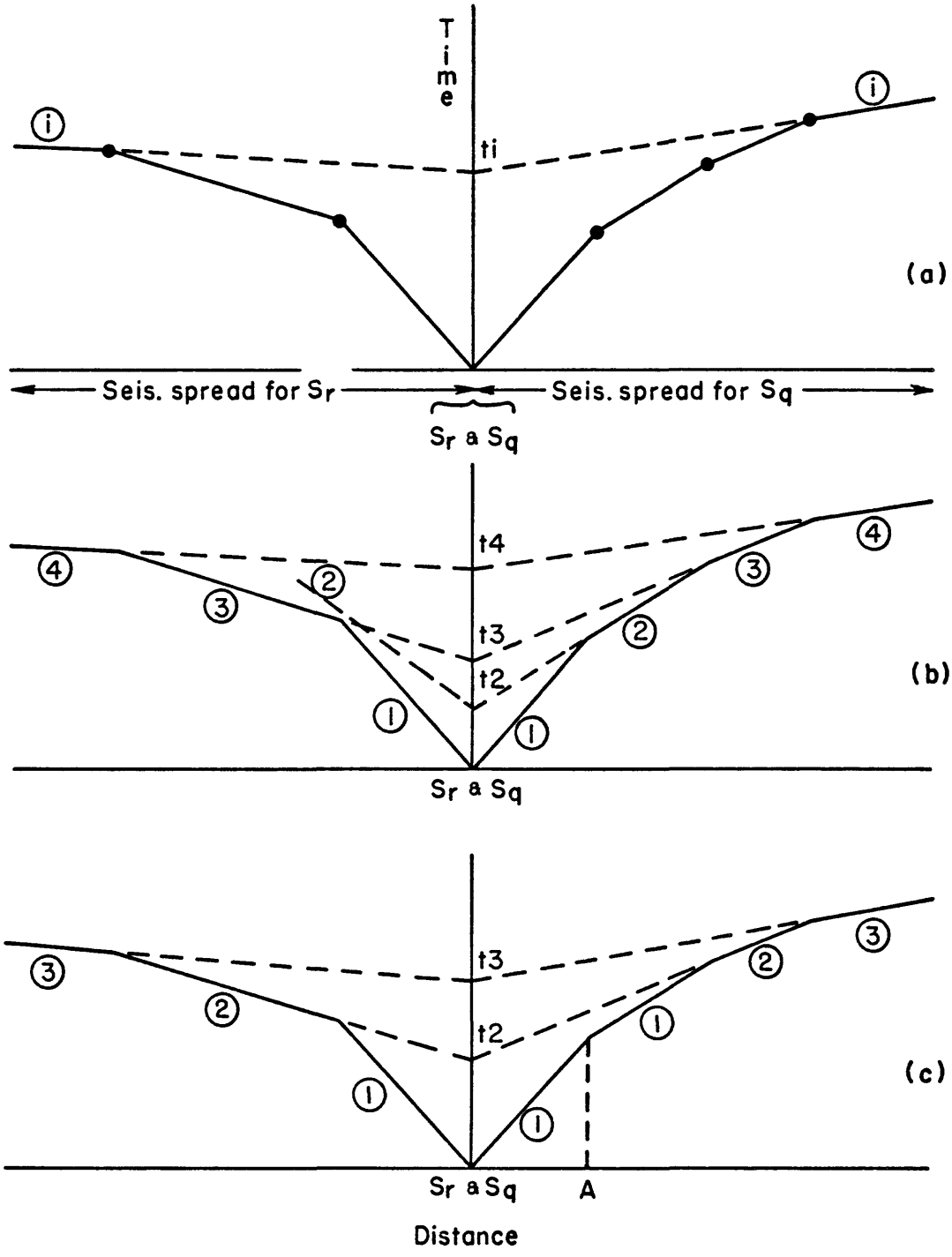

Figure 11.--Identical opposed arrival-time curves identified in terms of four horizons (b) and three horizons (c).....

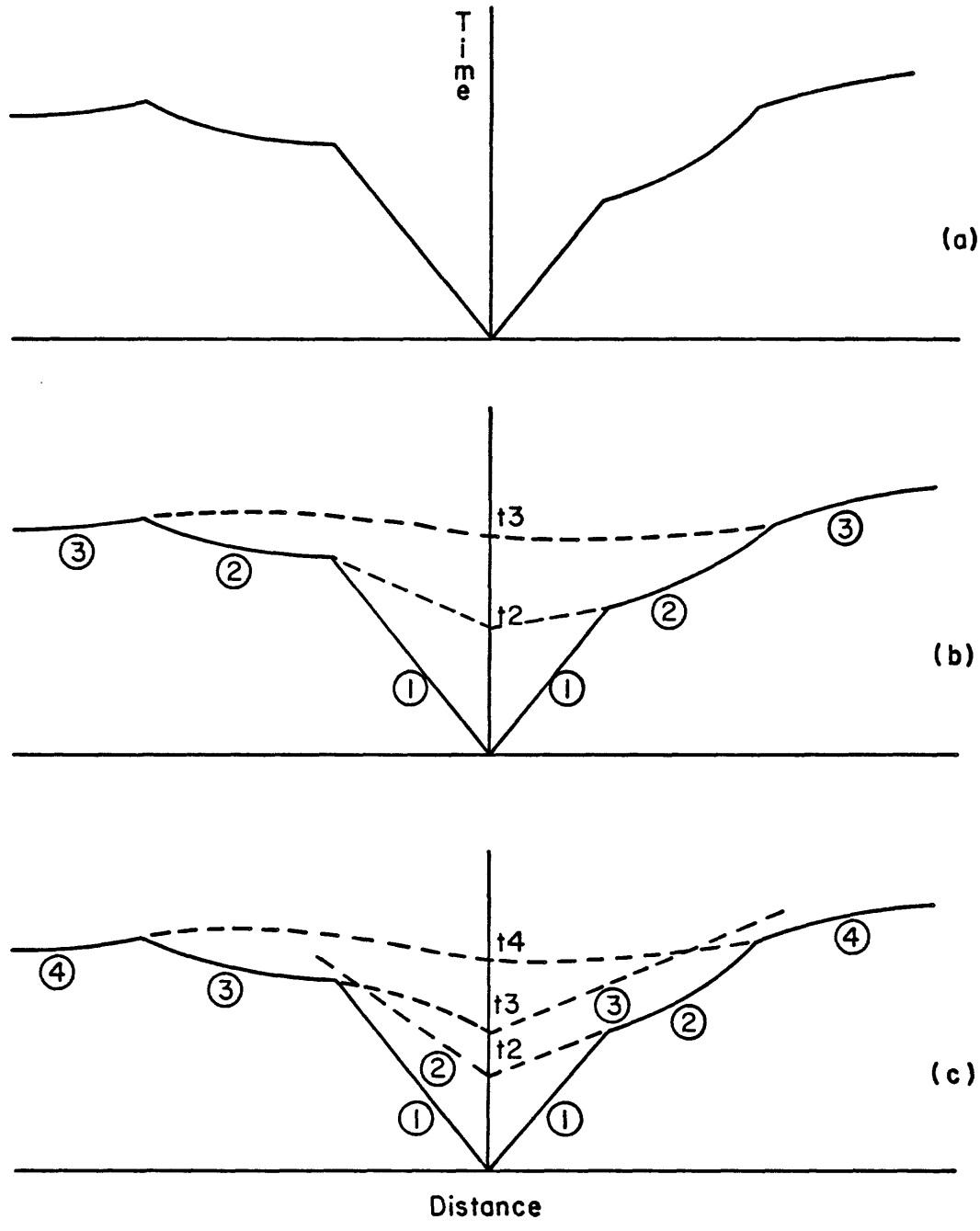

Figure 12.--Identical opposed arrival-time curves which are not straight line segments identified in terms of three horizons (b) and four horizons (c).....

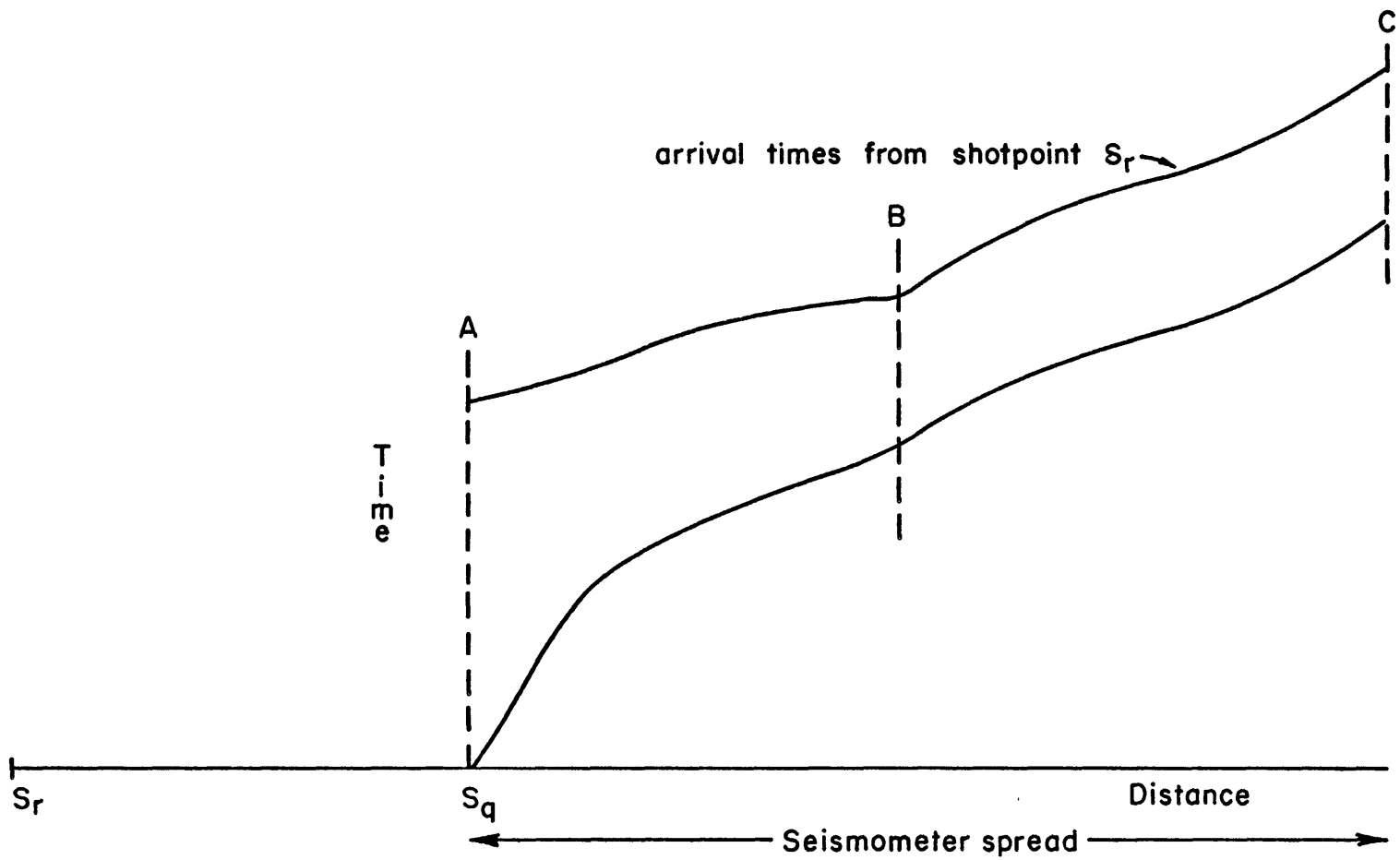

Figure 13.--An example of complementary arrival-time curves recorded from shotpoints Sq and Sr.....

horizon. Different slopes indicate that arrivals from the nearer shotpoint originate from a shallower horizon than the more distant shotpoint. Thus, in figure 13, arrivals for both plots originate from the same horizon in the interval BC where the curves are parallel. In the interval AB, arrivals from the more distant shotpoint originate from a deeper horizon. The point B therefore represents a crossover point for the nearer shotpoint, S_q. (Note that in this illustration the crossover is marked by a decrease in apparent velocity.)

The principal use of complementary plots is to distinguish between inflections due to crossover points from those due to lateral geologic changes. Crossover points (with rare exception) move with the shotpoint, whereas inflections due to lateral changes remain stationary on the time-distance graph. This is illustrated in figure 14, which represents arrival-time curves for a 3-layer problem involving five shotpoints, S_1 through S_5, with a change in dip of horizon 3 between S_1 and S_2. The individual segments on the arrival-time curves have been labeled to represent corresponding horizons from which first arrivals originate. We note that the inflection point between horizons 1 and 2 moves from point A_1 through A_5 as the shotpoint moves from location S_1 through S_5 and is therefore a crossover point. (For shotpoint S_1 the crossover points A_1 and B_1 are hidden.) The inflection point representing horizons 2 and 3 similarly moves from point B_1 through B_5 and is again a crossover point. The intervals between successive crossover points are not constant due to the effect of the change in dip of horizon 3. Point C on the distance axis marks the location at which the effect of the change in slope of horizon 3 appears on the first-arrival plots, and this point stays fixed for all shotpoints. Thus, for the case of shotpoint S_3, although initial inspection may suggest that the plot represents four layers, inspection of the complementary plots S_4 and S_5 indicates that this inflection is due to a lateral geologic change.

The use of complementary plots is a powerful tool for delineating between crossover points and lateral changes. They are most effective for horizons represented by a "long" segment of an arrival-time curve. For relatively short segments, the method becomes impractical because then many closely spaced shotpoints are required.

In most applications of seismic-refraction techniques, inflections that appear on time-distance graphs can be related to subsurface geology through the judicious application of reversed, opposed and complementary plots.

Extending arrival-time data

As discussed earlier, arrival-times from each horizon must usually be extended back to the time axis and forward to the reciprocal point in order to help determine a unique solution.

Extending arrival-time curves back to the intercept point

Without other constraining data, a wide degree of latitude exists for extending data back to the intercept. Take for an example, the 3-layer case illustrated in figure 15. Within limits, segment 2 may be projected back to

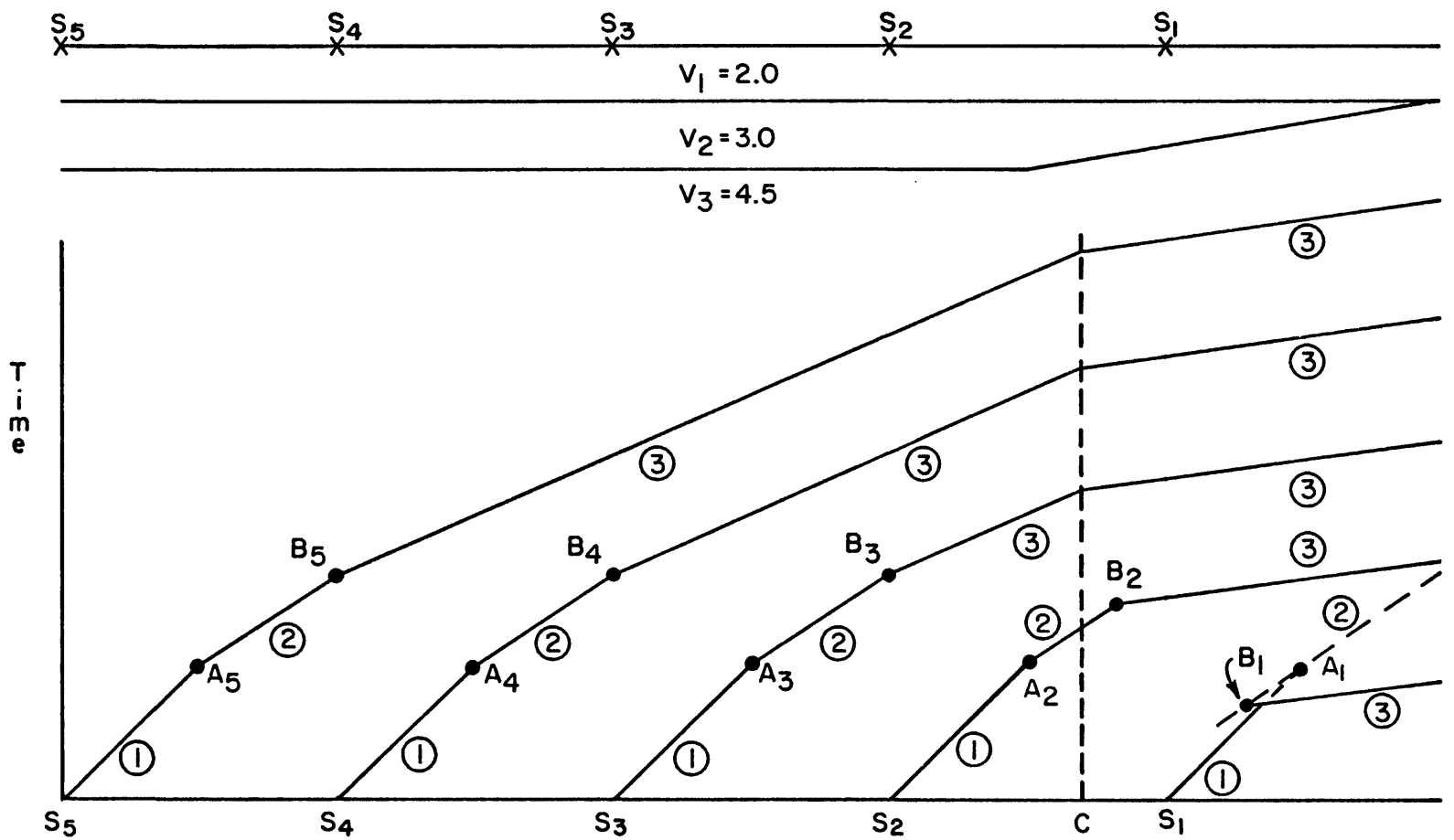

Figure 14.--Sketch illustrating that crossover points A_1 through A_5 and B_1 through B_5 move with their corresponding shotpoints S_1 through S_5, and that the effect of a lateral geologic change remains stationary at point C in the arrival-time curve.....

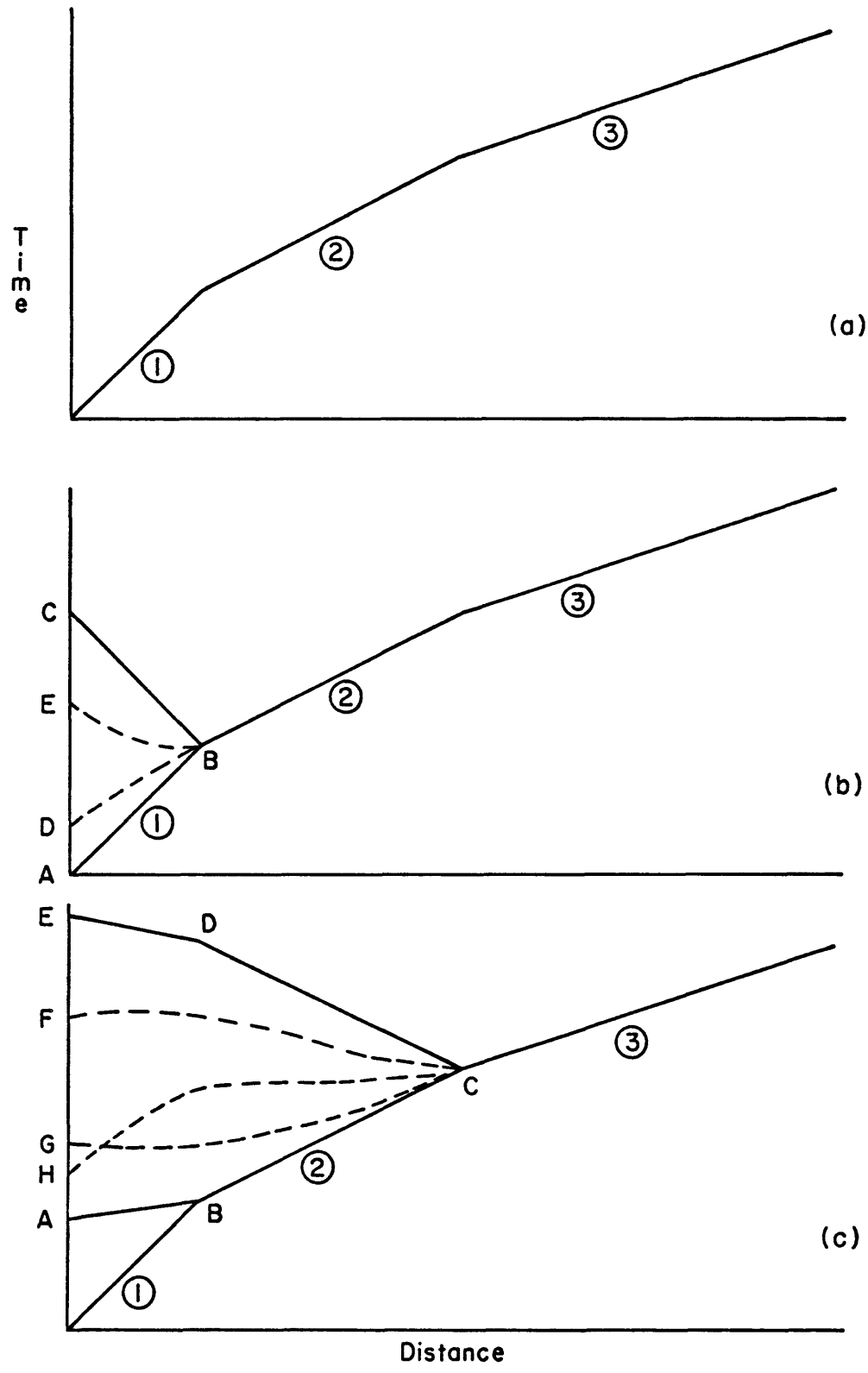

Figure 15.--Sketch illustrating valid and invalid extensions of arrival-time curves to the intercept. BE and BD are valid for horizon two (b) and CF and CH for horizon three (c). Extension CH for horizon three is invalid.....

the intercept as desired, provided the absolute value of the slope at corresponding points along the distance axis of the projected curve does not exceed that of the segment representing the next shallower horizon. Thus, in figure 15b, the projected arrival for horizon 2 is constrained to lie within triangle ABC. Projections such as BD and BE are both possible. In projecting segment 3 to the intercept, the projection for segment 2 must first be completed. If we assume that BA (fig. 15c) is the correct projection for horizon 2, then the projection for horizon 3 must lie within the figure ABCDE such that the slope requirements stated above are again satisfied. Projections CF and CG are possible whereas projection CH is not because the slope of segment BA is exceeded. It should be noted that projecting data near the limits of validity is generally unrealistic unless clearly substantiated from other data, and may result in computation problems.

The constraints on extending arrivals to the intercept are obtained from complementary and opposed plots.

The use of complementary plots may provide a guide for projecting arrivals, as shown in figure 16.In this case, the arrivals from horizon 3 of shotpoint S_r are extended to the intercept from point C by drawing a curve parallel to EF derived from offset shotpoint S_q.

The use of opposed plots constrains the extension of arrivals by demanding that intercept times agree, as shown in figure 17a. Although a degree of freedom exists in extending arrivals from both plots to the intercept, they must intersect at a common point, such as D. If the extension of one of the plots has already been completely constrained by complementary data (fig. 17b), the extension of the other must then be constrained to intersect at the intercept. If the extension of both opposed plots are completely constrained by complementary data (fig. 17c), their extensions will, by the laws of refraction, intersect at the intercept. If not, an error has occurred in segment identification.

Extending arrival-time curves forward to the reciprocal point

The forward extension of data usually implies the existence of a reciprocal plot in which case constraints are imposed by reciprocity. Complementary plots may provide further constraints.

Reciprocity demands that reciprocal times from all horizons agree. Figure 18a illustrates the forward extension of data from horizons labeled 2 and 3 in such a way that reciprocity is satisfied. However, forward extension of line segments for horizon 4 without change of slope would violate reciprocity. In the absence of constraints, there are an infinite number of ways for forward extension of arrivals. However, as with extending data back to the intercept, the slope of an extended segment must be greater than that representing the next deeper horizon and less than that for the next shallower.

Figure 18b is an example involving plots from three shotpoints labeled S_1, S_2 and S_3. The addition of one shotpoint in the center of the seismometer spread severely constrains the extensions for horizon 2. The

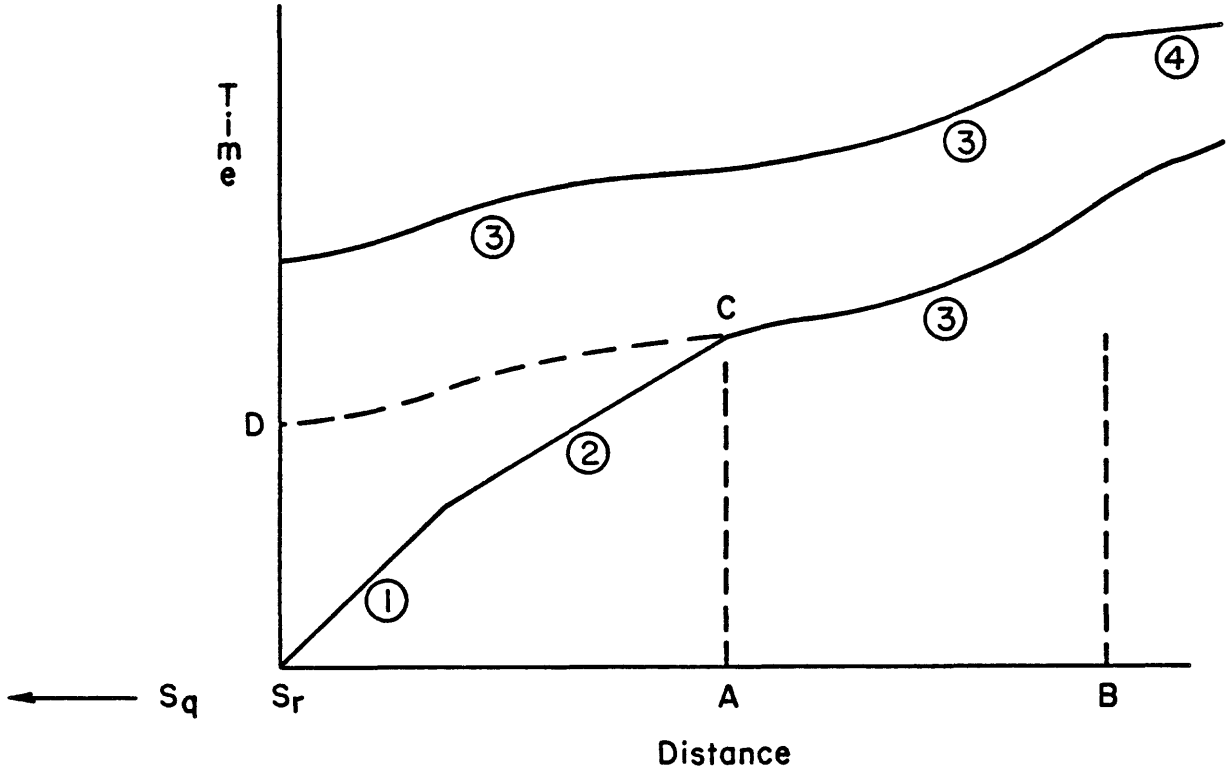

Figure 16.--Example illustrating extension for hozizon three to the intercept for shotpoint Sr as constrained by the complementary arrival-time curve for shotpoint Sq.....

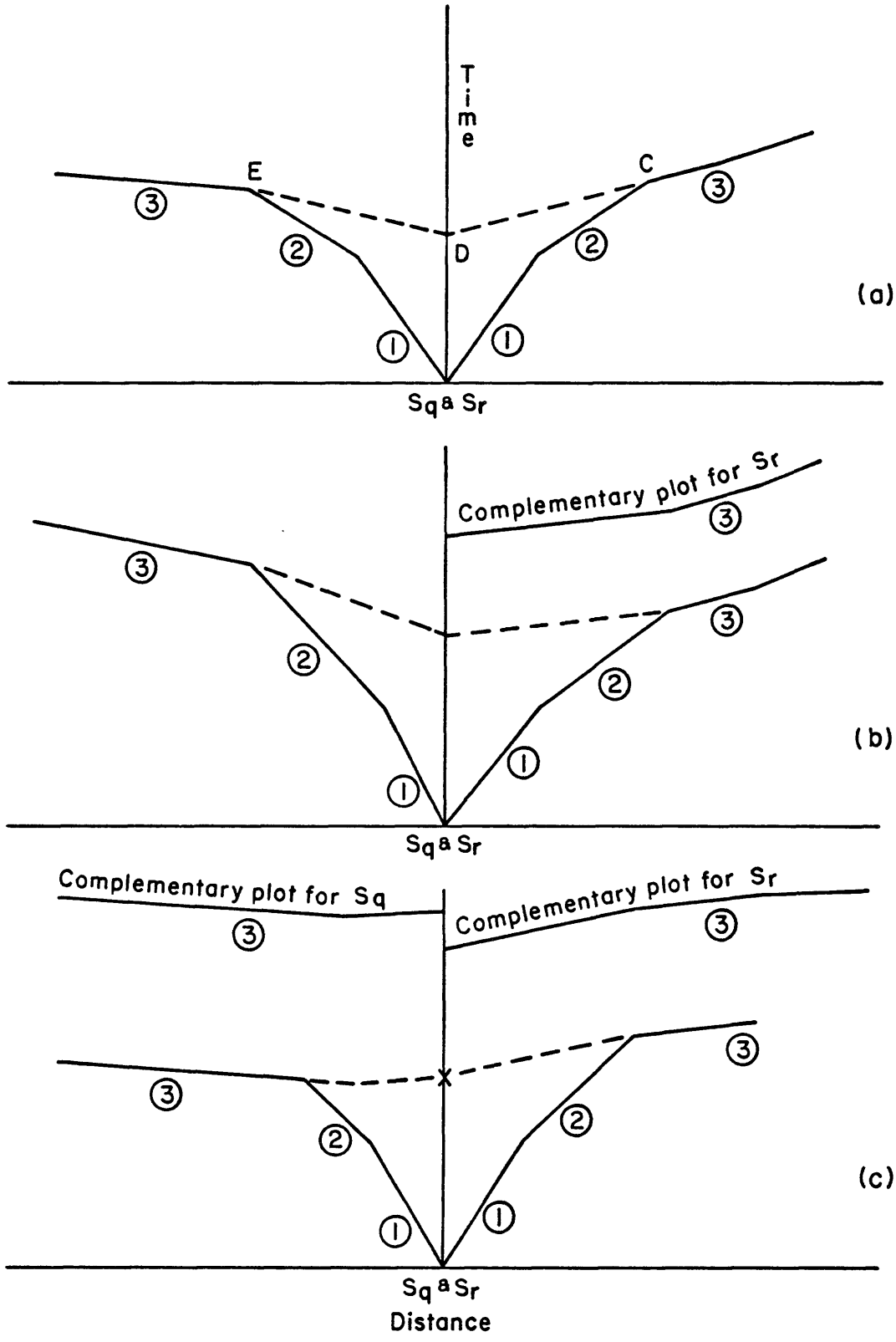

Figure 17.--Example illustrating extensions for horizon three to the intercept as constrained by opposed and complementary arrival-time curves.....

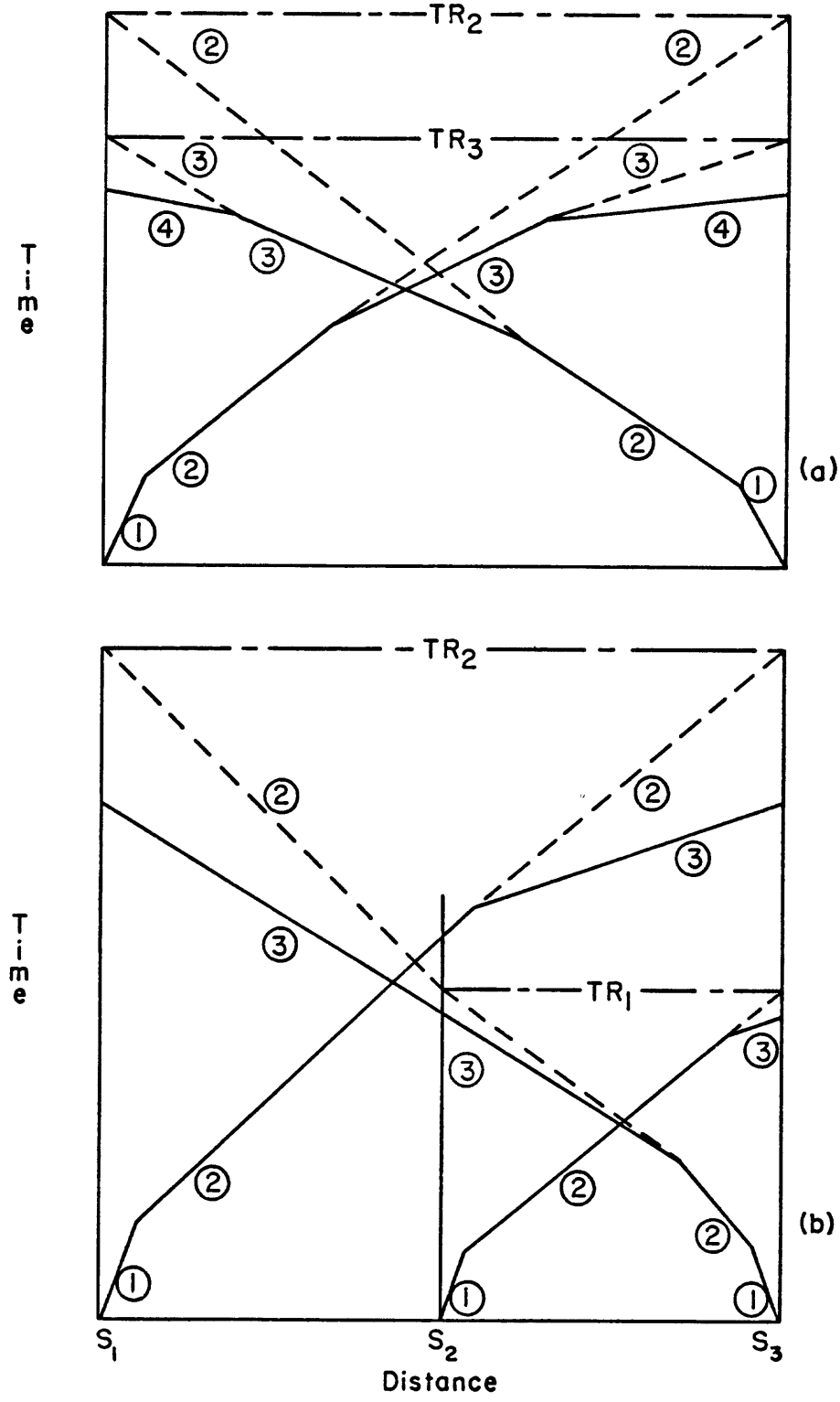

Figure 18.--Example (a) illustrating the forward extension of arrival-time curves using the constraint of reciprocity. Example (b) illustrates forward extension constrained by both reciprocity and complementary arrival-time curves.....

graph now consists of two reversed plots, one between S_2 and S_3 and other between S_1 and S_3, and one complementary plot involving shotpoints S_1 and S_2. Between S_2 and S_3, the line segments for horizon 2, when extended forward to the time axis provide a reciprocal time TR_1. The arrival-time curve from horizon 2 for shotpoint S_1 can then be extended forward to its reciprocal point by constructing a line parallel to that representing horizon 2 from complementary shotpoint S_2, resulting in a reciprocal time TR_2. The line segment representing horizon 2 for shotpoint S_3 is then constrained to intersect the time axis at TR_2 as shown.

An example

Figure 19 shows a hypothetical t-x plot for a problem which can beinterpreted in terms of 5 layers. The solid lines represent first-arrival-times and the dashed lines are data extensions as discussed below. This plot consists of two end-to-end spreads labeled spread I and spread II, each 6000 distance units long. Each spread is recorded from shotpoints shown as S_3 through S_7. In addition, both spreads are recorded from shotpoints S_2 and S_8 at distance coordinates -3000 and 15000 respectively and spread I was recorded from a shotpoint at -6000 and spread II at 20000. These shotpoint locations were picked to conveniently serve the hypothetical velocity model (fig. 20) from which figure 19 was derived. In order to simplify later discussion, spreads I and II have been subdivided into the intervals Ia, Ib, IIa, and IIb as shown.

We first note, in figure 19, that reciprocity is satisfied. At S_4 the reciprocal values, read vertically on the graph, of approximately 1190, 1260, 2060 and 2550. These times agree with the times from shotpoint S_4 recorded at shotpoint locations S_3, S_5, S_6 and S_7 respectively. Segments between crossover points, representing specific horizons, are labeled and crossover points are circled. Usually this labeling process proceeds as data are extended beginning with the shallow layers. The process may at times become a complex problem of changing segment labels and introducing hidden layers in order to maintain internal consistency. At distance 7750, for example, a distinct inflection appears on the graphs from shotpoints S_2 and S_3. However, because the two curves are parallel on both the left and right sides of the inflection point, they represent the same horizon (5). This suggests that the inflection is due to a lateral effect rather than the onset of arrivals from a deeper layer. At distance 4650 for S_7 and S_8, the curves are parallel to the left of the common inflection, but not on the right. It is concluded therefore that this distance marks a crossover point for shotpoint S_7, but not for S_8.

Normally each segment between inflections is also labeled with its corresponding approximate apparent velocity. Apparent velocities are extremely helpful in correlating horizons. However, to avoid cluttering the plot unnecessarily, only a few velocities have been posted.

Wherever calculations are to be made, data are extended back to the intercept if the the critical reflection method is to be used (subroutine CRIT) and forward to the reciprocal point and beyond for the reciprocal method (subroutine RECIP). For the reciprocal method, arrivals may also be extended back to the intercept, or beyond, whenever possible, in order to increase the coverage of calculation points.

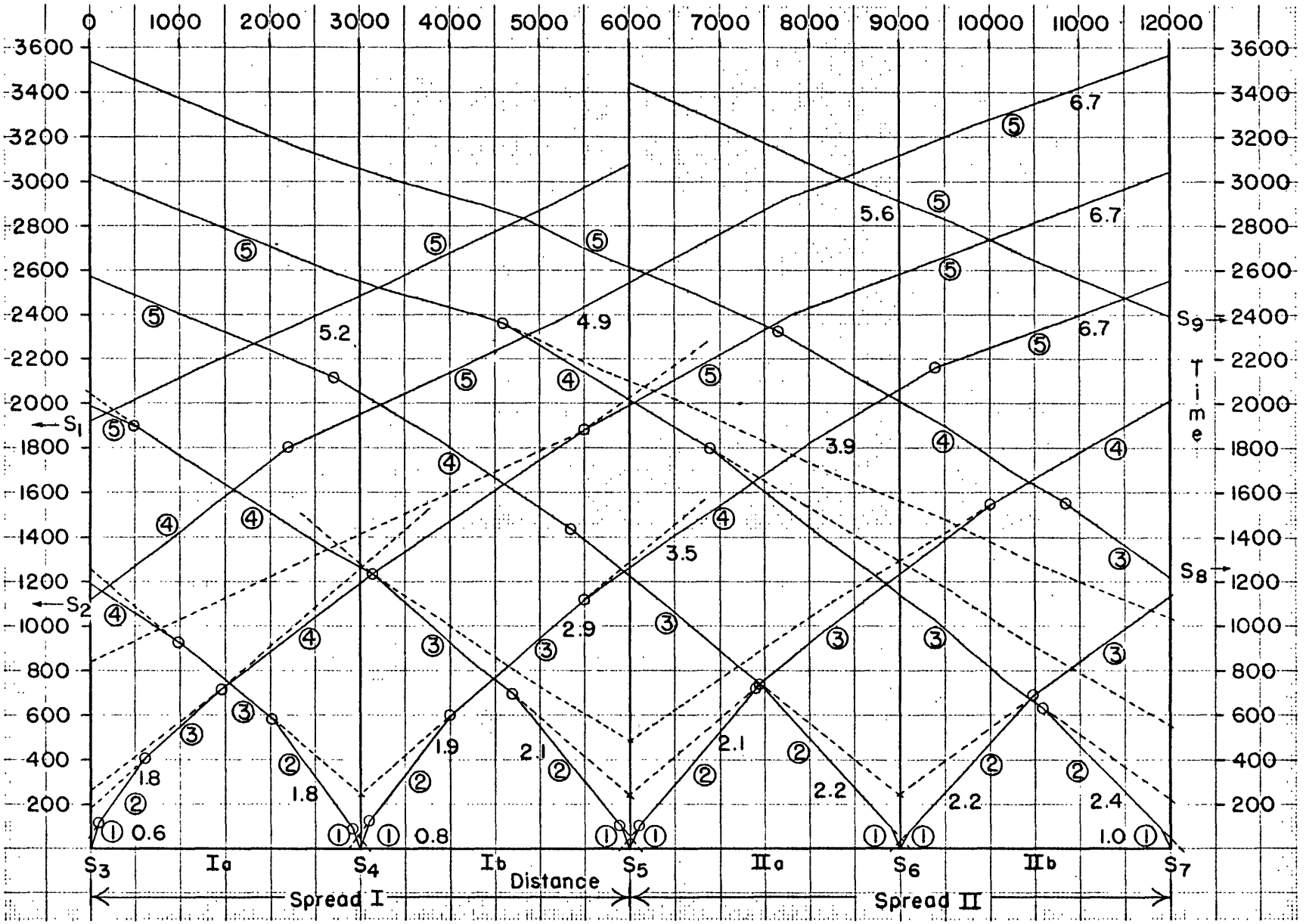

Figure 19.--Arrival-time curves, shown by solid lines, for the five-layer velocity model of Figure 20 with shotpoint locations S_1 through S_7. Arrival-time extensions are shown as the dashed lines.....

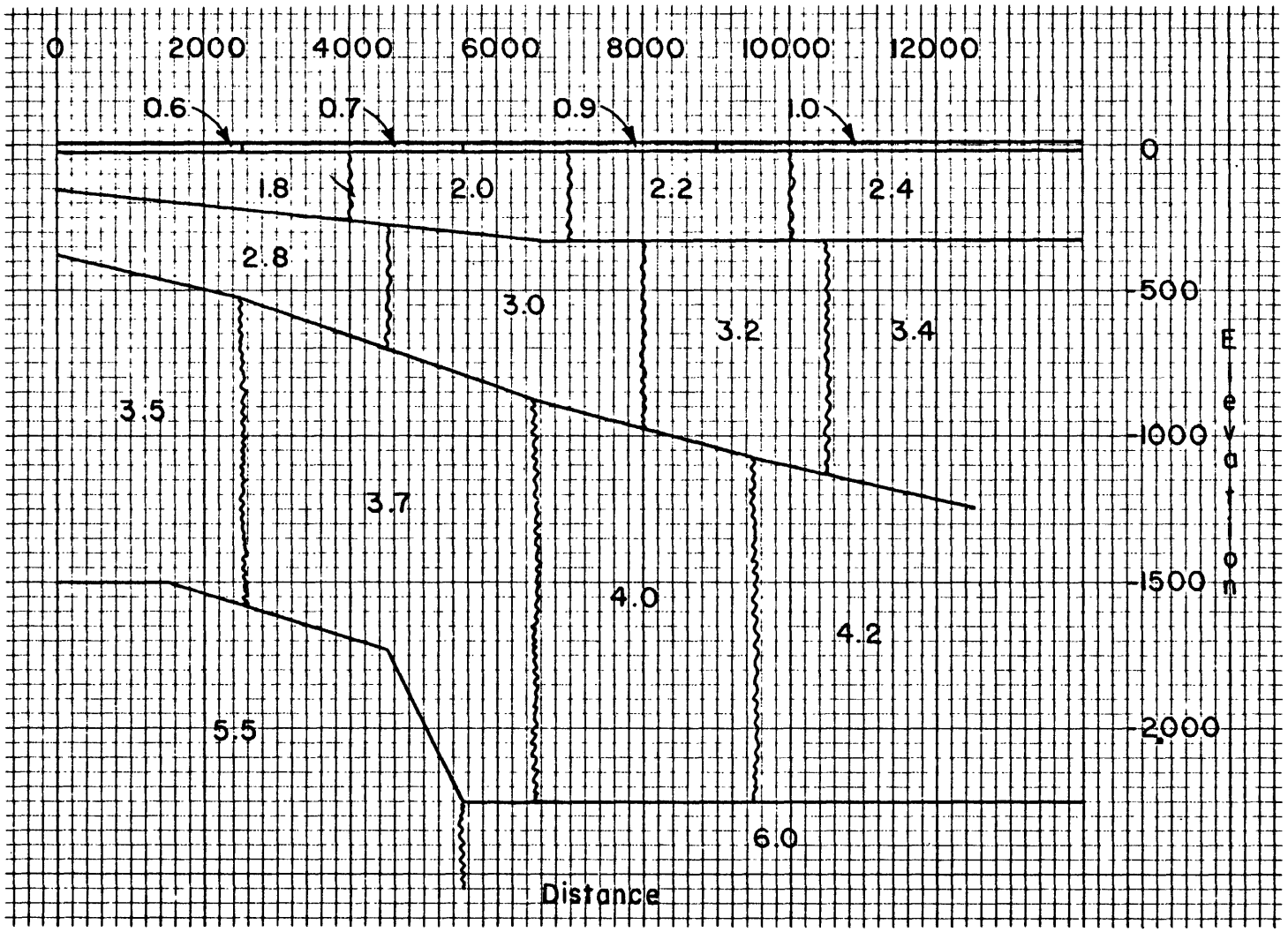

Figure 20.--Velocity model used for computing the arrival-time curves of Figure 19.....

Arrivals from horizon 1 are not extended. Their principal use is to obtain a velocity for the first layer. For this problem, the velocity of layer 1 was assumed to vary from 0.6 at S_3 to 1.0 at S_7. Data from horizon 2 are straightforward and limited to the vicinity of each shotpoint. Hence only a few depth calculations in the vicinity of each shotpoint can be calculated. Velocities in this layer appear to increase gradually from 1.8 at S_3 to 2.4 at S_7. Data for horizon 2 have therefore only been extended back to (and slightly beyond) the time intercept.

Data from horizon 3 are sufficient to allow point-by-point calculations beneath both spreads using reciprocal methods. For shotpoints S_3, S_4 and S_5, no complementary data are available for horizon 3; segments representing this horizon have been extended back to the time axis without any significant change in slope. For shotpoints S_6, complementary data from S_5 and S_7, and for S_7, complementary data from S_8 are used to control the extension of data to intersect at the intercept. In the intervals IIa and IIb, the data for horizon 3 already intersect the reciprocal points, eliminating the need to extend data forward. In the interval Ib, the data for horizon 3 are readily extended a few hundred distance units for a reciprocal time of about 1290. These data can be extended a bit further into interval Ia by using complementary data from shotpoint S_4 and into the interval IIa by using data from S_5. Similarly in the interval Ia, data from horizon 3 from shotpoints S_3 and S_4 are projected forward to a reciprocal time of about 1260, and the data from S_3 may be extended beyond S_4 using complimentary data from S_4. With these extensions, the data from horizon 3 form a completely internally consistent unit from which a set of unique depth and velocity calculations can be made.

Because longer data segments are available from horizon 4 than from horizon 3, the entire spread-length distances of spreads I and II are used for constructing reversed graphs rather than half-spread intervals as for horizon 3. For S_3, data are extended to the intercept for horizon 4 using complementary data from S_2 for control. For S_5 (spread I) data are extended to the intercept using data from complementary shotpoint S_6 as far as distance coordinate 5350 and from S_7 for the remaining distance. For shotpoint S_5 (spread II) data are extended to the intercept using S_4 for complementary control and for shotpoint S_7, data from S_8 provide complementary control. For shotpoints S_5 and S_7 of Spread II, data from horizon 4 already extend to the reciprocal point or beyond and therefore need not be extended forward. For spread I, data from both shotpoints S_5 and S_3 are extended to the reciprocal point using shotpoint S_4 for control. With these extensions, horizon 4 is ready for computations.

For horizon 5, a single set of reversed graphs are constructed in the combined interval of spreads I and II. For shotpoint S_3, data are extended to the time axis at a time of about 840 using complementary data from shotpoint S_1. For S_7, complementary data are obtained from shotpoints S_8 and S_9, and the intercept is about 1035.

Thus, in making interpretations, horizon 2 will be calculated using subroutine CRIT at individual shotpoints S_3 through S_7. Horizon 3 will be

```
pr play1.lay

                    play1.lay

 1  5  -90000.00        0.00         0.60
   2500.00         0.00         0.70
   5500.00         0.00         0.90
   9000.00         0.00         1.00
  90000.00         0.00         1.00
 2  5  -90000.00      -20.00         1.80
   4000.00       -20.00         2.00
   7000.00       -20.00         2.20
  10000.00       -20.00         2.40
  90000.00       -20.00         2.40
 3  7  -90000.00     -150.00         2.80
      0.00      -150.00         2.80
   4500.00      -275.00         3.00
   6300.00      -325.00         3.00
   8000.00      -325.00         3.20
  10500.00      -325.00         3.40
  90000.00      -325.00         3.40
 4  7  -90000.00     -325.00         3.50
      0.00      -325.00         3.50
   2500.00      -527.00         3.70
   6500.00      -875.00         4.00
   9500.00     -1075.00         4.20
  12500.00     -1250.00         4.20
  90000.00     -1250.00         4.20
 5  5  -90000.00    -1500.00         5.50
   2500.00     -1500.00         5.50
   4500.00     -1750.00         6.00
   5500.00     -2250.00         6.00
  90000.00     -2250.00         6.00
```

Table 1.--Digital representation of velocity model of Figure 20 in file play1.lay.....

calculated with subroutine RECIP within individual intervals Ia, Ib, IIa and IIb. Horizon 4 will be calculated using RECIP within the individual intervals of spread I and spread II, and horizon 5 within the combined interval of spreads I and II. Clearly, other constructions could be made to make calculations within different intervals. However, because the plots are already completely internally consistent, all other combinations should give the same solutions In practice, redundant combinations are often used to provide internal checks.

The data of figure 19 were computed from the model shown in figure 20. The layers in digital format (as described later) are given in table 1. The user may interpret the data of figure 19 using the computer methods to be described later and compare results with the initial model. Reading and plotting errors of roughly 5 to 10 time units should be considered in comparing results.

THEORY

In order to calculate the configuration of an (n+1)th horizon from its associated arrival-time curves, a model (e.g. velocity distribution) for the overlying n layers must be assumed. This distribution may have been calculated using refraction arrivals from the shallower horizons or from well data.

The Model

For these programs, the n overlying layers are assumed to be discrete and of variable thickness and velocity. The upper boundary of each layer referred to as a horizon, so that horizon 1 represents the ground surface. Ideally, the velocity of a layer should be expressable as a function of x and y, $v = v(x,y)$; where x equals the horizontal distance and y the elevation above some arbitrary datum. In this way, lateral and vertical velocity variations can be expressed mathematically. In the method outlined in this paper, however, each horizon is represented by connected straight line segments and each layer is subdivided into compartments with vertical boundaries (fig. 21). The velocity within a compartment is assumed to be constant and a compartment boundary marks a change in velocity and/or a change in slope.

The arrival-time curve

The arrival-time plots for the (n+1)th horizon consist of a series of points expressed in time and distance (t vs x) using the same coordinates of x as for the velocity model described previously. An arrival-time curve is obtained by approximating these points by a series of straight line segments; constructed either by hand, in a least squares sense, using an arbitrary number of points for each segment (fig. 22), or by a cubic spline (Anderson, 1971). In either case, the value of t, and the derivative, dt/dx, are readily calculable from the arrival-time curve for any given x. (We have found that straight line approximations are generally more satisfactory because of problems with derivatives using splines.)

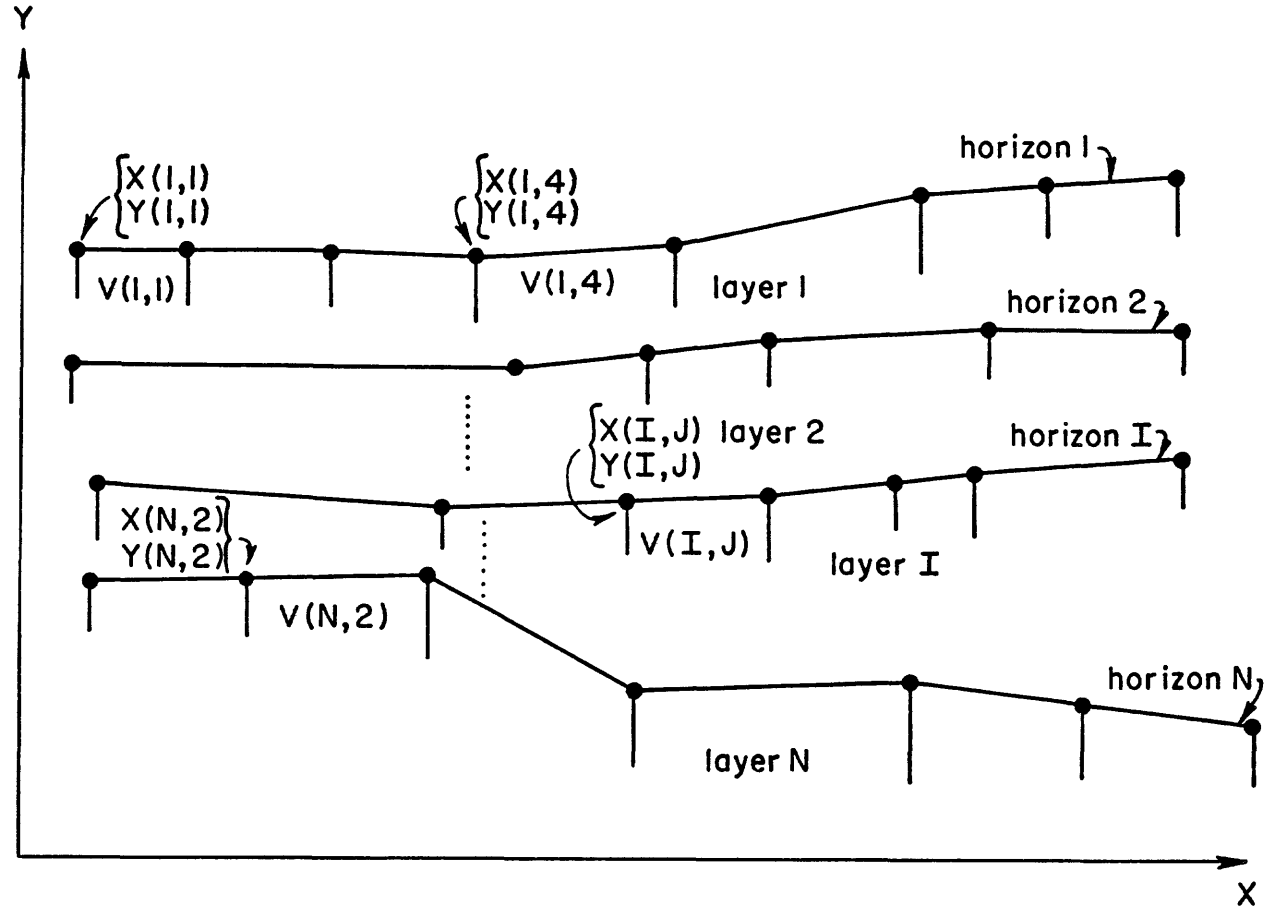

Figure 21.--Graphic representation of a velocity model of N layers. Horizons are represented by connected straight line segments. Vertical lines represent compartment boundaries.....

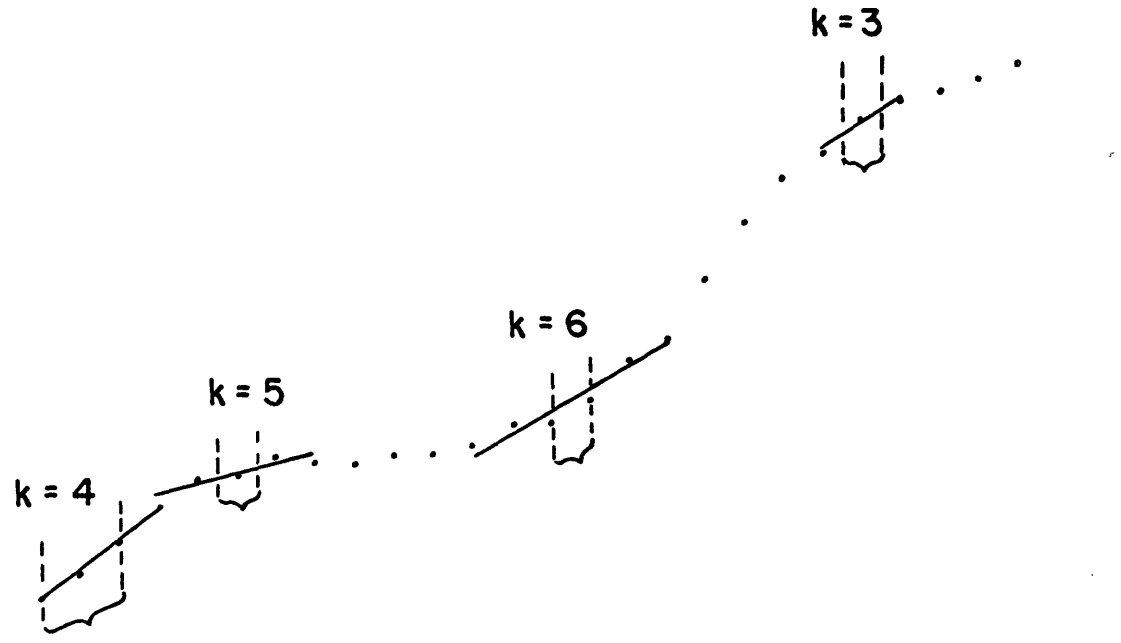

Figure 22.—Representation of a set of arrival-time points are represented by least square straight line segments. The parameter k, represents the number of points used.....

Reduction of arrival-time curves

Using the notation in figure 21, the equation of the Ith horizon within its Jth compartment (fig. 23) is

$$\frac{y - Y(I,J)}{x - X(I,J)} = \frac{Y(I,J+1) - Y(I,J)}{X(I,J+1) - X(I,J)} = M(I,J) \qquad (1)$$

and the expression for y in terms of x is

$$y = Y(I,J) + M(I,J)\,(x - X(I,J)). \qquad (2)$$

The expression for the dip of the horizon is

$$\Theta(I,J) = \arctan M(I,J). \qquad (3)$$

Thus for any given distance xl_k (fig. 23) representing the kth point on the Ith horizon and in the Jth compartment, its corresponding elevation yl_k is

$$yl_k = Y(I,J) + M(I,J)\,(xl_k - X(I,J)) \qquad (4)$$

The dip of the Ith horizon at point k is

$$\theta l_k = \Theta(I,J), \qquad (5)$$

and the velocity of the Ith layer below point k is

$$vl_k = V(I,J) \qquad (6)$$

The emergence angle (Slotnick, 1959, p 3) αl_k (fig. 23) is defined as the angle an emerging ray makes with the perpendicular to the horizon at the point of emergence (xl_k, yl_k).

Given an arrival-time curve (t vs x) defined on the Ith horizon, as shown in figure 23, the emergence angle may be calculated from the derivative of the arrival-time curve and the layer velocity at the point of emergence (Slotnick, 1959, page 9). At the point (xl_k, yl_k) the emergence angle αl_k is given by

$$\sin \alpha l_k = vl_k \left.\frac{dt}{dx}\right|_{x=xl_k} \qquad (7)$$

or solving for αl_k,

$$\alpha l_k = \arcsin \left(vl_k \left.\frac{dt}{dx}\right|_{x=xl_k}\right) \qquad (8)$$

The value of $\left.\frac{dt}{dx}\right|_{x=xl_k}$ is the inverse of the apparent velocity vl'_k of the arrival-time curve at $x=xlk$, and equation (8) in more familiar form would be

$$\alpha l_k = \arcsin \left(\frac{vl_k}{vl'_k}\right)$$

where $vl_k \leq vl'_k$.

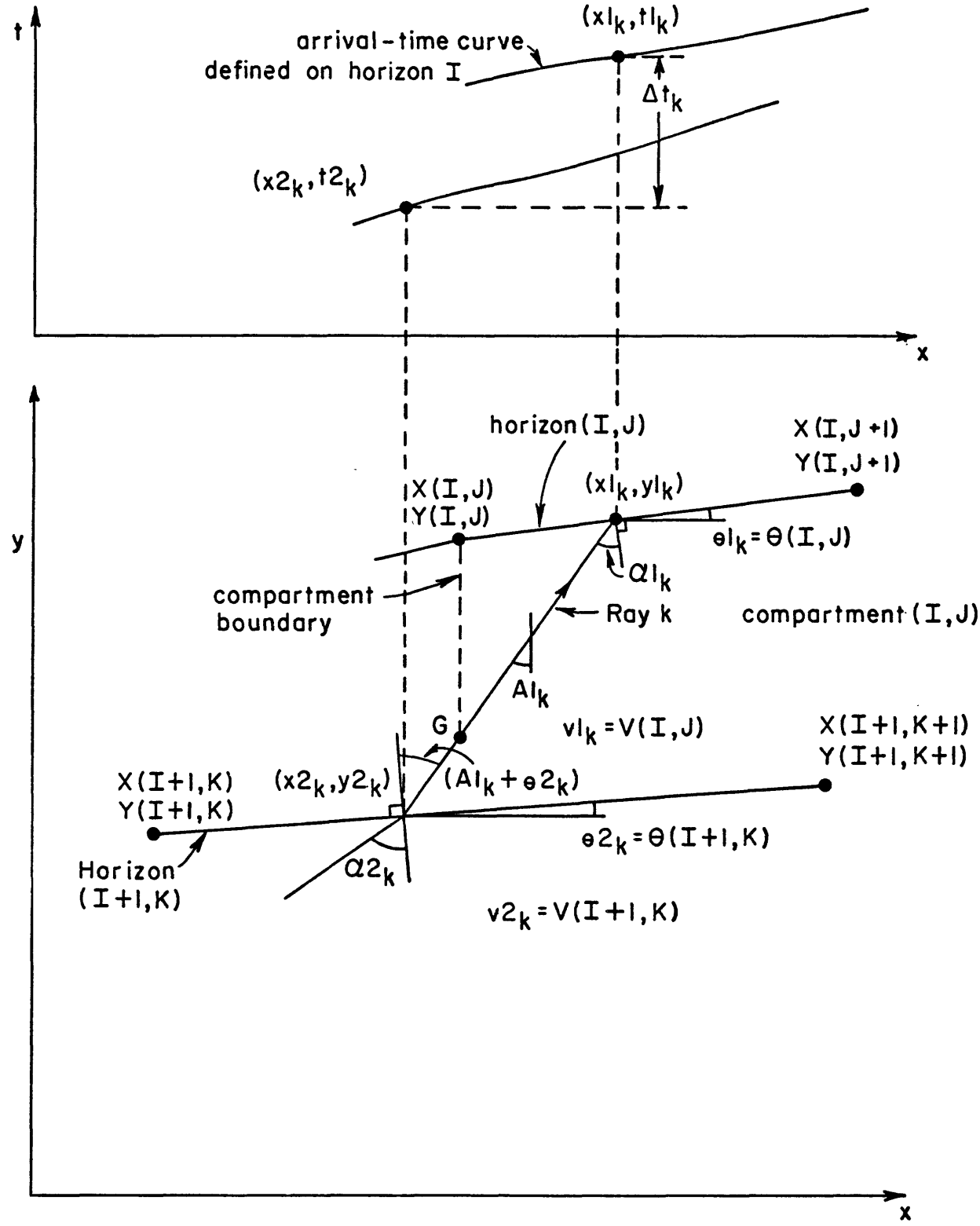

Figure 23.--Sketch showing parameters used in equation 1 through 21 for tracing an emerging ray through one complete layer.....

The emergence angle Al_k with respect to the vertical at (xl_k, yl_k) is seen from figure 23 to be

$$Al_k = \alpha l_k - \theta l_k \tag{9}$$

and hence the equation of the kth ray path which emerges at (xl_k, yl_k) is

$$\frac{y-yl_k}{x-xl_k} = \tan\left(\frac{\pi}{2} - Al_k\right) \tag{10}$$

Because the emergent ray is usually not vertical, it may intersect compartment boundaries (e.g., at point G in figure 23) and be refracted within a layer. In this case equation (9), and equation (6) cannot be used to calculate Al_k and vl_k respectively. (The programs contain the option not to refract rays at compartment boundaries.) However, the details regarding such changes at compartment boundaries are largely a matter of bookkeeping within the programs and are not discussed here.

The ray path defined by equation (10) (shown as Ray k in figure 23) will intersect the next underlying (I+1)th horizon. If it intersects in the Kth compartment of this horizon, then the equation of the (I+1)th horizon at the intersection point, shown as $(x2_k, y2_k)$ is

$$\frac{y - Y(I+1,K)}{x - X(I+2,K)} = \frac{Y(I+1,K+1) - Y(I+1,K)}{X(I+2,K+1) - X(I+1,K)} = M(I+1,K) \tag{11}$$

The point of intersection $(x2_k, y2_k)$ is found by solving equations (10) and (11) simultaneously, giving

$$x2_k = \frac{M(I+1,K)\,X(I+1,K) - Y(I+1,K) - \tan\left(\frac{\pi}{2} - Al_k\right) + yl_k}{M(I+1,K) - \tan\left(\frac{\pi}{2} - Al_k\right)} \tag{12}$$

$$y2_k = (x2_k - xl_k)\tan\left(\frac{\pi}{2} - Al_k\right) + yl_k \tag{13}$$

and as with horizon I (equations (5) and (6)), the dip of the (I+1)th horizon at $(x2_k, y2_k)$ and the velocity beneath this point are

$$\theta 2_k = \Theta(I+1,K) = \text{arc tan } M(I+1,K)\ , \tag{14}$$

and

$$v2_k = V(I+1,K) \tag{15}$$

The travel-time Δt_k for propagation between $(x1_k,y1_k)$ and $(x2_k,y2_k)$ is given by

$$\Delta t_k = \frac{\{(x1_k - x2_k)^2 + (y1_k - y2_k)\}^{2\ 1/2}}{v1_k} \tag{16}$$

and the arrival-time $t2_k$ at $(x2_k,y2_k)$ is given by

$$t2_k = t1_k - \Delta t_k \tag{17}$$

where $t1_k$ = arrival time at $x1_k$ (fig. 23).

Equations (1) through (17) provide the information to trace a ray from a kth point $(x1_k, y1_k)$ on the Ith horizon through one complete layer. We furthermore note that the triplet $(x1_k,y1_k,t1_k)$ on the Ith horizon maps into the triplet $(x2_k,y2_k,t2_k)$ on the (I+1)th horizon. Thus if an arrival-time curve on the Ith horizon is defined by the set of coordinates $(x1_k,y1_k,t1_k)$, the corresponding set of coordinates $(x2_k,y2_k,t2_k)$ define the arrival-time curve which would have been recorded had the seismometers been placed in the (I+1)th horizon. The Ith layer has been stripped! Clearly then all the data are available for stripping yet another layer through the change of variables

$$(x2_k,y2_k,t2_k,\theta 2_k,v2_k,I+1,K) \rightarrow (x1_k,y1_k,t1_k,\theta 1_k,v1_k,I,J) \tag{18}$$

and repeating equations (7) through (17) for the next deeper layer. Thus from a time-distance curve, which represents data recorded on the ground surface, we can derive another curve which represents data that would have been recorded had the seismometers been placed on the nth horizon.

It is theoretically sound to trace a complete arival-time curve to the nth horizon one ray at a time by applying Snells law instead of redefining the arrival-time curve at each successive horizon. However in practice it is far superior not to consider rays separately but rather through their relationship with each other at frequent intervals (horizons) by the use of derivatives.

Ray Tracing

At times an individual ray must be traced into the subsurface without having an arrival-time curve defined on the surface. In this case, the emergence angle in the first layer can be assigned a trial value, and those associated with deeper layers can be calculated by applying Snell's law. Thus, in figure 23, the emergence angle $\alpha 2_k$ at $(x2_k,y2_k)$ is calculated from the equation

$$\frac{\sin(\alpha 2_k)}{\sin(Al_k + \theta 2_k)} = \frac{v2_k}{v1_k} \tag{19}$$

or

$$\alpha 2_k = \arcsin\left\{\frac{v2_k}{v1_k} \sin(Al_k + \theta 2_k)\right\} \tag{20}$$

where the absolute value of { } <1. Thus the individual ray is traced through the next underlying layer by making the change in variables

$$(x2_k, y2_k, t2_k, \theta 2_k, v2_k, \alpha 2_k, I+1, K) \rightarrow (x1_k, y1_k, t1_k, \theta 1_k, v1_k, \alpha 1_k, I, J) \quad (21)$$

and repeating equations (9) through (20) (excluding equation 18). An individual ray can thus be traced to the deepest known horizon, the nth, by repeating equations (9) through (21) n-1 times.

In a similar manner rays can be traced upwards from the subsurface (e.g., from the nth horizon to the first) provided again that some initial angle such as the critical angle, is given. This is equivalent to turning figure 23 upside down. In the discussions to follow, tracing up-going or down-going rays or travel-time curves, will be considered identical and will be referred to by equations (1) through (21).

Notation

In calculating the (n+1)th horizon, it is assumed that all calculations to obtain the configuration and velocity of the overlying n layers, and all the necessary extensions and constructions of its arrival-time curves, as discussed in the previous section, have been made. Furthermore, it is assumed that arrival-times have been corrected for shot depth by adding an uphole time, a provision automatically handled in the programs. Thus we deal with arrivals from just one horizon at a time (the n+1th) with the shotpoint on the ground surface (horizon 1).

To simplify discussions and avoid lengthy mathematical expressions, the ith horizon will be represented by the function $y_i(x)$, and its layer velocity by $v_i(x)$ (fig. 24).

Because the (n+1)th horizon may have been recorded from more than one shotpoint, the arrival-time curves for each shotpoint are also subscripted. Thus $t_q(x)$ will be the functional notation of the arrival-time curve from the (n+1)th horizon for the qth shotpoint $Sq(xS_q, yS_q)$ with both shot and seismometers on horizon 1 (fig. 24).

Calculations involved in a 2-layer problem are relatively straightforward. For the (n+1)th horizon, the arrival-time curves $t_q(x)$ are first reduced to the deepest known horizon, namely the nth, which may then in a sense be thought of as datum. The expression $T_q(x)$ represents the arrival-time curve for the qth shotpoint that would have been recorded if the shotpoint remained on horizon 1 but the recording seismometers were on the nth (fig. 24). (Thus if the second horizon is the one being calculated, n=1 and $t_q(x)=T_q(x)$).

For purposes of demonstration, a four-layer problem (n=3) with a single shotpoint with index q is used (fig. 25).

To obtain $T_q(x)$ from $t_q(x)$, successive points $(x1_i, t1_i)$ are selected on $t_q(x)$ and through the use of equations (1) through (17), their respective

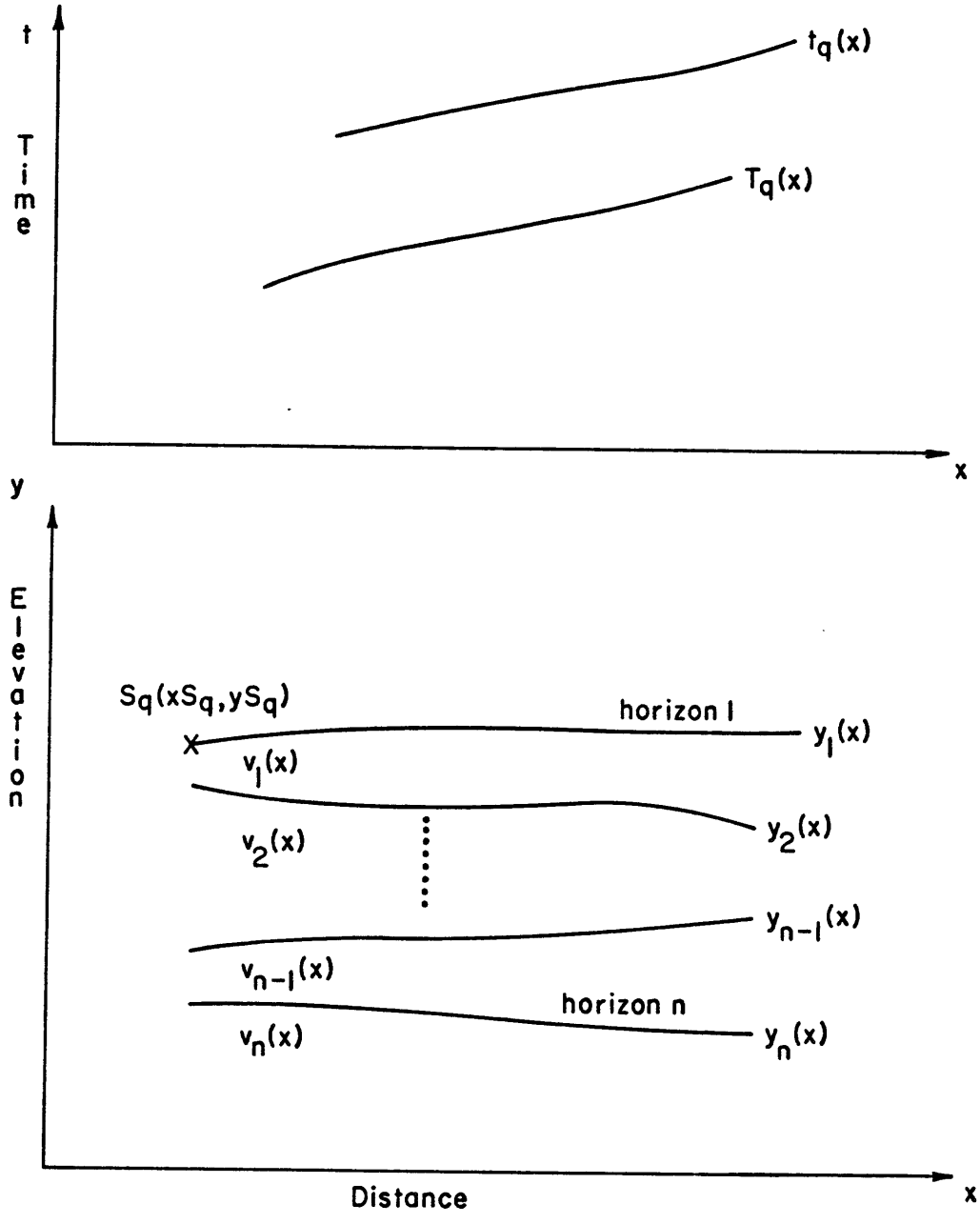

Figure 24.--Simplified notation for expressing the equation of arrival-time curves and a layered velocity model.....

emergent rays are traced through one layer, resulting in an arrival-time curve for horizon 2, shown as the dashed curve. The process is repeated (equation (18)) resulting in $T_q(x)$. This function then represents the arrival-time curve from horizon 4 which would have been recorded on horizon $y_3(x)$ with the shotpoint on the surface. (Initially the programs were written without constructing the intermediate arrival-time curves by using equations (19) and (20). This sometimes resulted in errors due to intersecting ray paths.)

The process of deriving Tq(x) from tq(x) results, in nearly all cases, of a shift of the arrival-time curves in the direction of the shotpoint as shown in figure 25.

Depth and velocity calculations from reduced data using reversed plots (the reciprocal method)

The basic assumption used in interpreting data from reversed graphs is that part of the ray path down to and along a refracting interface is common to all the rays that are refrcted from that interface. This is illustrated in figure 26a by the common path along the (n+1)th horizon. For the ray paths ABCE and ABCDF, ABC is common to both ray paths. This condition, however, is never completely satisfied. In the case of a velocity increase with depth, (fig. 26b) in the (n+1)th layer, rays to successively more distant receiver locations propagate at ever increasing depths. In the case where a horizon is offset, perhaps by a fault (fig. 26c), the refraction paths are also variable. Another example (not illustrated) is if travel paths to successive receivers are not all in the same plane; this is usually the case when seismic lines are crooked.

Given the reversed arrival-time curves $t_q(x)$ and $t_p(x)$, (fig. 27a) the reduced curves are $T_q(x)$ and $T_p(x)$ (fig. 27b). Curves Tq(x) and Tp(x) represent arrivals from the (n+1)th horizon that would have been recorded if the shotpoints were on the surface and the receivers on the nth horizon. (We note that the reduced curves extend behind the shotpoints, which cannot occur unless the functions $t_q(x)$ and $t_p(x)$ represent data extended back toward the intercept.) Because of reciprocity, the travel-time between shotpoints S_q and S_p equals that in the reverse direction from S_p to S_q and is called the reciprocal time shown as TR in figure 27a. By assumption, all arrivals refracted from the (n+1)th horizon travel in part along the reciprocal ray path S_qABS_p in figure 27c. The path between A and B defines the (n+1)th horizon and each point on it, by Huygens principal, may be thought of as a source point. Hence, any point such as E_i on this horizon, is a common source point from the reversed shotpoints S_q and S_p, which are recorded at points $Q_i(xQ_i,yQ_i)$ and $P_i(xP_i,yP_i)$ respectively on horizon n. On the reduced reversed plots, the corresponding arrival-times are indicated as TQ_i and TP_i.

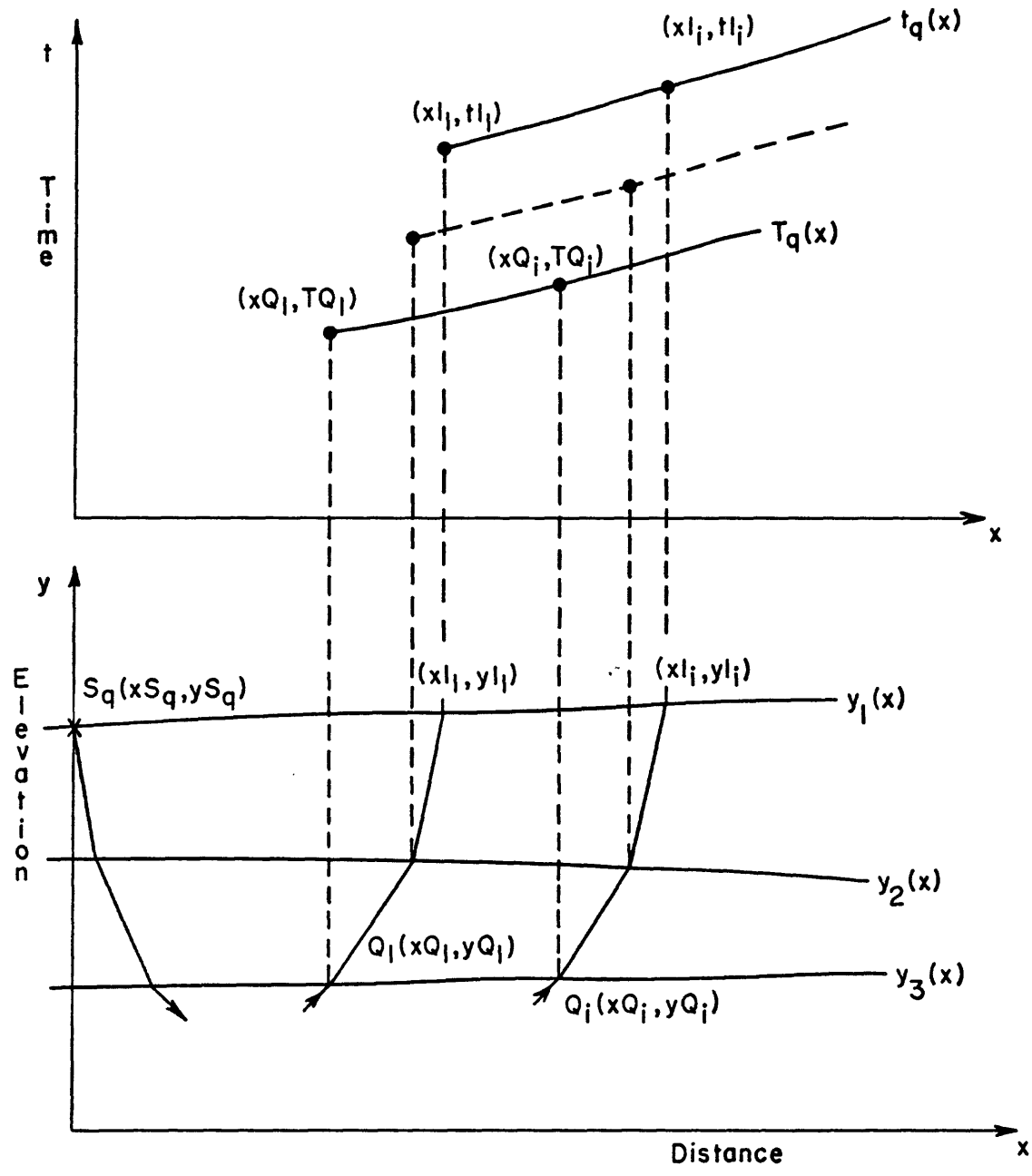

Figure 25.--Sketch illustrating the reduction of an arrival-time curve from horizon 1 to horizon 3 for the case of a four-layer problem.....

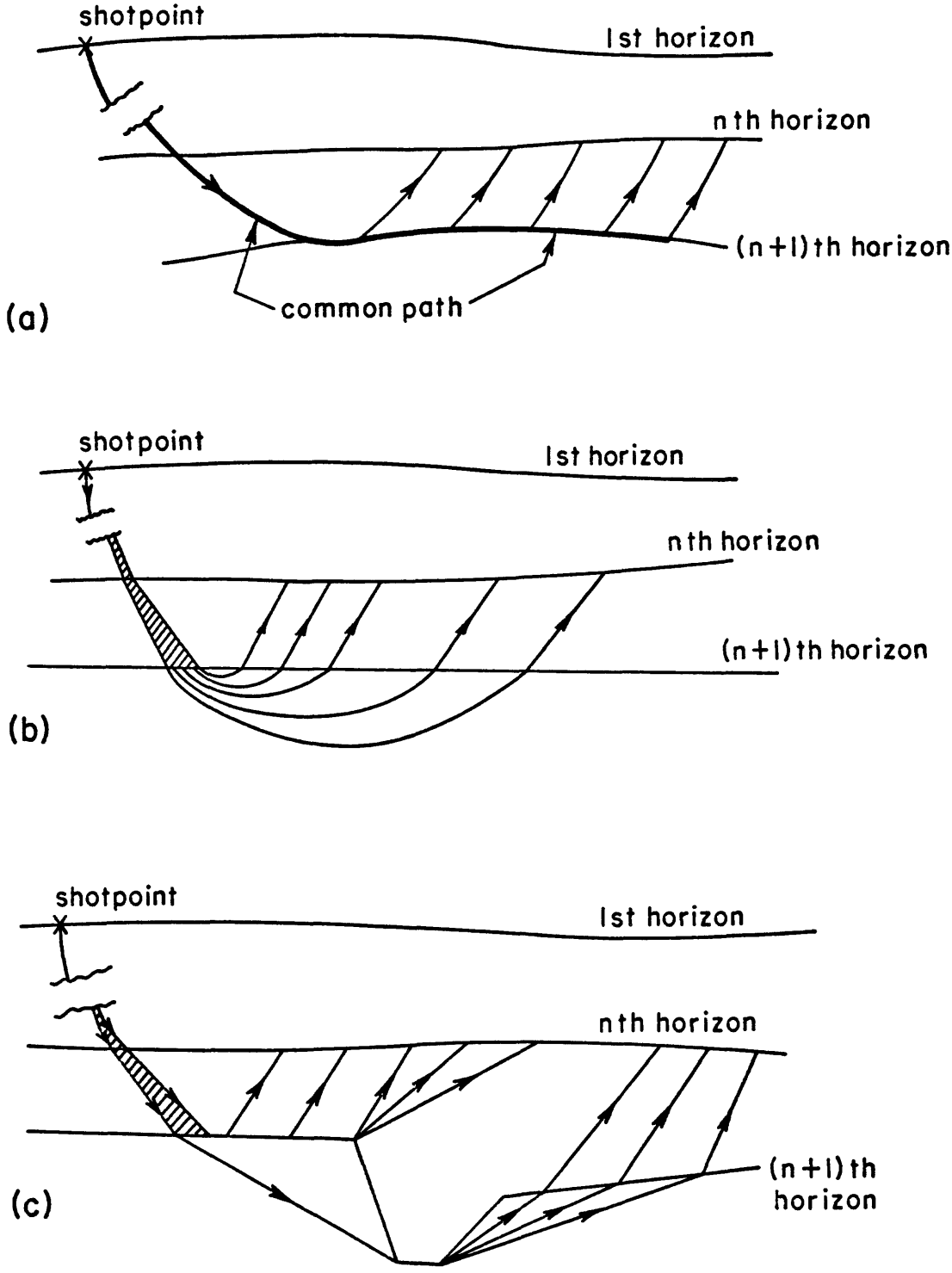

Figure 26.--Sketch showing a common for a refracted ray path (fig a). Figures b and c show cases where the ray paths are not common.....

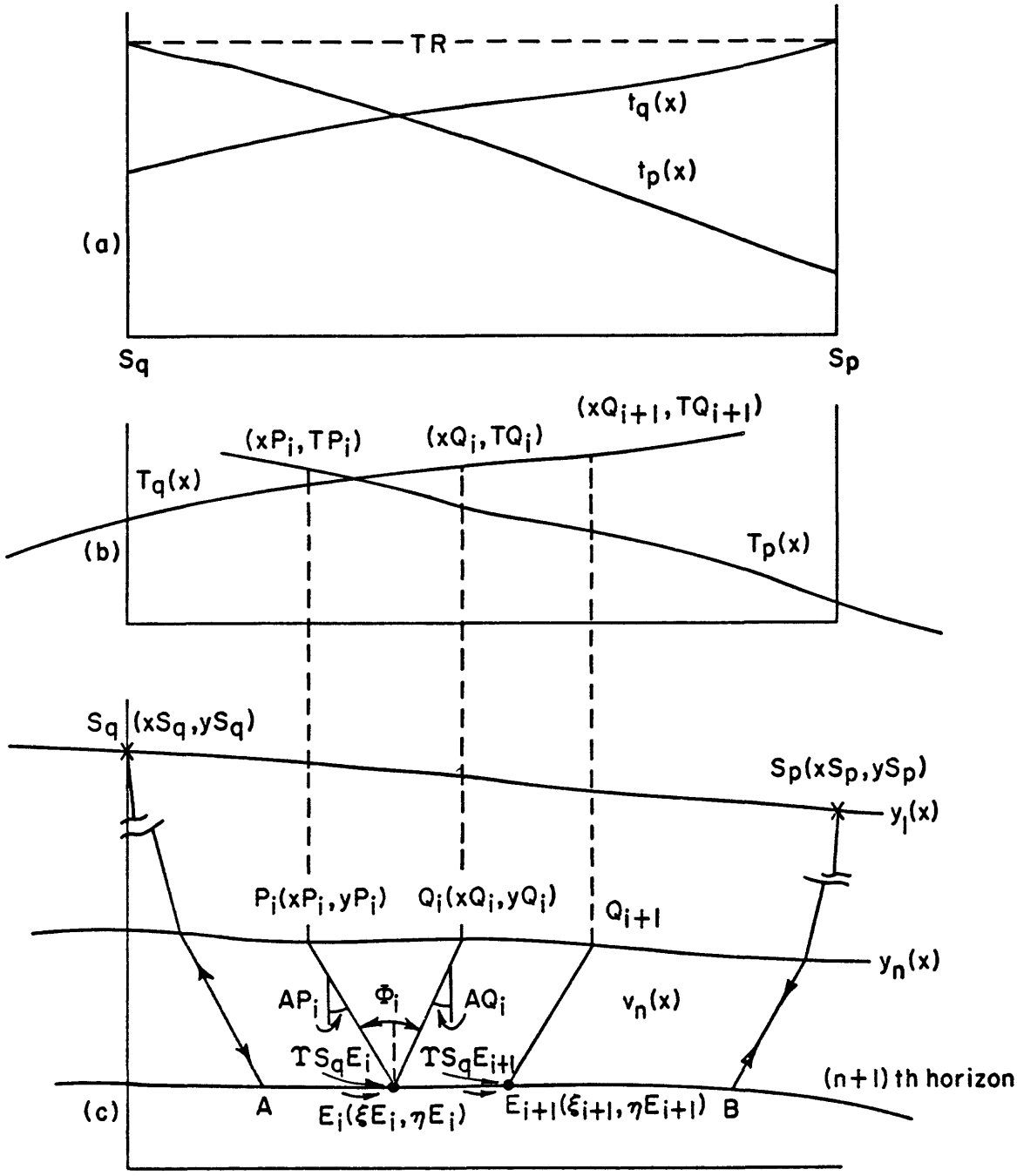

Figure 27.--Sketch showing parameters used in equations 22 through 31 for calculating points on the (n+1)th horizon and the velocity between successive points.....

Because the travel-path along the (n+1)th horizon for arrivals recorded at points P_i and Q_i from reversed shots both terminate at E_i, and the combined paths S_p to E_i plus S_q to E_i correspond to the reciprocal path, the reciprocal time TR equals the sum of the travel-times to P_i and Q_i less the time for the slant emerging paths from E_i to P_i and E_i to Q_i, or

$$TR = (TQ_i - TE_iQ_i) + (TP_i - TE_iP_i) \tag{22}$$

where
TR = reciprocal time
TQ_i = arrival-time at Q_i
TE_iQ_i = travel-time from E_i to Q_i
TP_i = arrival-time at P_i
TE_iP_i = travel-time from E_i to P_i

It is further inferred from the above argument, that any two points such as (xQi, TQi) and (xPi, TPi), on reversed travel-time curves which satisfy equation (22) also define a point, such as E_i, on the refracting horizon.

Points on the (n+1)th horizon are found by systematically searching the reversed arrival-time curves for values which satisfy equation (22).

For any two trial points such as P_i and Q_i with distance coordinates xP_i and xQ_i (fig. 27), the corresponding arrival-times TP_i and TQ_i are determined by evaluating $T_p(x)$ and $T_q(x)$ (e.g., $TP_i=Tp(xP_i)$ and $TQ_i=T_q(xQ_i)$.) The reciprocal time TR is obtained by evaluating tp(x) and tq(x) at the reversed shotpoint locations (e.g., $TR=tp(xS_q)=tq(xS_p)$.) Equation (22) can then be tested by computing values for TE_iQ_i and TE_iP_i. This is done by first calculating the elevations yP_i and yQ_i (fig. 27) (e.g., $yP_i=y_n(xP_i)$ and $yQ_i=y_n(xQ_i)$.) By applying the ray-tracing methods derived from equations (1) through (9) to the functions $T_p(x)$ and $T_q(x)$ at points P_i and Q_i respectively, the angles AP_i and AQ_i that the emergent ray makes with a vertical line (fig. 27) at P_i and Q_i are determined. The two raypath equations, by analogy with equation (10) may then be written as

$$\frac{y-yP_i}{x-xP_i} = \tan(\pi/2 - AP_i) \tag{23}$$

$$\frac{y-yQ_i}{x-xQ_i} = \tan(\pi/2 - AQ_i) \tag{24}$$

Equations (23) and (24) are solved simultaneously for their intersection point, shown as the common source point $E_i(\xi E_i, \eta E_i)$ in figure 27c. The

"average" velocities along the emergent rays between E_i and P_i and between E_i and Q_i(vP_i and vQ_i) are then obtained from the known function $v_n(x)$, and the travel-times TE_iP_i and TE_iQ_i may be written as

$$TE_iQ_i = \frac{\{(xQ_i - \xi E_i)^2 + (yQ_i - \eta E_i)^2\}^{1/2}}{vQ_i} \tag{25}$$

$$TE_iP_i = \frac{\{(xP_i - \xi E_i)^2 + (yP_i - \eta E_i)^2\}^{1/2}}{vP_i} \tag{26}$$

Having now computed all the parameters in equation (22) at the trial points P_i and Q_i, they are substituted and tested to determine whether the right side actually equals the reciprocal time. If they do,then point E_i lies on the (n+1)th horizon. If not, other pairs such as P_i and Q_{i+1} are tested until a successful pair are found which satisfy equation (22). By repeating this procedure, a sequence of points ..... E_i, E_{i+1}, ... (fig. 27c) are determined which satisfy the reversed arrival-time curves and hence define the (n+1)th horizon.

The velocities in the intervals between successive calculated points E_i and E_{i+1} in the (n+1)th layer are obtained from the parameters already calculated. The travel-times TS_qE_i (fig. 27c) from shotpoint S_q to E_i and TS_qE_{i+1} from S_q to E_{i+1} are given by

$$\begin{aligned} TS_qE_i &= TQ_i - TE_iQ_i \\ TS_qE_{i+1} &= TQ_{i+1} - TE_{i+1}Q_{i+1} \end{aligned} \tag{27}$$

The travel-time ΔT_i in the interval between E_i and E_{i+1} then is

$$\Delta T_i = TS_qE_{i+1} - TS_qE_i \tag{28}$$

and the velocity V_i in the ith interval is

$$V_i = \{(\xi E_{i+1} - \xi E_i)^2 + (\eta E_{i+1} - \eta E_i)^2\}^{1/2} / \Delta T_i \tag{29}$$

The velocity of the (n+1)th layer is thus represented by a series of velocities within successive increments, ...V_i, V_{i+1},....., from which the lateral velocity variations in this layer are represented.

The velocity of the (n+1)th layer may also be calculated by assuming that the rays E_iQ_i and E_iP_i (fig. 27c) emerged at the critical angle from the (n+1)th horizon. Thus the angle Φ_i formed between these two emerging rays is twice the critical angle and from Snell's law

$$\sin(\Phi_i/2) = \frac{v_n(\xi E_i)}{V_i} \tag{30}$$

or

$$V_i = \frac{v_n(\xi E_i)}{\sin(\Phi_i/2)} \tag{31}$$

where $v_n(\xi E_i)$ = velocity of the nth layer at ξE_i
V_i = velocity of the (n+1)th layer at ξE_i.

The angle Φ_i can be determined geometrically because the coordinates of the points P_i, Q_i and E_i have already been determined. If emergence at E_i is not at the critical angle, such as at a diffraction point, equation (31) will not yield a correct value of V_i.

In practice, velocities are calculated using both equations (29) and (31) permitting a comparison of two independent methods.

The reciprocal method just described utilizes only those parts of reversed arrival-time curves which have common depth points (e.g., have recorded arrivals from the same part of a horizon). Because of this restriction significant portions of reversed arrival-time curves which have been extended to the time intercept frequently can not be used. For example, the overlapping arrival-time curves of figure 27a may suggest complete subsurface overlap in reversed directions. However, the shifting of these curves (fig. 27b) due to the reduction process indicates a lack of subsurface overlap from the (n+1)th horizon. The extent of true subsurface overlap is shown in figure 28. As indicated by the emergent raypaths, the curve $T_q(x)$ represents subsurface coverage between points A and C, whereas $T_p(x)$ represents coverage between points B and D. Thus, true subsurface overlap occurs only between points B and C which corresponds to the heavy portions of the arrival-time curves (fig. 28). Hence if arrival-time curves have been extended, the reciprocal method utilizing equations (27 through 31) may apply to only parts of the curves. Depths corresponding to the lightly drawn portions must be estimated using other algorithms.

Figure 28 shows that the lightly drawn portions of the reduced travel-time curves do not represent actual refraction arrivals because they lie between the shotpoint and the point of reflection at the critical angle (e.g., point B for shotpoint S_q and point C for shotpoint S_p). They must, therefore, represent portions of the data which have been extended using other information such as complementary plots for control. Nevertheless, they can be used for making computations. If, for example, travel-times from shotpoint S_p had been obtained to the left of point S_q, then the curve between points d and e could be used in the reciprocal method of depth calculation.

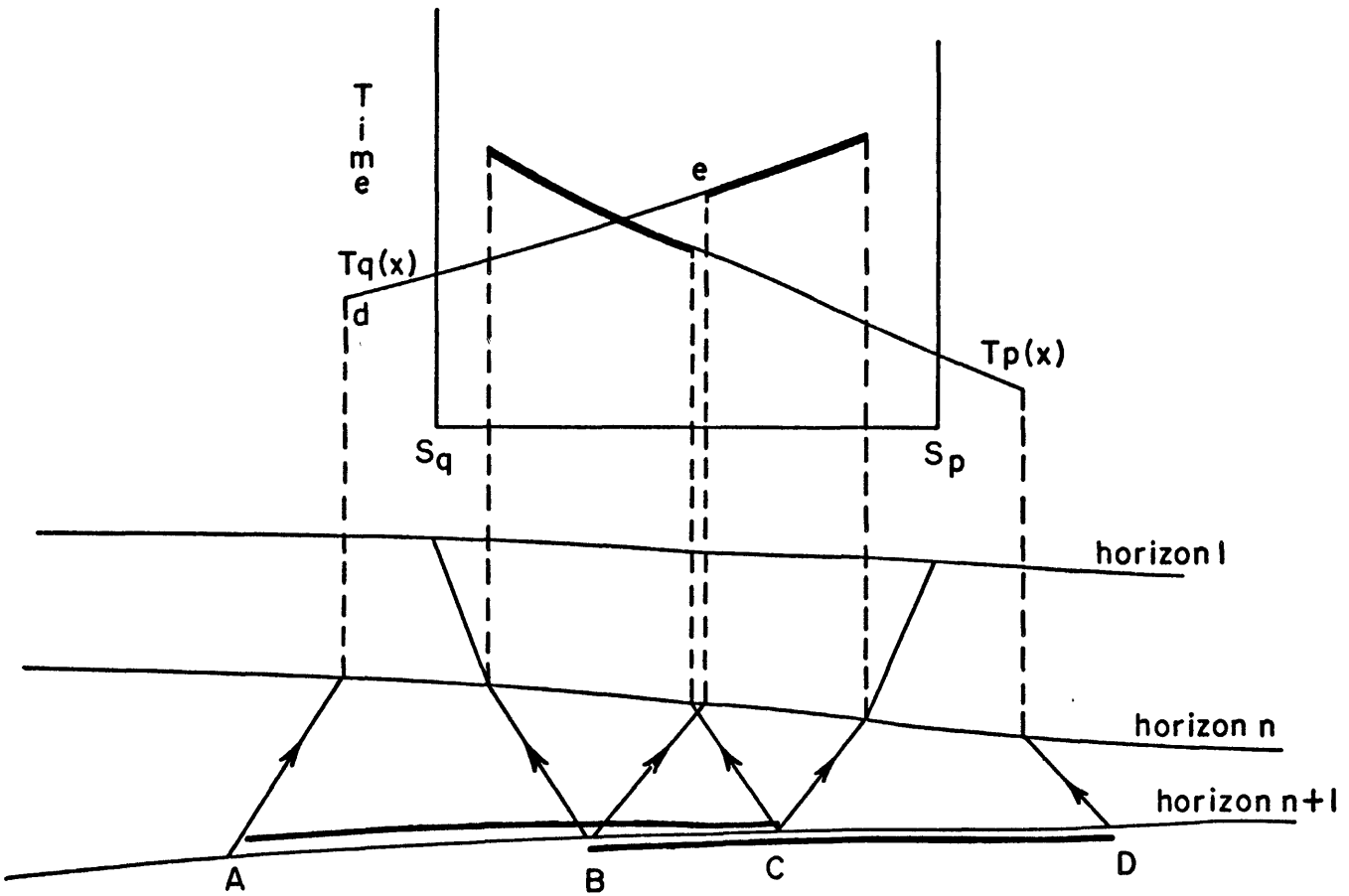

Figure 28.--Sketch illustrating that only portions of reversed travel-time curves between shotpoints may contain the subsurface overlap for which reciprocal methods apply.....

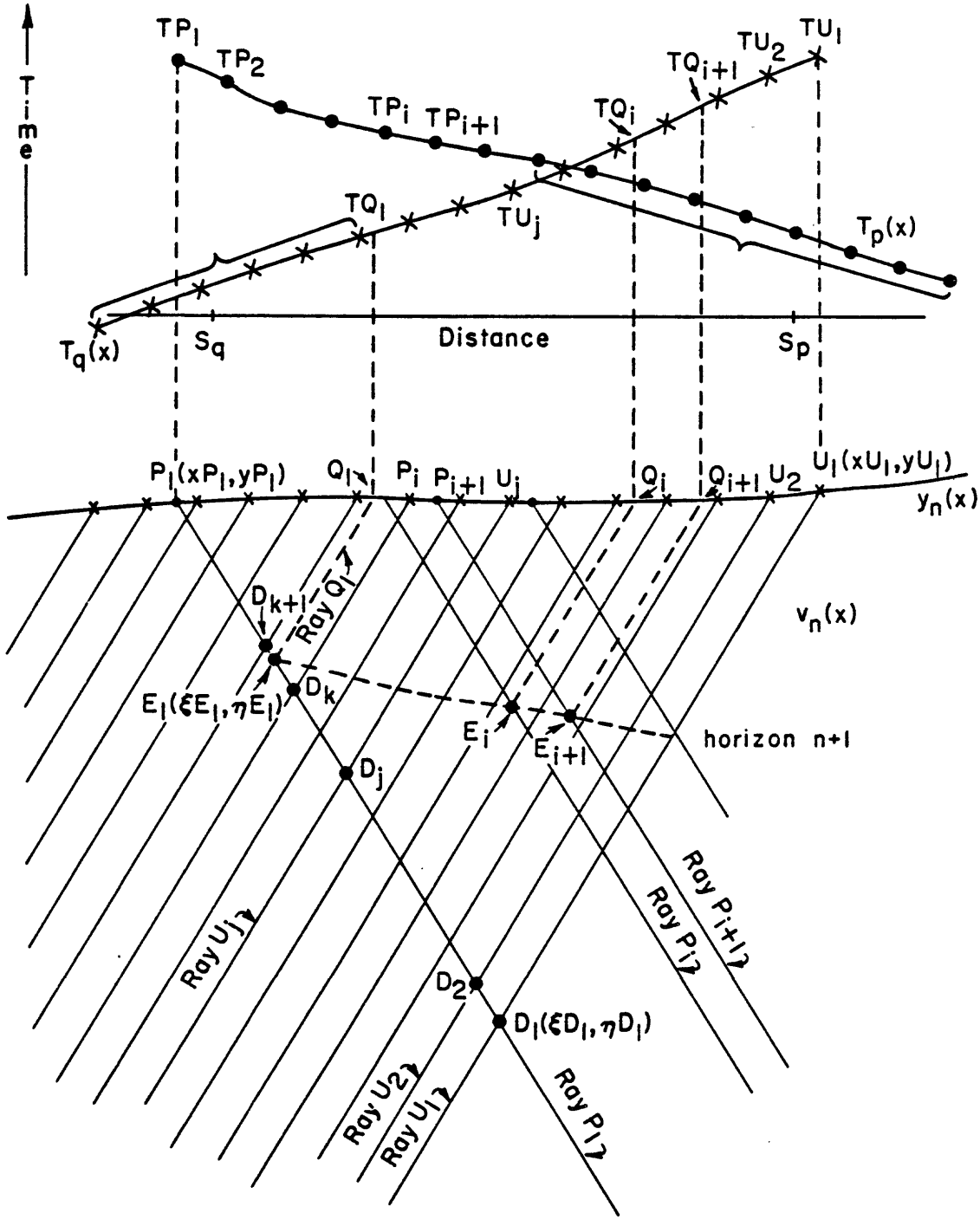

Figure 29.--Sketch illustrating the parameters used in the algorithm for the reciprocal method discussed in the text.....

The example in figure 28 may represent a rather extreme example of only a small percentage of real subsurface overlap. This situation arises, however, when deploying offset shotpoints to map horizons which begin to approach a depth equal to the spread length, such as shown in figure 28.

A procedural outline of the reciprocal method from reduced arrival-time curves $T_q(x)$ and $T_p(x)$ used by the computer programs is given with the aid of figure 29. First, the reciprocal time TR is calculated as described previously.

At points U_1, U_2 $\cdots$ $U_j(xU_j,yU_j)$, $\cdots$ taken at regular intervals from right to left along horizon n, the arrival-times TU_j (e.g., $TU_j = T_q(xU_j)$) and the equations for emerging rays (analogous to equation (24)), simply designated at Ray U_j in figure 29) are calculated and stored. Thus,on the curve representing the function Tq(x), TU_1 is the arrival-time and Ray U_1 is the representation of the equation of the emerging ray at the far right-hand point.

Beginning at P_1, corresponding to the far left-hand point on $T_p(x)$, the arrival-time TP_1 (e.g., $TP_1 = T_p(xP_1)$) and the emerging ray equation Ray P_1 (analogous to equation (23)) are also calculated. The points of intersection between Ray P_1 and other rays, Ray U_j, are calculated resulting in the points D_1, D_2, $\cdots$ D_j, $\cdots\cdot$. Using the function $v_n(x)$, the travel-time from each D_j to P_1 (e.g., TD_jP_1) and from each D_j to U_j (e.g., TD_jU_j) are calculated (analogous to equations (26) and (25)). The expression

$$TR - [(TU_j - TD_jU_j) + (TP_1 - TD_jP_1)] \qquad (32)$$

(analogous to equation (22)) is evaluated for each D_j. If a solution exists along Ray P_1, expression (32) will change sign from negative to positive between two calculation points shown as D_k and D_{k+1} in figure 29. Between these two points, a point $E_1(\xi E_1,\eta E_1)$ is found which is the interpolated zero crossing of expression (32). The interpolation process also yields values used for later calculations: $Q_1(xQ_1,yQ_1)$ which is the point of emergence on horizon n, of Ray Q_1 which intersects E_1; TQ_1, the arrival-time at Q_1; and TE_1Q_1, the travel-time along Ray Q_1 between points E_1 and Q_1. With these parameters in hand, equation (27) is used to calculate TS_qE_1, namely the travel-time from shotpoint S_q to the subsurface point E_1.

This process is repeated for all successive points $\cdots$ P_i, P_{i+1} $\cdots$ along $y_n(x)$ yielding a succession of points $\cdots$ $E_i(\xi E_i,\eta E_i)$, $E_{i+1}(\xi E_{i+1},\eta E_{i+1})$ $\cdots$ and transit times $\cdots$ TS_qE_i, TS_qE_{i+1}, $\cdots$. The E_i's define the (n+1)th horizon. Equations (28) and (29) and equation (31) are used to calculate the velocity along the (n+1)th layer in two different ways.

It may be noted from the geometry of figure 29 that the parts of the travel-time curves included within brackets are the result of extensions obtained from complementary data. Depths cannot be calculated from these parts using this method due to the lack of subsurface overlap.

Depth calculations from reduced data using one-way plots
(the critical reflection method)

In the absence of overlapping reversed plots, horizon depths may be calculated from one-way data by assuming values for the velocity V of the (n+1)th layer. If non-overlapping portions of arrival-time curves recorded in opposite directions are available, reasonably accurate estimates of V are possible by averaging apparent velocities.

A basic assumption used in calculating one-way data is that the reduced arrival-time curve $T_q(x)$ (fig. 30) contains a point (xC, TC) which represents a reflection from a point (ξC,ηC) on the (n+1)th horizon at the critical angle $\Phi C/2$. The point (xC,TC) represents the intersection between the time-distance plot of the theoretical reflection curve and the refraction arrival-time curve extended back toward the intercept. The subsurface point (ξC,ηC) on the (n+1)th horizon at which the reflection occurs is called the critical reflection point (not to be confused with Nettleton's (1940, p. 249) critical distance).

The point on the nth horizon at which the critical reflection emerges is labeled (xC,yC). Points (xQ_1, TQ_1) on $T_q(x)$ shown to the right of (xC,TC), have a raypath which also extends downward to (ξC,ηC), then laterally with positive travel-time to (ξQ_i,ηQ_i) and then upward to (xQ_i,yQ_i). Points (xQ_j,TQ_j) on $T_q(x)$, in the reverse direction, to the left of (xC,TC), have a raypath which also extends downward to (ξC,ηC), then laterally with negative travel-time to (ξC_j,ηC_j) and then upward to (xQ_j,yQ_j). Such rays with negative travel-time between (ξC,ηC) and (ξQ_j,ηQ_j) do not actually occur, but are obtained geometrically by extending arrival-time curves toward or beyond the shotpoint. They only have mathematical significance and can be used for estimating depths to horizons. We will refer to ray paths having only positive segments as "realizable", and those containing negative segments as "non-realizable".

The point (ξC,ηC) on the (n+1)th horizon is calculated by systematically searching the subsurface below the nth horizon for a point giving a reflection at the critical angle which satisfies the arrival-time curve $T_q(x)$. This is illustrated by Figures 31 and 32.

Points $Q_i(xQ_i,yQ_i)$ (fig. 31) are selected at constant intervals along $y_n(x)$. The equations (Ray Q_i) of the emerging rays are calculated using equation (24). At successive points W_j (ξW_j,ηW_j) along Ray Q_i, this emerging ray is rotated through twice the critical angle ΦC {$\Phi C/2$ = arc sin $\{v_n(\xi W_j)/V\}$ in the direction toward the shotpoint S_q. This rotated ray represents a possible downgoing (submerging) ray. Its equation (shown as Ray U_j in fig. 31) is determined from geometry. The intersection between Ray U_j with horizon $yn(x)$ is calculated, resulting in point $Uj(xUj,yU_j)$ as well as the submergence angle β_j. The angle β_j is analogous to $\alpha 2_k$ in figure 23, and hence the submerging ray can be traced to the surface (horizon 1) by using equations (1) through (21). It intersects the surface horizon $y_1(x)$ at the point $R_j(xR_j,yR_j)$. The position of R_j is compared with that of the shotpoint S_q and is shown in figure 31 to lie to

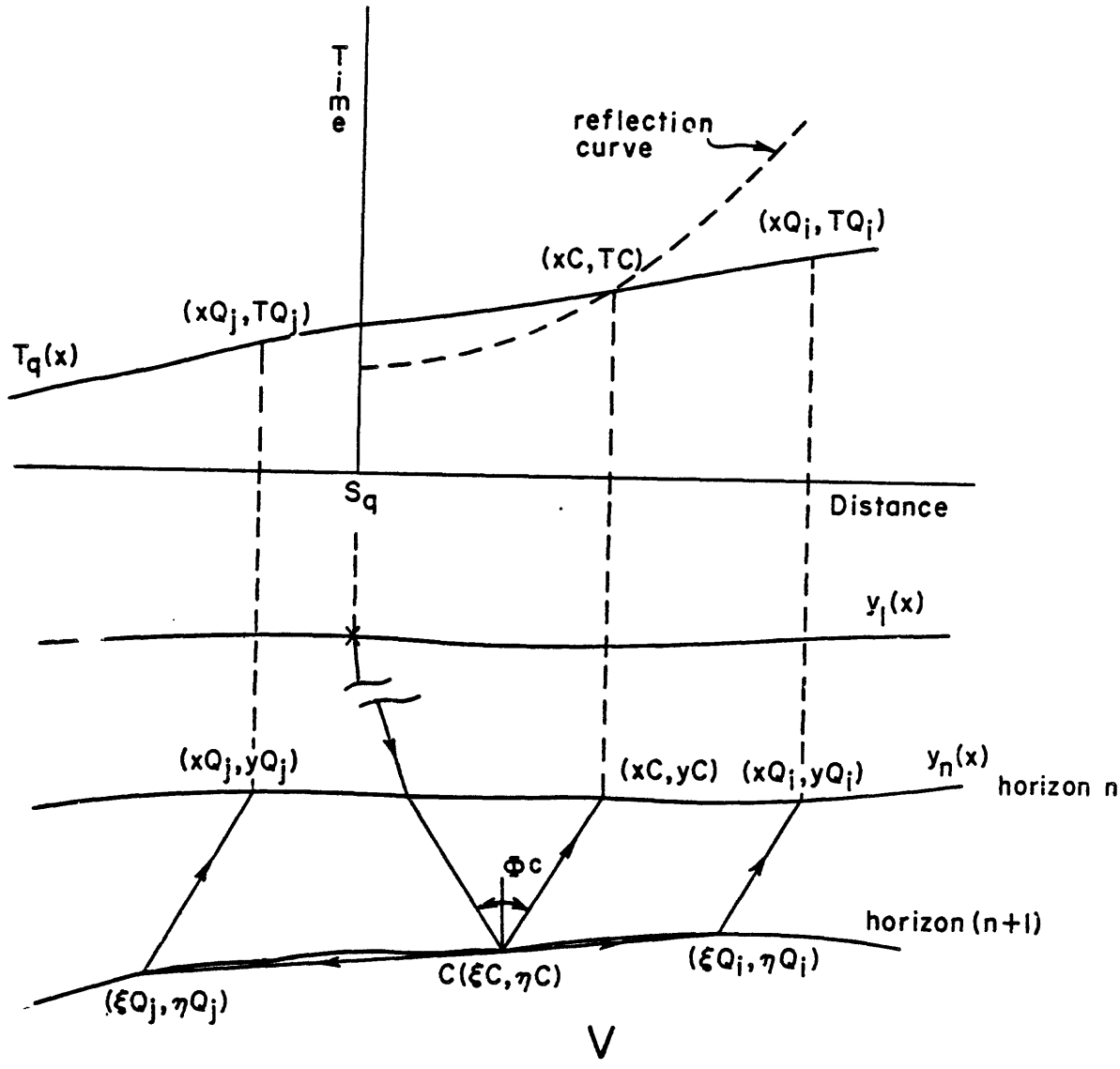

Figure 30.--Sketch illustrating basic assumptions used for calculating the critical reflection point from one-way data and points representing positive and negative travel-time along the refracting horizon.....

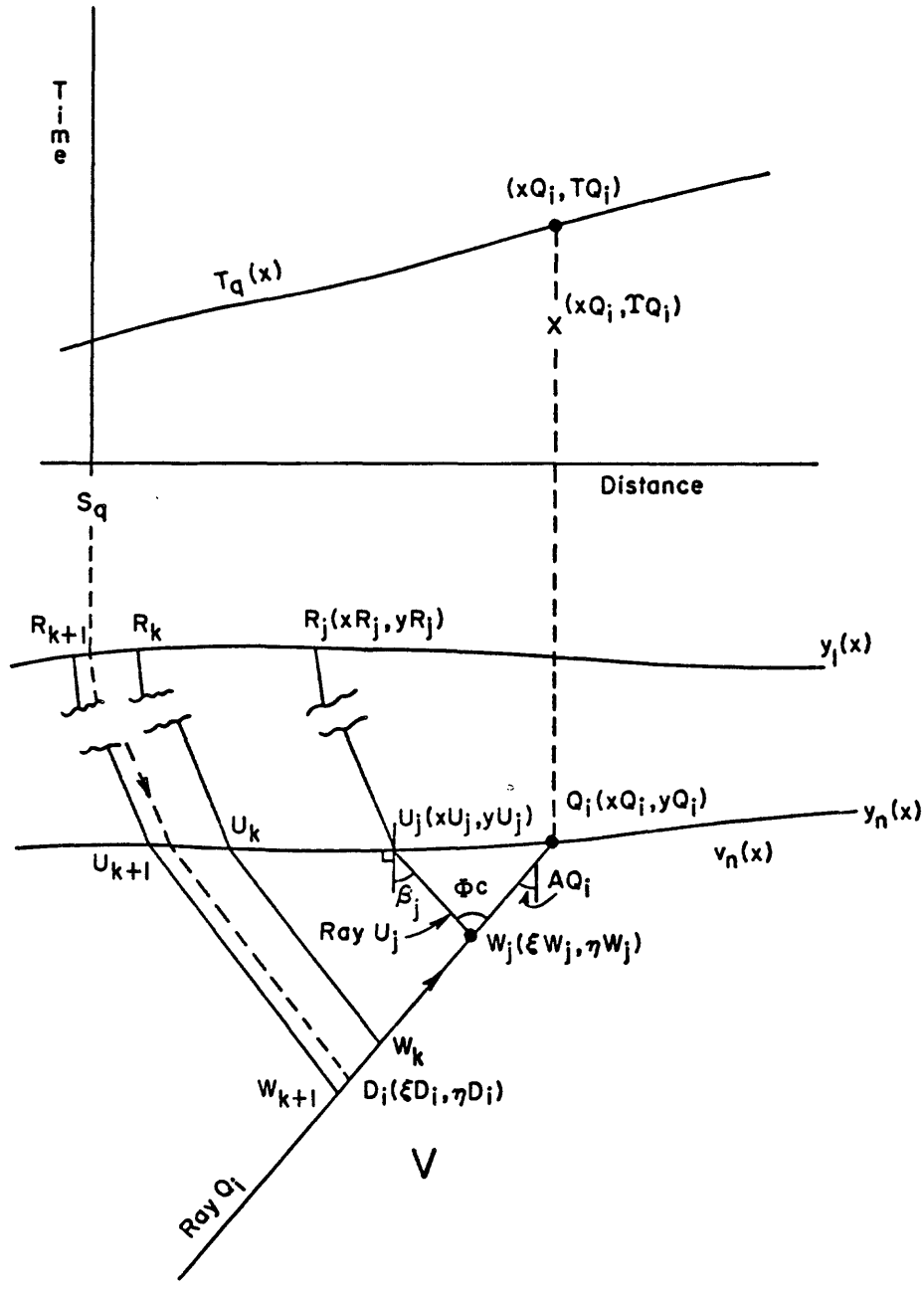

Figure 31.--First sketch illustrating the parameters used in the algorithm for the critical reflection method discussed in the text.....

the right of S_q. By testing the successive points W_j, two points shown as W_k and W_{k+1} may be found for which the corresponding surface points R_k and R_{k+1} straddle S_q. Interpolation then yields the coordinates of a point D_i $(\xi D_i, \eta D_i)$ for which the rotated ray intersects the shotpoint S_q and hence represents a possible critical reflection point. Its travel-time TQ_i is calculated from the travel path, shown plotted as coordinate (xQ_i, TQ_i) on the graph of figure 31. If this point falls on the function $T_q(x)$, then $D_i(\xi D_i, \eta D_i)$ must be the critical reflection point on the (n+1)th horizon.

It may be seen from figure 32 that for the points of Q_i nearer the shotpoint S_q, TQ_i will be less than the actual arrival-time TQ_i (e.g., $TQ_i - TQ_i > 0$). For Q_i more distant from the shotpoint, the curve defined by the (xQ_i, TQ_i) eventually intersects $T_q(x)$ and the critical reflection point $C(\xi C, \eta C)$ on the (n+1)th horizon corresponds to the intersection of these two curves. This point of intersection is determined by incrementing Q_i until the point Q_{k+1} (fig. 32) is found for which $(TQ_{k+1} - TQ_{k+1} > 0)$. The critical reflection point C then falls between D_k and D_{k+1} and is found by interpolation between these two points. The critical arrival-time TC and the point of emergence $QC(xQC, yQC)$ on the nth horizon are similarly found by interpolation.

The submerging ray from S_q, which is critically refracted into the (n+1)th layer also intersects this horizon at the critical reflection point $C(\xi C, \eta C)$. The travel-time TS_qC from S_q to C is important for subsequent depth computations, obtained by making use of the remainder of the arrival-time curve $T_q(x)$. The value of TS_qC is obtained by subtracting the travel-time between C and QC from TC, or

$$TS_qC = TC - \frac{\{(\xi C - xQC)^2 - (\eta C - yQC)^2\}^{1/2}}{vC} \qquad (33)$$

where vC is a representative velocity for the nth layer between the points C and QC, which were derived from $v_n(x)$.

Once the critical parameters $C(\xi C, \eta C)$ and TS_qC have been determined, other points on the (n+1)th horizon satisfying the arrival-time curve $T_q(x)$ can be computed. This is illustrated in figure 33. It shows a realizable raypath the right of C, critically refracted at $D_i(\xi D_i, \eta D_i)$ and emerging at $Q_i(xQ_i, yQ_i)$on the nth horizon with arrival-time TQ_i. Also shown is a non-realizable raypath with negative travel-time to the left of C, critically refracted at $D_j(\xi D_j, \eta D_j)$ and emerging at $Q_j(xQ_j, yQ_j)$ with arrival-time TQ_j. These latter points arise from having extended the arrival-time curve to the time intercept or beyond (e.g., in the direction of the shotpoint from the point of emergence QC of the critically reflected ray).

The raypath, within reasonable tolerance, between the critical reflection point $C(\xi C, \eta C)$ and a point $Q_i(xQ_i, yQi)$ on the nth horizon can be thought of as consisting of two segments, one from C to D_i on the (n+1)th horizon and the other from D_i to Q_i. The total travel-time along these segments is given by $(TQ_i - TS_qC)$.

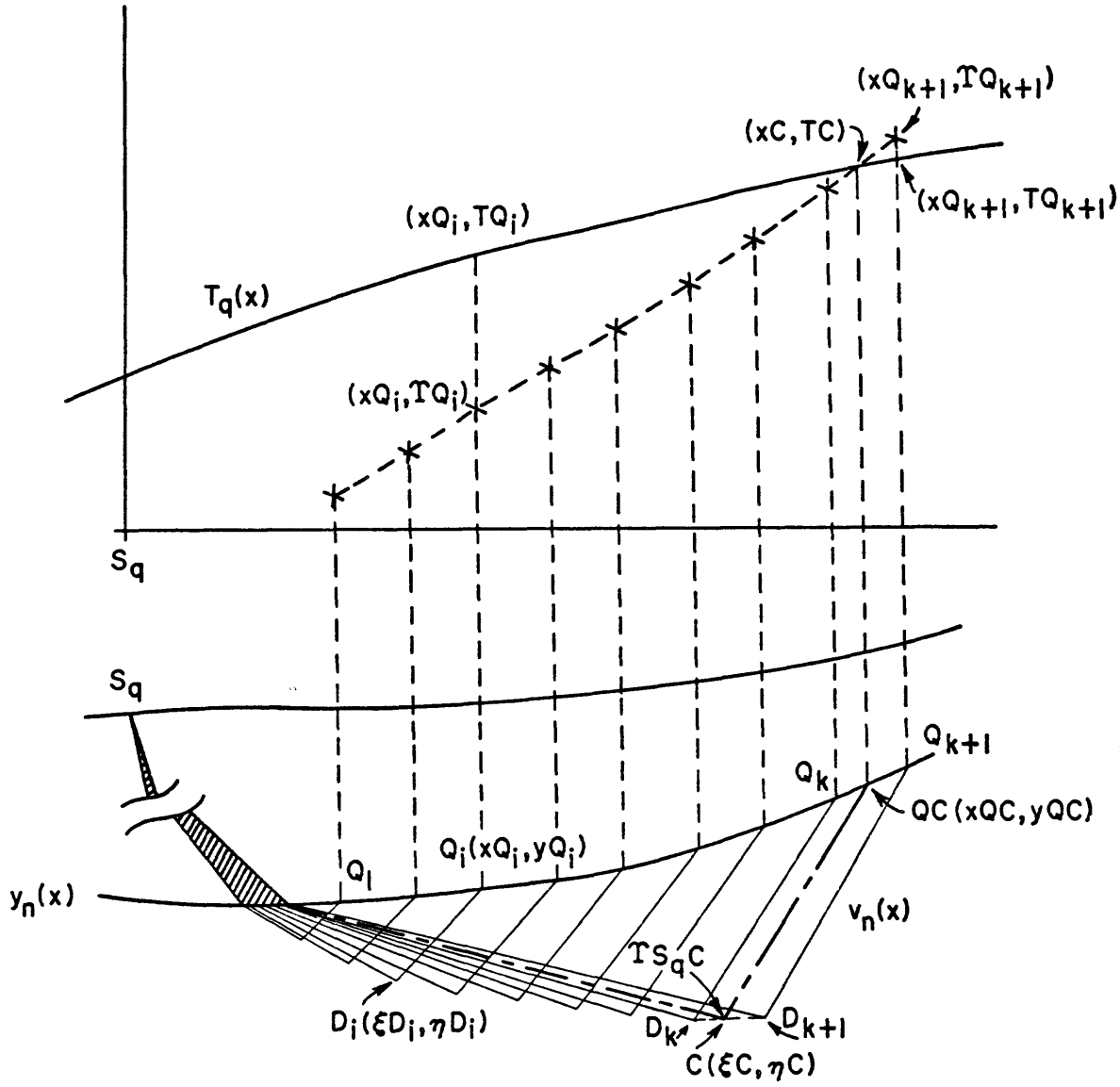

Figure 32.--Second sketch illustrating the parameters used in the algorithm for the critical reflection method discussed in the text.....

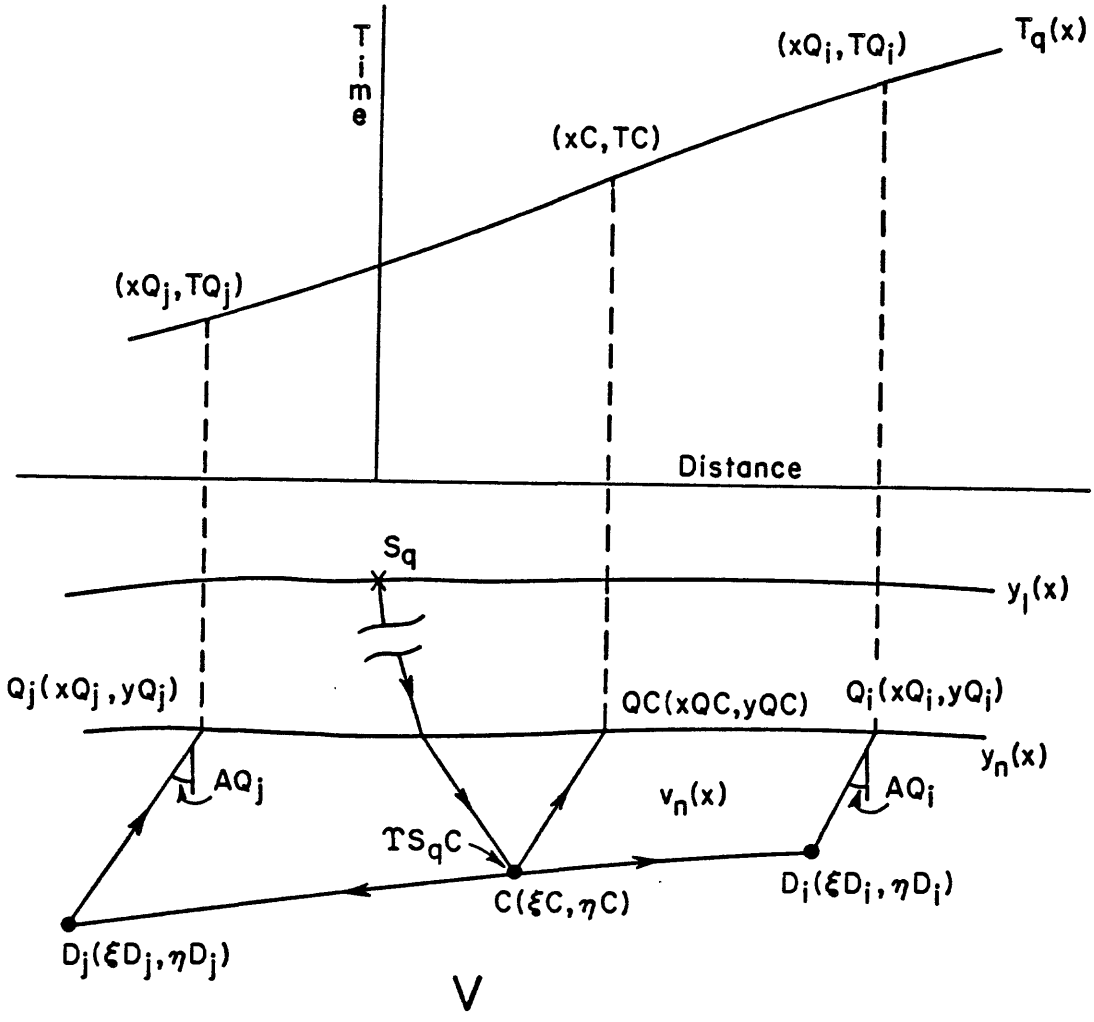

Figure 33.--Third sketch illustrating the parameters used in the algorithm for the critical reflection method discussed in the text.....

Thus one can write,

$$\frac{\{(xQ_i - \xi D_i)^2 + (yQ_i - \eta D_i)^2\}^{1/2}}{vD_i} + \frac{\{(\xi C - \xi D_i)^2 + (\eta C - \eta D_i)^2\}^{1/2}}{V}$$

$$= TQ_i - TS_qC \tag{34}$$

where vD_i is a representative velocity between D_i and Q_i obtained from $v_n(x)$. The expression for the emerging ray at Q_i is given by

$$\frac{yQ_i - \eta D_i}{xQ_i - \xi D_i} = \tan\left(\frac{\pi}{2} - AQ_i\right) \tag{35}$$

where AQ_i is obtained by applying equations (1) through (10). Equations (34) and (35) can be solved simultaneously for D_i (ξD_i, ηD_i). The set of D_i then define the (n+1)th horizon on the positive side of the critical reflection point.

On the side of C nearer the shotpoint, the travel-time between C and D_j is negative and hence the plus sign on the left side of the equation (34) must be replaced by a minus sign. Hence the equations to be solved to obtain the points $D_j(\xi D_j, \eta D_j)$ are

$$\frac{\{(xQ_j - \xi D_j)^2 + (yP_j - \eta D_j)^2\}^{1/2}}{vD_j} - \frac{\{(\xi C - \xi D^2) + (\eta C - \eta D^2)\}^{1/2}}{V}$$

$$= TQ_j - TS_qC \tag{36}$$

$$\frac{yQ_j - \eta D_j}{xQ_j - \xi D_j} = \tan\left(\frac{\pi}{2} - AQ_j\right) \tag{37}$$

It can be shown that the calculation of the critical reflection point C, using the critical reflection method is fairly insensative to errors in the value of V used for the velocity of the (n+1)th layer. However, large errors in the calculation of the D_i and D_j through the use of equations (34) through (37) may result from an incorrect choice of V, and hence these equations must be used with caution.

In most cases when reversed data are available and the reciprocal method is used for interpretation, the critical reflection method can also be used. Because the two methods employ unrelated algorithms, their combined use my serve as a quick check of results.

Arrival-time calculations from a velocity model

A scheme to calculate arrival-times from a particular computed velocity model is discussed in this section.

The method is described with the aid of figure 34, for which an arrival-time curve from the mth horizon with shotpoint at $S_q(xS_q,yS_q)$ is to be calculated. A systematic search is first made for that particular ray which, when traced into the subsurface from the shotpoint, intersects the mth horizon at the critical angle, thus defining the critical point $C(\xi C,\eta C)$ and its travel-time TS_qC. Points $E_i(xE_i,yE_i)$ are then selected at constant intervals along the mth horizon and their travel-times TCE_i along the shortest possible path (also assumed to be the shortest time path) within the mth layer between C and E_i is determined. From point E_i, the refracted ray returning to the surface is traced and its intersection point $R_i(xR_i,yR_i)$ and travel-time TE_iR_i from E_i to R_i determined. The arrival-time TR_i at R_i is then given by

$$TR_i = TS_qC + TCE_i + TE_iR_i \qquad (38)$$

The set of points (xR_i,TR_i) then define the refracted portion of the arrival-time curve illustrated in figure 34.

Arrival times of defracted rays are calculated by selecting appropriate diffraction points, such as $D(\xi D,\eta D)$ on the nth horizon (fig. 34). The travel-time to this point is approximately TS_qC+TCD. Arrivals at the surface are calculated by projecting rays upwards at numerous angles and tracing the resulting up-going rays to the surface resulting in intersection points $Q_i(xQ_i,yQ_i)$ and travel-times TDQ_i. The arrival-times TQ_i for the diffracted arrivals are given by

$$TQ_i = TS_qC + TCD + TDQ_i$$

and the diffracted arrival-time curve is illustrated by the dashed line in figure 35. The arrival-time curve for the nth horizon is the minimum of the refracted and diffracted curves.

The method of calculating the point of entry $C(\xi C,\eta C)$ into the nth layer is illustrated with the aid of figure 35a. Initially a ray is traced into the subsurface using equations (1) through (21), with submergence angle A (e.g., C=A in equation (8a)). The angle A is set equal either to zero or a trial value is chosen which will cause the submerging ray to impinge on the nth horizon at an incidence angle less than the critical angle (e.g., the absolute value of the term in brackets in equation (20) is less than 1). The angle A is then incremented by ΔA, and the process repeated using the angles $A+\Delta A$, $A+2\Delta A$,$\cdots\cdots A+n\Delta A$, until a ray is found (Ray D_n in fig. 35a) for which the angle of incidence with the nth horizon is greater than the critical angle. In the figure, the intersection point of this ray with the

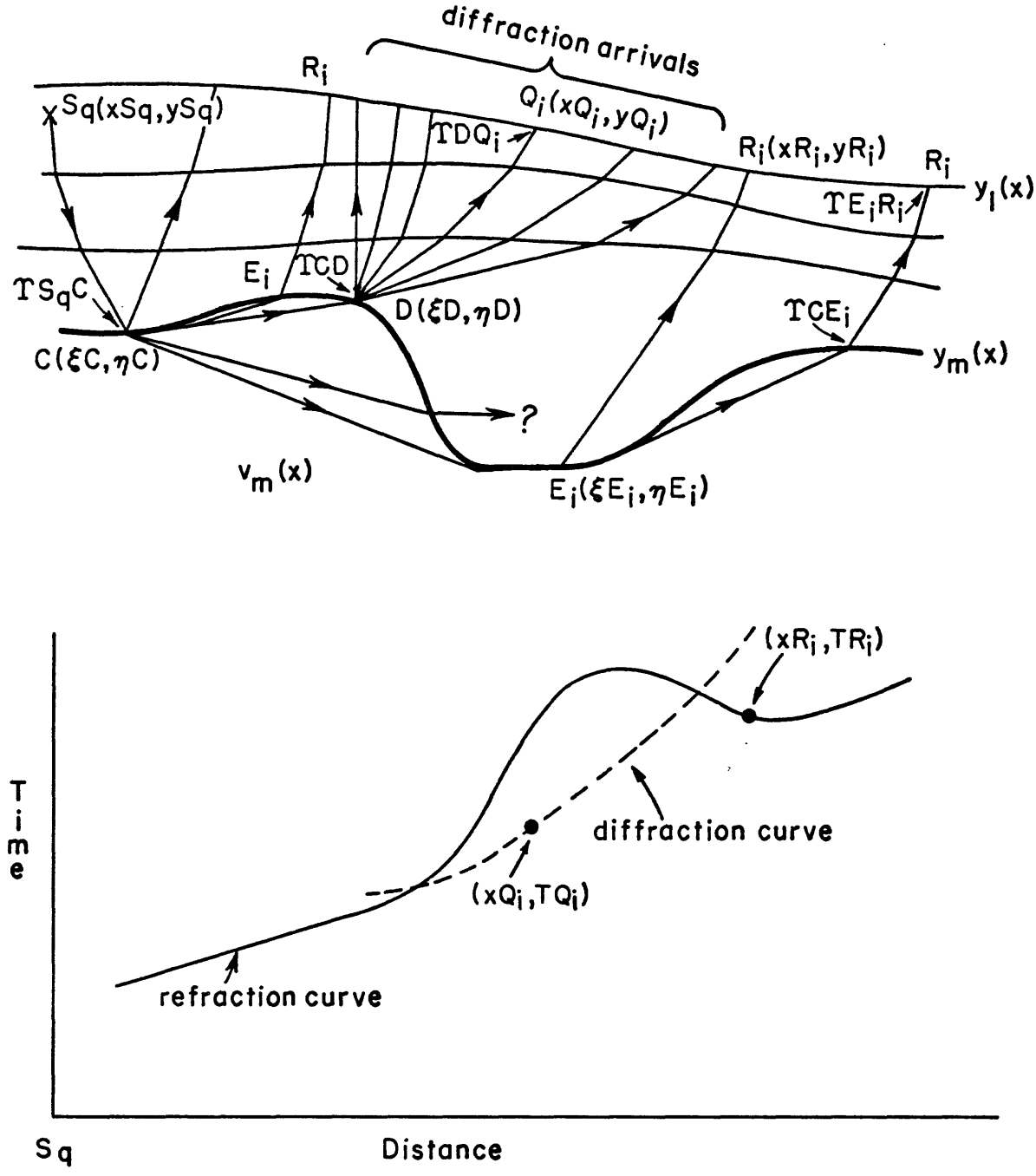

Figure 34.--First sketch illustrating the parameters used in the algorithm for calculating an arrival-time curve from a velocity model discussed in the text.....

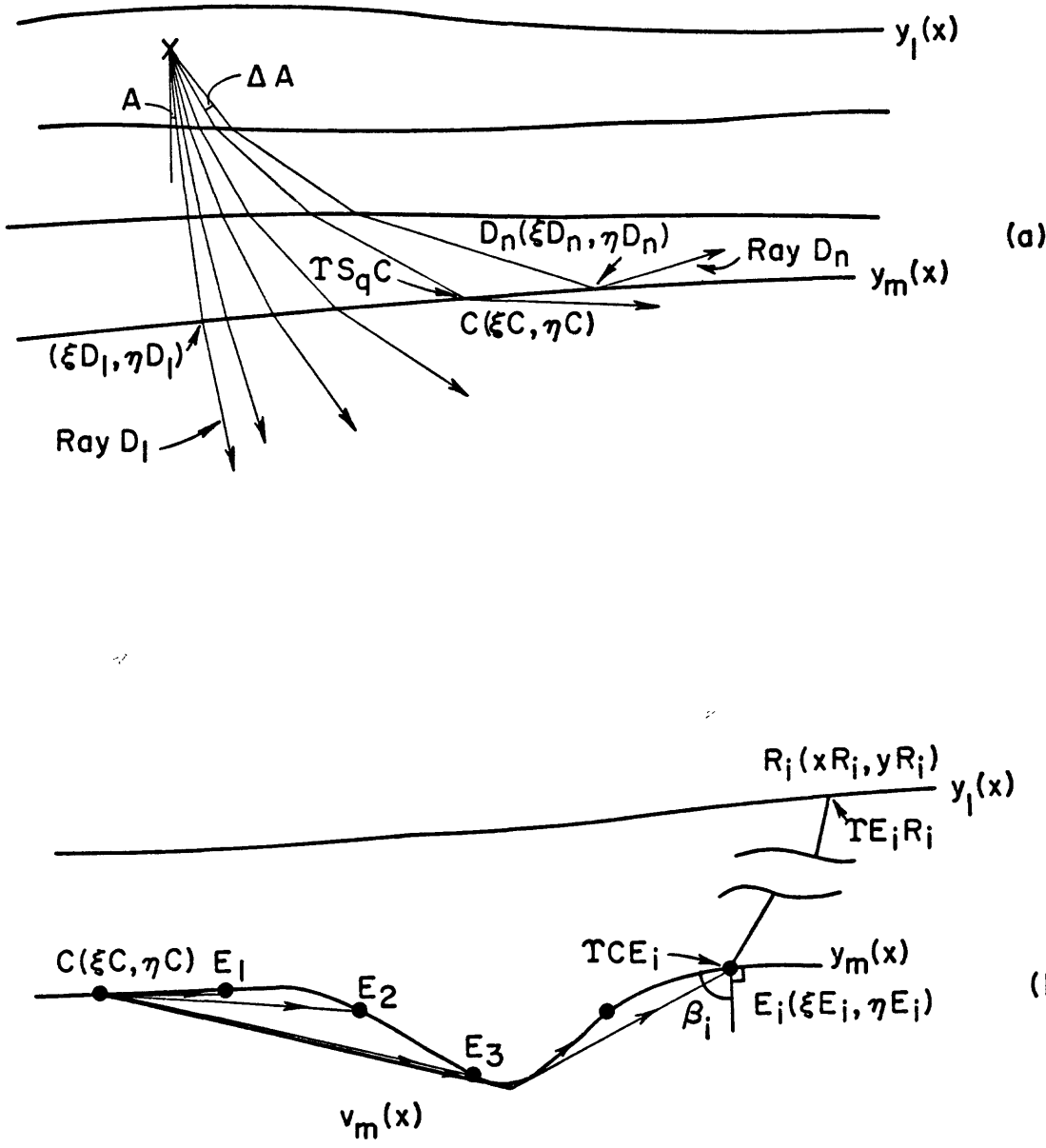

Figure 35.--Second sketch illustrating the parameters used in the algorithm for calculating an arrival-time curve from a velocity model discussed in the text.....

nth horizon is labeled $D_n(\xi D_n, \eta D_n)$. If ΔA has been taken small enough the intersection parameters of the prior ray (e.g., ray n-1) may be assumed adequate for the critical refraction point $C(\xi C, hC)$ with travel-time TS_qC.

To calculate the lateral travel-time TCE_i between C and successive points E_i on the nth horizon, an algorithm is used which seeks the shortest path (fig. 35b) between two points without entering the overlying layer. TCE_i is computed by dividing these shortest paths by the approximate velocity values derived from $v_n(x)$. The incident angle β_i of this "shortest" ray with the mth horizon at E_i is also computed.

The travel-time TE_iR_i from E_i to the surface and the corresponding surface point $R_i(xR_i, yR_i)$ (fig. 35b) are determined by applying equations (1) through (21), using the incident angle β_i as the initial emergence angle.

THE COMPUTER PROGRAMS AND SAMPLE PROBLEMS

Data storage files

Data are stored in three files (or segments) each of which have the same arbitrary file name, but different extensions (e.g., filnam.lay, filnam.org, filnam.com). Data are usually input into these files through subroutines DATAIN or BUILD. The variable names used for data are contained in the common block datal and are initially declared in the main driver program SEISMC. The variable names in this common block, to be discussed under the following subheadings, will be used throughout these discussions.

The data in the file with extension .lay (henceforth to be referred to simply as filnam.lay) describe the model used to define the velocity function overlying the horizon for which depths are to be calculated. Thus, for example, if the (n+1)th horizon is being calculated, this file must contain a model consisting of at least n layers. Layers may not pinch out, but they can be made arbitrarily thin.

Filnam.org contains data in the form of time-distance data sets. Each data set refers to a particular shotpoint, though more than one data set may refer to the same shotpoint, such as cases where more than one set of arrival times are picked. It is important to note that, for convenience, distance values are initially input into this file in a different configuration (intervals between successive data points or seismometers rather than absolute distances) than is used for computations. A subroutine named PERP must be invoked to convert data point intervals into data point distance coordinates before any computations are made.

Filnam.com may contain time-distance data sets which have been combined from others to form extended data sets in a manner prescribed by the user through the use of subroutine BUILD. The only difference between data contained in filnam.org and filnam.com are the dimensions and the variable names. Data from either file are used in an identical way for making calculations.

Filnam.lay--; The programs require a model (velocity distribution) to make calculations. The model is usually constructed from well log information, or from previous refraction calculations. The ground surface is considered as horizon 1. Thus to calculate the thickness of the first layer (e.g., horizon 2) the topography of horizon 1 and the velocity distribution of the first layer must have been input.

An example of a model consisting of four horizons is shown in Figure 36. Each horizon is represented by a series of connected and straight line segments separated into compartments. Compartment boundaries are vertical and represent locations at which a horizon changes slope and/or the corresponding layer changes velocity. Hence, a compartment boundary represents a lateral change, either in the slope of the horizon or the layer velocity or both.

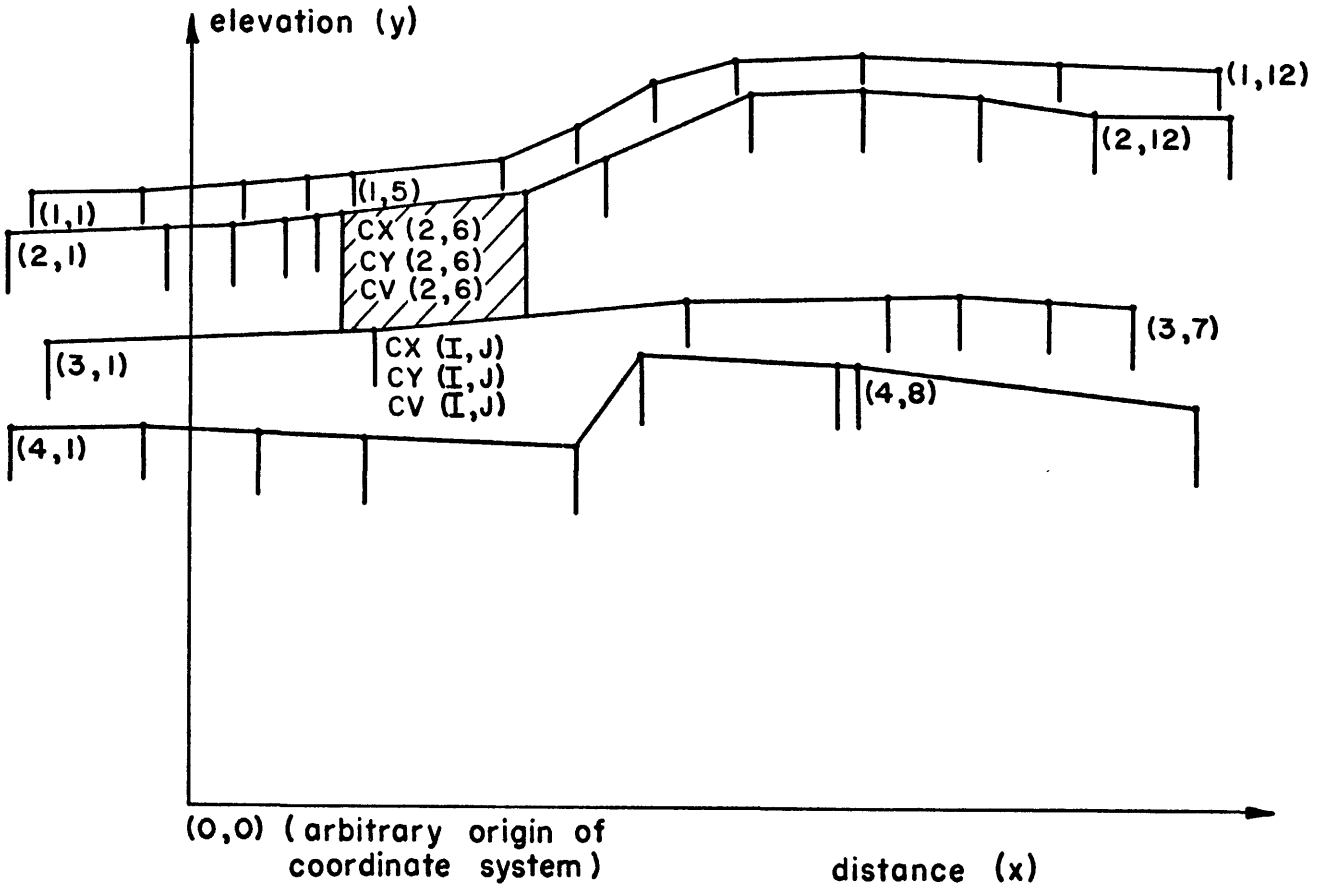

Figure 36.--Example of a velocity model consisting of four horizons.....

The variable names representing horizons and layers are doubly subscripted, with the first subscript I, denoting the horizon or layer index, and the second J, the compartment index. Each compartment boundary is defined by three coordinates, a distance CX(I,J) and an elevation CY(I,J) which define the upper left-hand compartment corner and a velocity CV(I,J) which defines the compartment velocity. Thus the shaded compartment in figure 36 has a velocity given by CV(2,6) and the spatial coordinates for its upper left-hand corner are CX(2,6), CY(2,6). The variable NC(I) denotes the number of compartment boundaries for the Ith horizon. Thus in figure 36, NC(3) = 7.

The number of horizons and compartments permitted for any model are controlled by the dimension statements in common block, datal. As presently set up, 10 horizons, each with 50 compartments are permitted.

In practice, it is important to construct a model which extends well beyond the limits of the arrival-time data and offset shotpoint locations. The programs construct numerous rays during execution, and all of them (including strays) should remain within the confines of the model.

Filnam.org--; The time-distance (t-x) data sets are tied to the velocity model in filnam.lay through the use of the same distance coordinate system. Thus the origin representing distance for the arrival-time data in filnam.org. and the model in filnam.lay are the same. It is also assumed that arrivals were recorded on horizon 1, and that the seismometer spreads are along straight lines.

Figure 37a illustrates a t-x data set for the spread configuration shown in figure 37b which consists of 11 data points. Each t-x data set is represented by a shotpoint index K (fig. 37b); the number of t-x data points, NG(K); shotpoint distance coordinate, SX(K); shot depth, SD(K); and the perpendicular offset distance of the shotpoint from the line, OFFSET(K). The variable names representing the distance and arrival-times of t-x points are doubly subscripted, with the first subscript K denoting the shotpoint index, and the second J the t-x point number index. Thus the distance coordinates of the successive points of a t-x plot from left to right are given by GX(K,J) (fig. 37b) and the corresponding arrival-times by GT(K,J) (fig. 37a).

For input of t-x data sets (in subroutine DATAIN) intervals between successive data points (seismometers) are used instead of the true distance coordinates, except for the first data point, GX(K,1), (fig. 37b) for which the distance from the arbitrary origin is used. Subroutine PERP calculates the distance coordinate for each of the remaining data points using GX(K,1) and the interval values. Thus, the variables K,NG(K), SX(K), SD(K), OFFSET(K), GX(K,J), and GT(K,J) completely identify the spread configuration and the arrival-times. The dimension of K is 50, and of J is 24.

Filnam.com--; Combined t-x data sets are contained in file filnam.com. These data are stored and used in exactly the same way as data

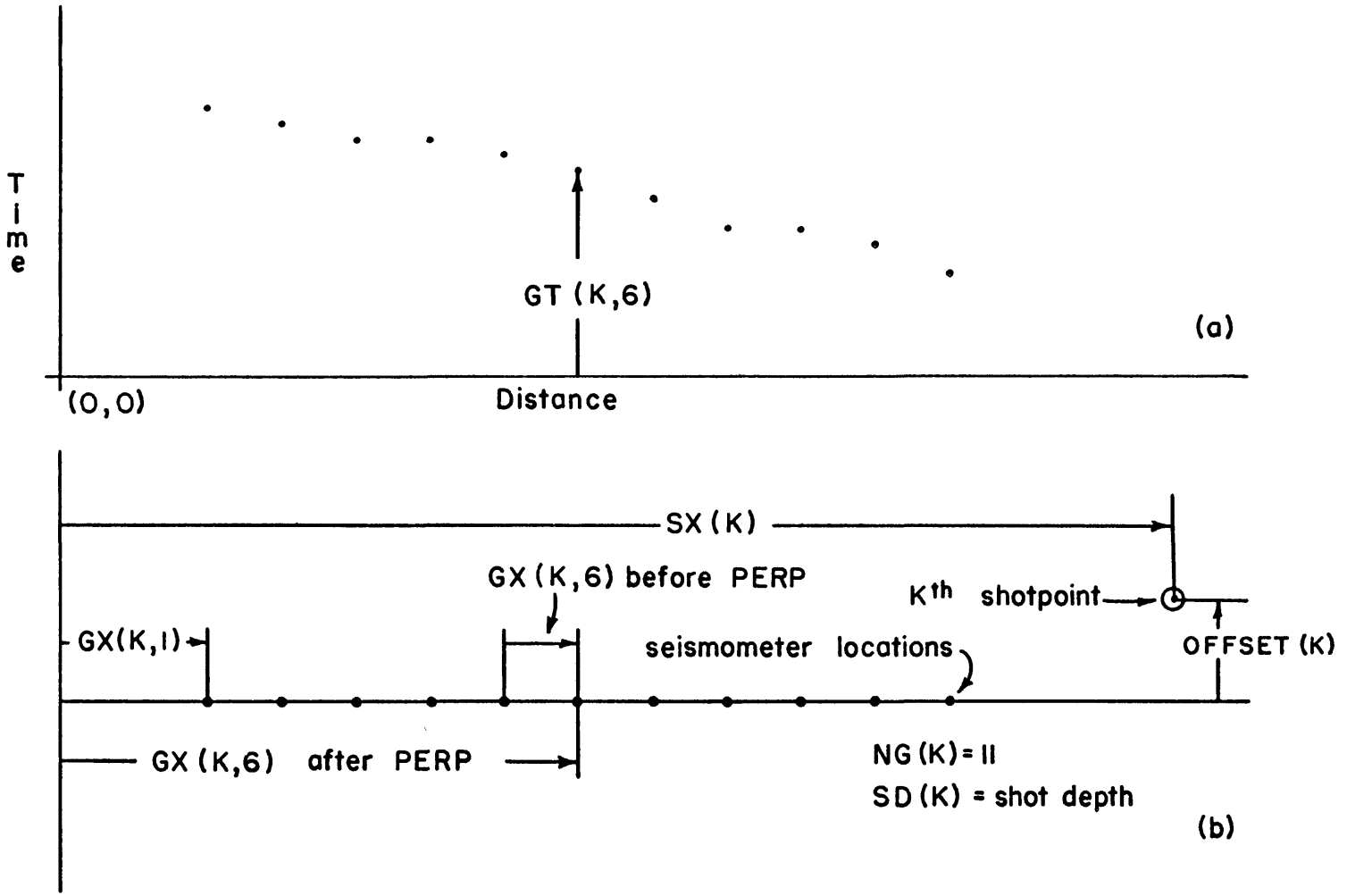

Figure 37.--Illustration of a t-x data set (fig.a) for the spread configuration in figure b as discussed in text.....

in file filenam.org. The only difference between the two files are in the dimensions. In filnam.com, the dimension of K is 10 and of J is 100. Corresponding variable names between the two files are;

	filenam.org	filenam.com
shot index	K	K
number of data points	NG(K)	NGC(K)
shot distance coordinate	SX(K)	SCX(K)
shot depth	SD(K)	SCD(K)
perpendicular offset distance	OFFSET(K)	OFFSC(K)
data point distance coordinate	GX(K,J)	GXC(K,J)
data point arrival-time	GT(K,J)	GTC(K,J)

Sign convention

As indicated previously, the normal cartesian system is used throughout, with elevation (y) positive upwards and distance (x) positive to the right. Occasionally, program execution requires that a direction be assigned to a ray path. The usual cartesian system is again followed. A ray with positive slope is assigned a positive angle (no matter which direction it propogates, up or down) and with negative slope, a negative angle.

Extended and artificial arrival-time data

Seismic-refraction data are seldom complete. Complete data from all refracting horizons can only be obtained through innumerable shotpoints and combining the data (subroutine BUILD) from each. The luxury of complete data can usually be had for only one or two important horizons. Frequently, seismic-refraction surveys are done without obtaining complete data from any horizon.

Given incomplete data from a given horizon, arrival-time values must be derived or estimated by extending the data (see section Manipulating Arrival-Time Curves). For one-way data, arrivals must extend back to the shotpoint (somewhat analogous to intercept time calculations), and for reversed data, they may have to be extended forward at least as far as the reciprocal or reversed shotpoint. Provisions are made within the programs to extend data sets forward or backward as far as desired using connected straight line segments when data have not been recorded. In fact, one is free to not use any actual data points but instead an interpretation of them obtained by constructing straight line segments "through" plots of the t-x points contained either in filnam.org or filnam.com. We call these constructions "artificial data". This procedure is recommended in cases of limited data from particular horizons, or for arrival-time data which are otherwise too scattered for satisfactory results, or for hidden layers.

General program structure

A driver program named SEISMC must be invoked to execute any of the other programs (except DIGIT) which are in the form of subroutines.

For each subroutine requiring user input, there is a program entry list which describes the input in detail. All entry lists are included together in Appendix A because they must be referred to frequently during program execution. Therefore, they are not included in the following discussions of each subroutine. The input in some cases, is facilitated by writing it out, for which input forms are also included in Appendix A.

Because of the considerable manipulation of data, other factors related to incorrect interpretation, conditions which lead to errors, may arise. Therefore, many of the subroutines contain "condition" statements which print out and refer to a written explanation of the error condition having occurred. These condition statements and brief explanations of them are also included in Appendix A.

It is suggested that Appendix A be copied for convenient referral during the following discussion of subroutines.

Mainline driver program SEISMC

Execution of any of the subroutines is done by first executing the mainline program named SEISMC. This program automatically executes subroutine INPUT which inputs existing data files from disk into memory. The program then prompts the user to execute a subroutine which may be any of the following: RAYTR, PLOTX, RECIP, DATAIN, CRIT, OUTPUT, INPUT, FUDGE, PERP, PICK, TERM, BUILD, or DUM1. Once a subroutine is executed, control returns to the mainline. Normal exit is achieved by typing OUT instead of one of the subroutine names, causing execution of subroutine OUTPUT which outputs the data in memory back to disk before program termination.

Table 2 is an example of terminal printout during execution of SEISMC. Six steps are illustrated as follows:

1. Execute program SEISMC.
2. Subroutine INPUT is automatically executed which prompts user to request a data file name. The file name requested here is TEST. Because no data files with this name exist, null files TEST.ORG, TEST.COM and TEST.LAY are created and input into memory.
3. Program prompts user to type a subroutine name. The subroutine DUM1 is selected which does nothing except return control back to mainline.
4. Program prompts user to type a subroutine name. The subroutine chosen is INPUT, which has already been executed.
5. Subroutine INPUT prompts user to select a data file name for input into memory. File name TEST is chosen, but because these files have already been created (step 2.). The user is simply informed that input is completed.
6. Normal exit is requested by typing OUT instead of a subroutine name. Subroutine OUTPUT is automatically executed before end of program.

```
1. seismc

2. type input file name: test
   new file--test.org created.
   new file--test.com created.
   new file--test.lay created.

   input complete from 3 dsk files named     test
3. type subroutine: dum1
4. type subroutine: input

5. type input file name: test

   input complete from 3 dsk files named     test
6. type subroutine: out
   output of 3 files to dsk complete

   end of program

   STOP
```

Table 2.--Example of terminal printout during execution of mainline driver program SEISMC and subroutines DUM1 and INPUT.....

Subroutine DUM1

Subroutine DUM1 is a do-nothing subroutine which simply returns control to the mainline.

Subroutine INPUT

Subroutine INPUT reads the data in the requested files, (e.g., filnam.org, filnam.com and filnam.lay) from disk into memory.

The file name may be up to only six characters long.

An example of execution of subroutine INPUT has been given under subheading Mainline driver program SEISMC and table 2.

Subroutine OUTPUT

Subroutine OUTPUT outputs the data in memory into the files filnam.org, filnam.com and filnam.lay.

Subroutine OUTPUT is automatically executed during normal exit from the mainline program SEISMC (table 2) and also within some other subroutines (DATAIN, BUILD, and PERP) which read data into memory. (Note: Message format has been changed since these examples were run.) There is seldom any need to command execution of this subroutine OUTPUT. (In the revised version, if files to be output have not yet been created, they are automatically created.)

Subroutines DATAIN and PERP

Subroutine DATAIN permits reading of either t-x data or velocity model data into memory. Upon request, it executes subroutine OUTPUT.

(Two consistent sets of units are millisecond, meter and km/sec, or millisecond, feet, and kilofeet/sec. In all the discussions to follow, units will not be specified and either of these two may be assumed.)

Data may be read into memory directly from a terminal. However, when a large quantity of t-x data renders terminal input impractical, arrival-times may also be digitized automatically and input into filnan.org via punched cards. On the other hand, velocity model data for filnam.lay is always input directly from the terminal. For filnam.com, data is generally input via subroutine BUILD, but they may be input from the terminal through subroutine DATAIN as well.

The following are examples of data input using subroutine DATAIN.

Figure 38a shows a t-x plot consisting of two shotpoints with indices K=1 and K=2 at distance coordinates of -1 and 63 units respectively. (Normally plots are not prepared until after data have been input). The shot-seismometer configuration is shown in figure 38. The data set for K=1 contains 13 points, and for K=2 it contains 11 points. Shot depths are 5 and 2, and perpendicular offsets are 4 and 0 respectively.

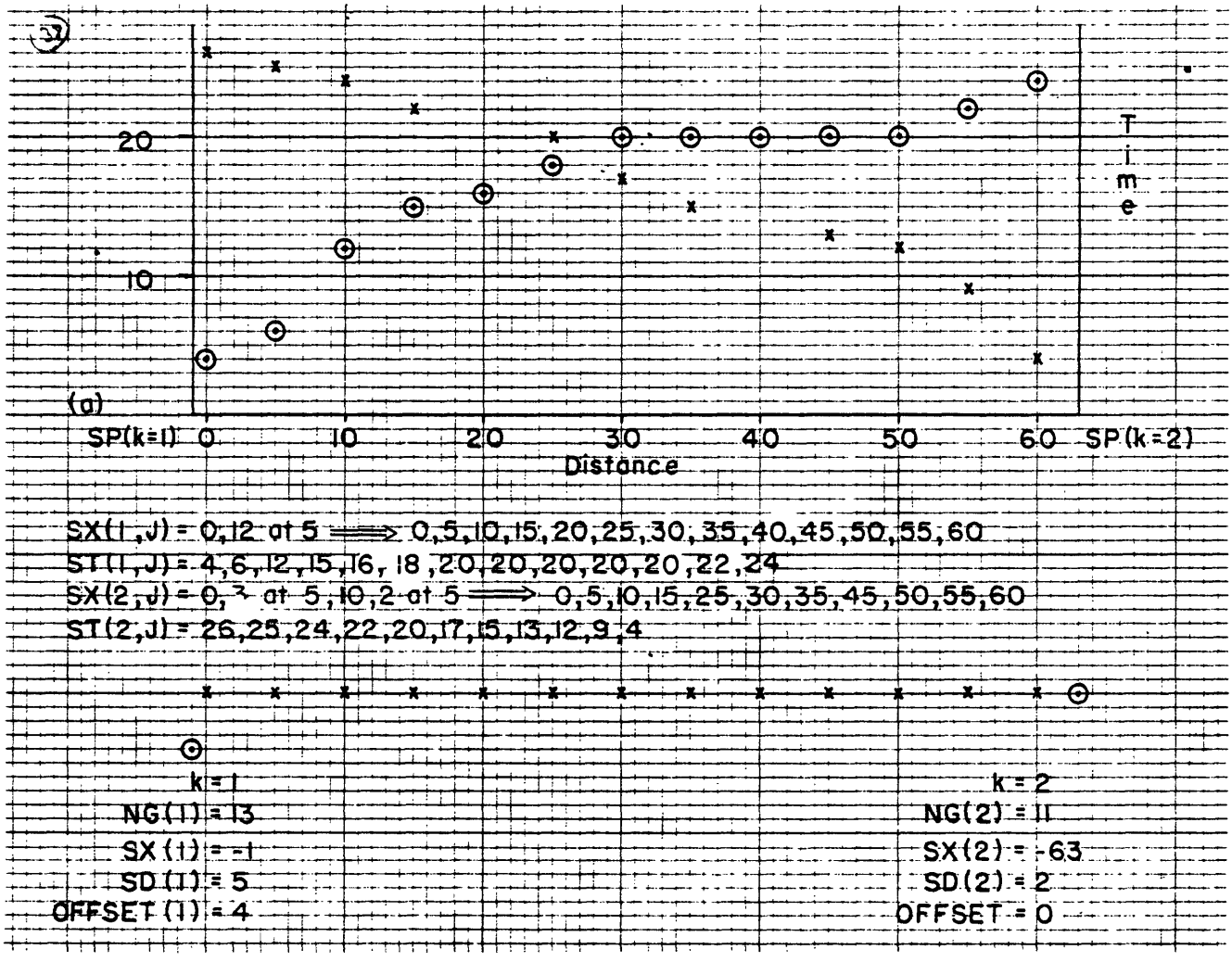

Figure 38.--T-x̄ data sets input into memory as described in text.....

Table 3.--Execution of subroutine DATAIN for the t-x values given in Figure 38.....

```
1. seismc
2. type input file name: test

   input complete from 3 dsk files named     test
3. type subroutine: datain
4. type file extension either org,com,or lay: org
5. enter  the word digit   if data digitized.
   anything else if not: no
6. datain entry 1; k*,ng(k),sx(k),sd(k),offset(k)
   *1 13 -1 5 4;
   gx(k,j):0 5 0;
   gt(k,j):4 6 12 15 16 18 20 20 20 20 20 22 24;
   for shot index=    1 xlast=       60.00
7. next set of data please. 0; or -n; for return to mainline
   datain entry 1; k*,ng(k),sx(k),sd(k),offset(k)
   *2 11 63 2 0;
   gx(k,j):0 5 5 5 10 5 5 10 5;
   gt(k,j):26 25 24 22 20 17 15 13 12 9 4;
   for shot index=    2 xlast=       60.00
8. next set of data please. 0; or -n; for return to mainline
   datain entry 1; k*,ng(k),sx(k),sd(k),offset(k)
   *0;
   output of 3 files to dsk complete
   datain complete
   type subroutine: QUIT
```

Table 4.--Printout of the two t-x data sets in file test.org input in Table 3.....

```
print test.org

                    test.org

 1 13       -1.00        5.00        4.00
      0.00        4.00
      5.00        6.00
      0.00       12.00
      0.00       15.00
      0.00       16.00
      0.00       18.00
      0.00       20.00
      0.00       20.00
      0.00       20.00
      0.00       20.00
      0.00       20.00
      0.00       22.00
      0.00       24.00
 2 11       63.00        2.00        0.00
      0.00       26.00
      5.00       25.00
      5.00       24.00
      5.00       22.00
     10.00       20.00
      5.00       17.00
      5.00       15.00
     10.00       13.00
      5.00       12.00
      0.00        9.00
      0.00        4.00
```

Appendix A contains the entry list and input form showing the sequence and format for data input for files filnam.org, filnam.com and filnam.lay directly from the terminal. Entry 1 refers to both filnam.org and filnam.com and entry 2 refers to filnam.lay. The execution of subroutine DATAIN for the values in figure 38 is shown in table 3 as eight steps described below. The file name chosen is TEST.

1. Execute SEISMC.
2. Declare the file name selected.
3. Declare the subroutine to be executed.
4. Declare the file name extension (either org, com, or lay)
5. Declare that data will be input from terminal.
6. Enter the data for shot index k=1 using the format of entry 1 of the entry form and the values shown in figure 38. Terminal output at this point indicates the distance coordinate of the last (13th) data point which provides a check of the input.
7. Repeat step 6 for shot index k=2.
8. Type 0; (or some negative integer -n;) for execution of subroutine

A printout of the two t-x data sets now contained in the test.org are shown in table 4. Data point distance intervals are in the left column. The distance coordinate of the first point for both data sets is zero, followed by a 5 indicating a 5 distance unit interval between data points 1 and 2. Zeros in the first column imply that the interval is the same as that between the two preceding data points. Arrival-times are in the second column. After inspection of the file filenam.org and values of xlast (step 6 above) for accuracy, the actual data point distance coordinates are computed from the distance coordinates of the first data point and the distance intervals thereafter by executing subroutine PERP. Correction for perpendicular offset is achieved through the Pythagorean theorem, effectively shifting the distance coordinate of each data point so that a t-x plot will show true distances between shotpoint and data point. The distance correction is usually minimal for distant data points and only important for those near the shotpoint.

Execution of PERP for the data in table 4 is illustrated in table 5 as two steps.

1. Execute SEISMC and declare the file name, test.
2. Declare the subroutine to be executed (PERP) and the file name extension (org). Output of the corrected data files through subroutine OUTPUT is automatic.

Table 6 shows the data now contained in file test.org. Note the corrected data point distance coordinates in the left column and the change in the perpendicular offset to -1 which insures that corrections will not be applied once more if PERP must be executed again for additional data. Sometimes it may be desired to input actual corrected distances instead of intervals in subroutine DATAIN, in which case a negative number is input for the perpendicular offset.

T-x plots of the data sets in filnam.org are usually prepared at this juncture by executing subroutine PLOTX. A manual plot of file test.org is

Table 5.--Execution of subroutine PERP for the t-x data sets shown in Table 4.....

```
seismc

type input file name: test

input complete from 3 dsk files named     test  type  subroutine:
perp  type extension--either org or com: org output of 3 files to
dsk complete type subroutine: QUIT
```

Table 6.--Printout of the two t-x data sets in file test.org after execution of subroutine PERP.....

```
print test.org

                    test.org

 1 13       -1.00          5.00        -1.00
      3.12          4.00
      6.21          6.00
     10.70         12.00
     15.49         15.00
     20.38         16.00
     25.31         18.00
     30.26         20.00
     35.22         20.00
     40.19         20.00
     45.17         20.00
     50.16         20.00
     55.14         22.00
     60.13         24.00
 2 11       63.00          2.00        -1.00
      0.00         26.00
      5.00         25.00
     10.00         24.00
     15.00         22.00
     25.00         20.00
     30.00         17.00
     35.00         15.00
     45.00         13.00
     50.00         12.00
     55.00          9.00
     60.00          4.00
```

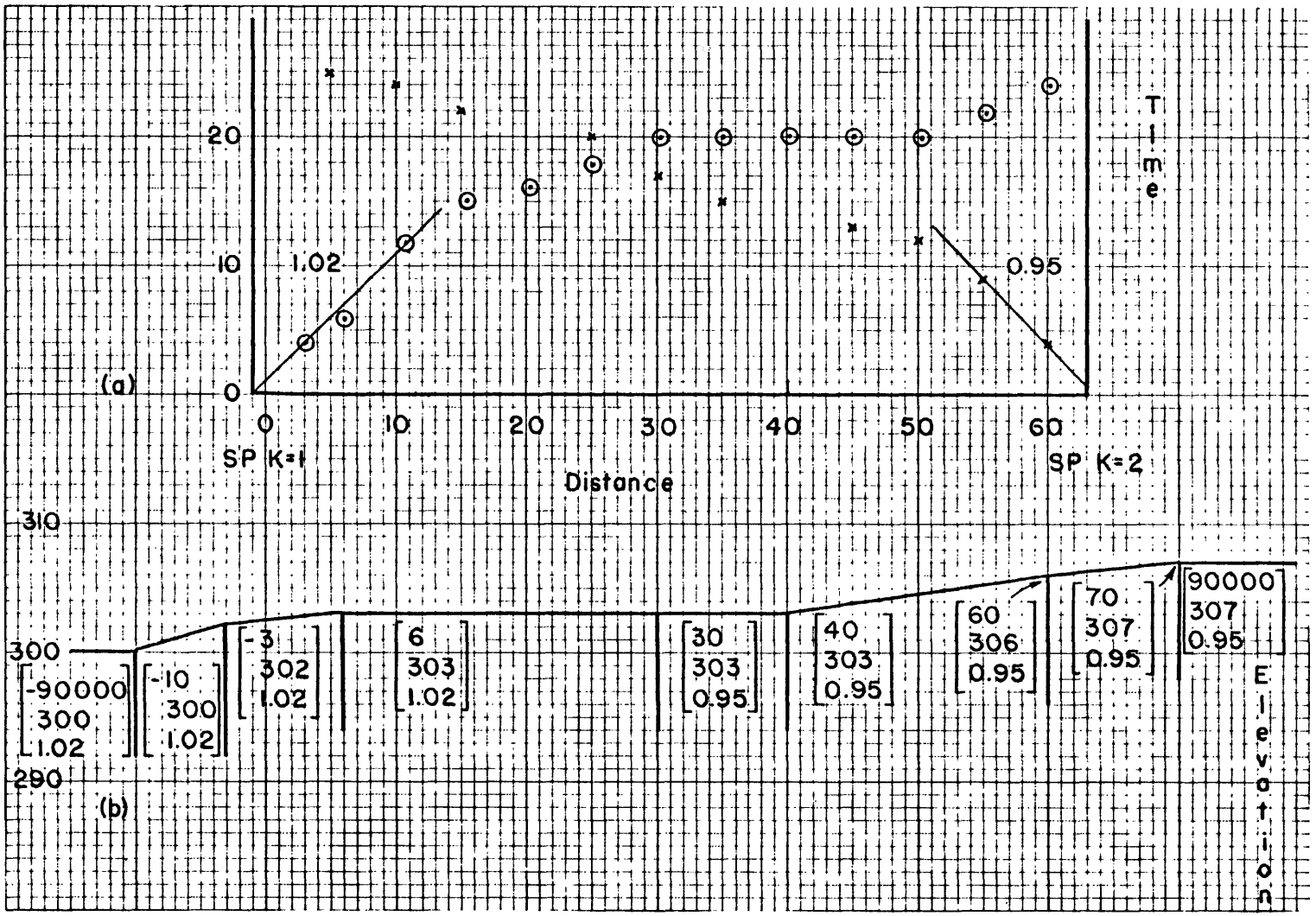

Figure 39.--T-x plot (fig.a) of data in file test.org after execution of subroutine PERP. Figure b shows profile of horizon 1 to be input into file test.lay.....

shown in figure 39a. Note the change in distance coordinates, compared with fig. 38, for the data points near the shotpoint for the shot with a perpendicular offset (K=1).

In order to start depth and velocity calculations of these data, a velocity model consisting of one horizon must be input into file data.lay. Therefore, the velocity of the first layer must be determined from the plotted values. Drawing straight lines through the initial few points of figure 39a at both shotpoints implies a velocity of approximately 1.02 at k=1 and 0.95 at k=2.

The profile of the surface (horizon 1) has been plotted in figure 39b. In addition, compartment boundaries have been constructed, and the coordinates assigned to each compartment are shown in brackets. Note that the velocity of layer 1 was arbitrarily changed from 1.02 to 0.95 at the 30-unit distance coordinate. The change in velocity could have been prorated from shotpoint 1 to shotpoint 2 in any manner desired. Actually, we could have used a constant velocity of 1.0.

Table 7 illustrates terminal input of horizon 1, shown in figure 39b, into file test.lay as three steps.

1. Execute SEISMC, declare file name, declare subroutine DATAIN to be executed and declare file extension lay.
2. Using the format of entry 2 in the DATAIN entry list (Appendix A) and the values posted in figure 39b, enter the data for horizon one.
3. Type 0; for output to disk and return to mainline.

A printout of the data now contained in file test.lay is shown in table 8.

In cases of numerous seismograms, manual reading of arrival-times and subsequent entry from a terminal is impractical. The remainder of this section, which includes figure 40 and tables 9 through 14, outlines a method we have recently used for automatic digitization of seismograms using punched cards, conversion of digitized values to arrival-times and entry into filnam.org. It has proved cumbersome and has been modified. Figure 40 represents an example of a seismogram consisting of 4 traces, a time break and timer lines at 10 millisecond intervals. Five auxiliary points, A through E picked on timer-lines are read manually. Points A, B, and C are used to establish the seismogram position when placed on a digitizing board and also the time scale. The times for A and B are shown as 15 and for C as 75. Points D and E, at 25 and 65 respectively, are check points used to estimate the digitizing accuracy. The seismogram is placed on the digitizing board interfaced with a keypunch. The digitizing process consists of three steps. First, the points A, B, and C are digitized and their time values key-punched manually. Second, the arrival-times, shown picked with tick marks are digitized. Third, the check points D and E are digitized and their time values key-punched manually. In addition to these cards, also included in the deck are cards specifying the file name and the shotpoint and data point parameters.

Table 7.--Execution of subroutine DATAIN for horizon 1 shown in Figure 39b.....

```
1. seismc

   type input file name: test

   input complete from 3 dsk files named     test
   type subroutine: datain
   type file extension either org,com,or lay: lay
2. datain entry 2; i*,nc(i):1 9;
   cx(i,j):-90000 -10 -3 6 30 40 60 70 90000;
   cy(i,j):300 300 302 303 303 303 306 307 307;
   cv(i,j):1.02 1.02 1.02 1.02 .95 .95 .95 .95 .95;
3. next layer please.  type 0; or -n; for return to mainline
   datain entry 2; i*,nc(i):0;
   output of 3 files to dsk complete
   datain complete
   type subroutine: QUIT
```

Table 8.--Printout of velocity model data for horizon 1 as input in Table 7.....

```
print test.lay

                    test.lay

 1  9  -90000.00      300.00        1.02
    -10.00      300.00        1.02
     -3.00      302.00        1.02
      6.00      303.00        1.02
     30.00      303.00        0.95
     40.00      303.00        0.95
     60.00      306.00        0.95
     70.00      307.00        0.95
  90000.00      307.00        0.95
```

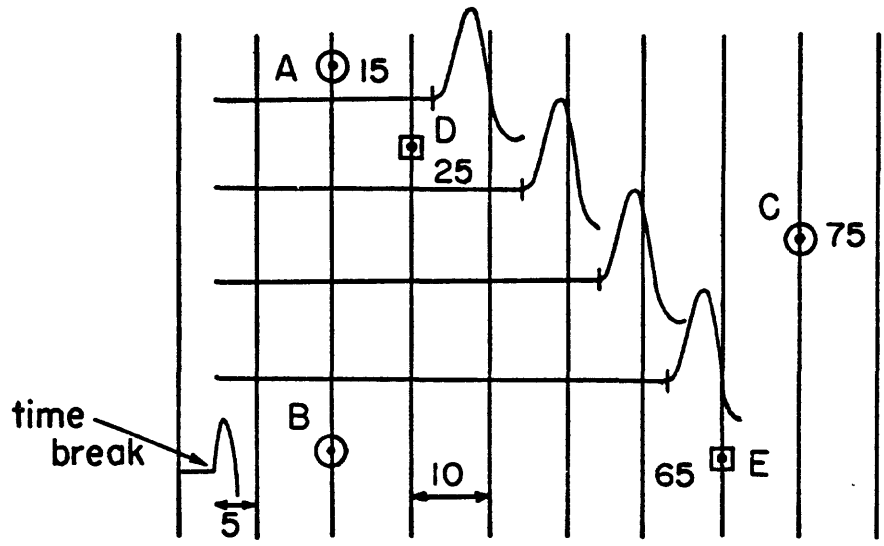

Figure 40.--Hypothetical seismogram of four traces to illustrate automatic digitization of arrival-times described in text.....

Table 9.--Example of digitized arrival-times and shotpoint and seismometer parameters contained in a file named digit.in.....

```
print digit.in

                    digit.in

1- club
   1 15 -51 7 0;
   0 120 120 120 120 120 120 120 120 120 120 1080 240 120;
   15 05131603 05100832   23 19631271 1223
      05321578 06081553 06831521 07641491 08311463 08881434 09431401 10011373
      10581341 11101314 17961049 18520985 18840957 19000922 19210899
7- 0633149819011046  123 1173
   2 17 3015 15 0;
   0 120 120 120 120 120 120 120 120 960 120;
   17 06691607 06750865  103 20861297 1273
      20261615 20021583 19741552 19481528 19221496 18931468 18691438 18521392
      18131371 11511132 10791101 10231075 09691042 09121011 08640985 08050945
      07270924
   0877110320251542  273 1223
   3 23 -51 9 0;
   2880 120 120 120 120 240 120;
   23 08301540 08340783 1165 14971196 1715
      08941524 09151496 09351465 09701438 09931410 10521348 11081315 11301286
      11421261 11631227 11861194 12061168 12321138 12591108 12851078 12971054
      13271024 13520995 13790958 13930931 14150904 14400876 14610845
   0954136514360988 1265 1665
   4 17 3015 9 21;
   2880 120 120 120 120 120 120 120 120 120 120 120 120 960 120;
   17 07151534 07130688   18 21051207 1168
      07971511 07211485 07741457 08501424 09281395 10011366 10581336 11111304
      11591275 12221249 12731218 13371178 13981154 20100915 20380887 20640861
      20860827
   0897135219810850  168 1068
29-0;
   mplace
   1 20 -3000 7 0;
   0 120 120 120 120 120 120 120 120 120 120 118 118 114 585 110 110 110 120 120;
   20 09631504 09630755  910 17501166 1560
      10251489 10521463 10801434 11111408 11451378 11721349 12021319 12211287
      12501258 12821235 13081200 13301170 13591139 13951107 15930957 16060926
      16320903 16530873 16740840 17000817
   1025141916900914  960 1510
   2 10 -175 12 68;
   0 120 120 120 120 120 240 1515 110 120;
   10 05721522 05700795  108 20841189 1358
      05931505 06711474 07411447 07921425 08401386 08941355 10111299 19830897
      20140880 20460837
   0634139620240954  158 1308
   3 23 -175 13 0;
   2792 120 120 120 120 120 120 120 120 120 240 120 120 120 120 120 120 116 112 112
    116 120 120;
   23 07781540 07730791 1319 15641210 1969
      08491533 08761501 08971469 09231444 09461412 09751380 10111355 10371324
      10691290 10861262 11971200 12121169 12371134 12641104 12951081 13221053
      13301028 13570995 13750961 14050934 14310910 14590872 14680842
   0958133015020970 1469 1919
   0;
   coline
   1 18 -3360 12 380;
   480 120 120 120 120 120 120 120 120 120 120 240 120;
   18 09561548 09550789 1508 15611208 2008
      10221423 10551391 10831361 11061335 11281302 11521272 11761244 12001212
      12271185 12561150 12871120 13481065 13661036 13921005 14140972 14390947
      14670926 14880886
   1017132815031039 1558 1958
   2 16 -56 3 13;
   0 120 120 120 120 120 120 120 120 120 120 1080 120;
   16 03341586 03310809    8 19701238 1358
      03781566 04511534 05271505 06081481 06691444 07221413 07781385 08351351
      08921322 09411293 10051268 18160990 18410966 18770941 19050901 19260874
   0515145018490889  158 1258
   3 13 2874 12 160;
   0 120 120 120 1440 120;
   13 03821547 04040784  112 19071250 1362
      18661583 18371552 18151520 17921493 08921105 08241072 07791058 07261022
      06580981 06120960 05440921 04750891 04200854
   0461093717791417  162 1262
   0;
```

The deck is then read into a file named digit.in. The process is illustrated beginning with table 9 which is a printout of file digit.in containing data for three data files named club.org, mplace.org and coline.org. Line 1 is a header specifying the data file name. Lines 2 and 3 contain shotpoint and distance values corresponding to the parameters in data strings 1 and 2 of subroutine DATAIN Entry 1 (Appendix A). Line 4 contains the number of t-x data points, 15 in this case, (repeated from line 2) followed in turn by the digitized values corresponding to points A and B (fig. 40), the keypunched time for these points (23), the digitized coordinates of point C and the keypunched time for this point (1223). Lines 5 and 6 contain the digitized coordinates of the 15 arrival-time picks. Because a maximum of 24 data points are allowed for any given data set in filnam.org, digitized arrival-times can occupy at most 3 lines. Line 7 contains the digitized coordinates of points D and E (fig. 40) followed by their keypunched times (123 and 1173). These cards then represent all the data for the shotpoint with index k=1. The sequence is shown repeated for shot indices 2 through 4 for file club.org. The 0; in line 29 signifies the end of a file and will request output to disk, analogous to that used for terminal input. This data set for file club.org is followed by similar data sets for files mplace.org and coline.org.

Execution of a program named DIGIT converts the values of the digitized points in file digit.in into arrival-times and writes them in the file digit.out in a format compatible to that required for subroutine DATAIN. In addition, it tests the accuracy of checkpoints D and E and prints the results.

Execution of program DIGIT for the data in table 9 is shown in table 10. Execution consists of one step, namely invoking the command digit. The comparison between the manually read times for points D and E and the corresponding digitized values are printed out for each shot index of the three file names for comparison. Differences are shown under the heading "error".

A printout of file digit.out is shown in table 11. The format is identical to that read for terminal input (steps 6 and 7, table 3) except for the inclusion of a header for each separate data file.

The execution of subroutine DATAIN using the data in file digit.out is illustrated in table 12 as six steps.

1. Execute SEISMC and declare file name club.
2. Execute subroutine DATAIN, declare extension org, and that digitized data will be read. The calculated distance coordinate xlast, of the last data point of each data set is printed out for checking.
3. Execute subroutine INPUT to input data from files named mplace.
4. Repeat step 2 except for file name mplace.org.
5. Repeat step 3 except for file name coline.
6. Repeat step 2 except for file name coline.org.

Table 13 is a printout of the data in the newly created file coline.org. Table 14 shows the execution of subroutine PERP on the data in file coline.org and subsequent printout after execution.

Table 10.--Execution of program DIGIT for digitized data in Table 9.....

```
1 digit
 header:  club

index readtime  digitime   error    readtime  digitime  error
  1      123.     123.       0.4      1173.     1172.      0.5
  2      273.     272.       1.3      1223.     1224.     -1.2
  3     1265.    1267.      -1.8      1665.     1664.      1.3
  4      168.     169.      -0.8      1068.     1066.      1.8
 header:  mplace

index readtime  digitime   error    readtime  digitime  error
  1      960.     961.      -1.2      1510.     1510.     -0.4
  2      158.     160.      -1.5      1308.     1309.     -1.0
  3     1469.    1469.       0.4      1919.     1919.     -0.2
 header:  coline

index readtime  digitime   error    readtime  digitime  error
  1     1558.    1559.      -0.6      1958.     1960.     -2.3
  2      158.     158.       0.3      1258.     1259.     -1.3
  3      162.     163.      -0.6      1262.     1260.      1.5
 ----normal end of file.----

STOP
```

Table 11.--Printout of file digit.out after execution of program DIGIT.....

```
print digit.out

                    digit.out

club
1 15 -51 7 0;
0 120 120 120 120 120 120 120 120 120 1080 240 120;
       39,      102,      164,      231,      286,      334,      379,      427,
      474,      518,     1086,     1132,     1159,     1172,     1189,;
2 17 3015 15 0;
0 120 120 120 120 120 120 120 120 960 120;
     1225,     1205,     1182,     1160,     1139,     1115,     1094,     1080,
     1048,      499,      439,      392,      347,      300,      260,      211,
      146,;
3 23 -51 9 0;
2880 120 120 120 120 240 120;
     1218,     1235,     1251,     1280,     1299,     1348,     1394,     1412,
     1422,     1439,     1458,     1474,     1496,     1518,     1539,     1549,
     1574,     1594,     1616,     1628,     1646,     1666,     1684,;
4 17 3015 9 21;
2880 120 120 120 120 120 120 120 120 120 120 120 120 960 120;
       86,       23,       67,      130,      194,      255,      302,      346,
      386,      438,      480,      533,      584,     1090,     1113,     1135,
     1153,;
0;
mplace
1 20 -3000 7 0;
0 120 120 120 120 120 120 120 120 120 120 118 118 114 585 110 110 110 120 120;
      961,      984,     1007,     1032,     1060,     1083,     1107,     1123,
     1147,     1173,     1195,     1213,     1237,     1267,     1430,     1441,
     1463,     1480,     1497,     1519,;
2 10 -175 12 68;
0 120 120 120 120 120 240 1515 110 120;
      125,      190,      248,      290,      330,      374,      471,     1275,
     1301,     1327,;
3 23 -175 13 0;
2792 120 120 120 120 120 120 120 120 120 240 120 120 120 120 120 120 116 112 112
 116 120 120;
     1378,     1400,     1418,     1439,     1458,     1482,     1512,     1534,
     1560,     1575,     1666,     1679,     1700,     1722,     1748,     1770,
     1777,     1799,     1815,     1839,     1861,     1884,     1892,;
0;
coline
1 18 -3360 12 380;
480 120 120 120 120 120 120 120 120 120 120 240 120;
     1563,     1590,     1613,     1632,     1650,     1670,     1690,     1710,
     1732,     1756,     1782,     1832,     1847,     1869,     1887,     1908,
     1931,     1948,;
2 16 -56 3 13;
0 120 120 120 120 120 120 120 120 120 120 1080 120;
       44,      105,      167,      234,      285,      328,      375,      422,
      469,      509,      562,     1232,     1253,     1282,     1305,     1323;
3 13 2874 12 160;
0 120 120 120 1440 120;
     1336,     1311,     1293,     1273,      522,      465,      428,      383,
      326,      288,      231,      173,      127,;
0;
```

Table 12.--Execution of subroutine DATAIN for the data in file digit.out of Table 11.....

```
1. seismo

   type input file name: club
   new file--club.org created.
   new file--club.com created.
   new file--club.lay created.

   input complete from 3 dsk files named     club
2. type subroutine: datain
   type file extension either org,com,or lay: org
   enter  the word digit   if data digitized.
   anything else if not: digit
   club
   for shot index=    1 xlast=     2760.00
   for shot index=    2 xlsst=     2760.00
   for shot index=    3 xlast=     5640.00
   for shot index=    4 xlast=     5640.00
   output of 3 files to dsk complete
   datain complete
3. type subroutine: input

   type input file name: mplace
   new file--mplace.org created.
   new file--mplace.com created.
   new file--mplace.lay created.

   input complete from 3 dsk files named     mplace
4. type subroutine: datain
   type file extension either org,com,or lay: org
   enter  the word digit   if data digitized.
   anything else if not: digit
   mplace
   for shot index=    1 xlast=     2705.00
   for shot index=    2 xlast=     2585.00
   for shot index=    3 xlast=     5528.00
   output of 3 files to dsk complete
   datain complete
5. type subroutine: input

   type input file name: coline
   new file--coline.org created.
   new file--coline.com created.
   new file--coline.lay created.
   input complete from 3 dsk files named     coline
6. type subroutine: datain
   type file extension either org,com,or lay: org
   enter  the word digit   if data digitized.
   anything else if not: digit
   coline
   for shot index=    1 xlast=     2640.00
   for shot index=    2 xlast=     2760.00
   for shot index=    3 xlast=     2760.00
   output of 3 files to dsk complete
   datain complete
   type subroutine: QUIT
```

Table 13.--Printout of data in file coline.org after execution of subroutine DATAIN.....

```
print coline.org
                        coline.org

  1 18    -3360.00          12.00        380.00
     480.00      1563.00
     120.00      1590.00
     120.00      1613.00
     120.00      1632.00
     120.00      1650.00
     120.00      1670.00
     120.00      1690.00
     120.00      1710.00
     120.00      1732.00
     120.00      1756.00
     120.00      1782.00
     240.00      1832.00
     120.00      1847.00
       0.00      1869.00
       0.00      1887.00
       0.00      1908.00
       0.00      1931.00
       0.00      1948.00
  2 16      -56.00           3.00         13.00
       0.00        44.00
     120.00       105.00
     120.00       167.00
     120.00       234.00
     120.00       285.00
     120.00       328.00
     120.00       375.00
     120.00       422.00
     120.00       469.00
     120.00       509.00
     120.00       562.00
    1080.00      1232.00
     120.00      1253.00
       0.00      1282.00
       0.00      1305.00
       0.00      1323.00
  3 13     2874.00          12.00        160.00
       0.00      1336.00
     120.00      1311.00
     120.00      1293.00
     120.00      1273.00
    1440.00       522.00
     120.00       465.00
       0.00       428.00
       0.00       383.00
       0.00       326.00
       0.00       288.00
       0.00       231.00
       0.00       173.00
       0.00       127.00
```

Table 14.--Execution of subroutine PERP for data in file coline.org and printout of this file after execution.....

```
seismo

type input file name: coline

input complete from 3 dsk files named     coline
type subroutine: perp
type extension--either org or com: org
output of 3 files to dsk complete
type subroutine: term
type input filename and extension: coline.org

                    coline.org

  1 18   -3360.00       12.00       -1.00
     498.76     1563.00
     618.19     1590.00
     737.66     1613.00
     857.16     1632.00
     976.68     1650.00
    1096.23     1670.00
    1215.81     1690.00
    1335.40     1710.00
    1455.02     1732.00
    1574.65     1756.00
    1694.31     1782.00
    1933.66     1832.00
    2053.35     1847.00
    2173.06     1869.00
    2292.79     1887.00
    2412.52     1908.00
    2532.27     1931.00
    2652.02     1948.00
  2 16     -56.00        3.00       -1.00
       1.49       44.00
     120.48      105.00
     240.29      167.00
     360.20      234.00
     480.16      285.00
     600.13      328.00
     720.11      375.00
     840.09      422.00
     960.08      469.00
    1080.07      509.00
    1200.07      562.00
    2280.04     1232.00
    2400.03     1253.00
    2520.03     1282.00
    2640.03     1305.00
    2760.03     1323.00
  3 13    2874.00       12.00       -1.00
      -4.45     1336.00
     115.36     1311.00
     235.14     1293.00
     354.91     1273.00
    1788.15      522.00
    1906.68      465.00
    2024.79      428.00
    2142.29      383.00
    2258.83      326.00
    2373.72      288.00
    2485.52      231.00
    2590.53      173.00
    2677.54      127.00

type subroutine: QUIT
```

Subroutine FUDGE

Subroutine FUDGE can be used to add a constant to any of the doubly subscripted arrays in the data files. These are the arrival-times GT(K,J) and GTC(K,J), distances GX(K,J) and GXC(K,J) and the model parameters CX(I,J), CY(I,J) and CV(I,J). Of these, changes are made most frequently to arrival-times due to timing errors. Changes to any of the other variables are seldom required and are usually made by editing files.

Appendix A contains the entry list for subroutine FUDGE. In the example given (table 15) changes are made to file test.org (table 15a) for which the arrival-times for shot index 1 are to be increased by 10 for all 13 data points, and the distances for shot index 2 by 20. The procedure is illustrated by table 15b. A printout of the corrected file is obtained (table 15c) by executing subroutine TERM.

Subroutine TERM

Subroutine TERM permits printing of any file on the terminal without first exiting from SEISMC. An example is shown in table 15c.

Subroutine BUILD

Subroutine BUILD is used to combine arrival-time data from more than one shotpoint in such a way that the combined data represents arrival-times from a single shotpoint. This is illustrated in figure 41. The index k=1, represents data for spread III recorded from shotpoint 1 (S_1), the index k=2 for spread II from shotpoint S_1, k=3 for spread II from S_2, k=4 for spread I from S_2 and k=5 for spread I from S_3. The graphs suggest that these data sets represent arrivals from more than one horizon, but that arrivals in the intervals j to l, e to h, and a to c probably all represent arrivals from the same horizon. The data in these intervals can then be combined to represent data from this horizon recorded by the shotpoint at S_1, as shown in the lower portion of figure 41. Combining these data may be done in several different ways. For example, the data for k=5 may be lowered an amount add1 (fig. 41) and combined with those from k=3 as though the shotpoint were at S_2. These combined data may then be lowered an amount add2 and combined with those of k=1 as though the shotpoint were at S_1. If the same is done for the data from shots 4, 5, and 6 (individual data sets not shown) and combined as the dots in the lower portion of the figure, one has, in effect, reversed t-x data sets from shotpoints S_1 and S_4.

In combining two t-x data sets, the shotpoint distance coordinate and the shot depth of one of them, called the control set, are retained to represent that of the combined data. The set which is combined with the control data and which usually has a constant added to the arrival-times, such as add1 and add2 in figure 41, is called the subsidiary set. The newly combined data set is automatically stored in filnam.com. The data sets from which this new set is derived may have been stored in either filnam.org or filnam.com. Furthermore, a combined data set may replace one of those from which it was derived, or some other existing data set no longer needed.

```
print test.org

                    test.org

(a)  1 13      -1.00        5.00       -1.00
        3.12         4.00
        6.21         6.00
       10.70        12.00
       15.49        15.00
       20.38        16.00
       25.31        18.00
       30.26        20.00
       35.22        20.00
       40.19        20.00
       45.17        20.00
       50.16        20.00
       55.14        22.00
       60.13        24.00
     2 11       63.00        2.00       -1.00
        0.00        26.00
        5.00        25.00
       10.00        24.00
       15.00        22.00
       25.00        20.00
       30.00        17.00
       35.00        15.00
       45.00        13.00
       50.00        12.00
       55.00         9.00
       60.00         4.00

(b)  seismo

     type input file name: test

     input complete from 3 dsk files named     test
     type subroutine: fudge
     fudge entry 1; const*,(either gt,gx,cx,cy,cv,gtc,gxc),k,is,ie
     *10 gt 1 1 13;
     variable gt   shot or layer no.    1  indeces   1 to  13
     changed with const =       10.0
     fudge entry 1; const*,(either gt,gx,cx,cy,cv,gtc,gxc),k,is,ie
     *20 gx 2 1 11;
     variable gx   shot or layer no.    2  indeces   1 to  11
     changed with const =       20.0
     fudge entry 1; const*,(either gt,gx,cx,cy,cv,gtc,gxc),k,is,ie
     *0;
     output of 3 files to dsk complete
     type subroutine: term
(c)  type input filename and extension: test.org

                    test.org

     1 13      -1.00        5.00       -1.00
        3.12        14.00
        6.21        16.00
       10.70        22.00
       15.49        25.00
       20.38        26.00
       25.31        28.00
       30.26        30.00
       35.22        30.00
       40.19        30.00
       45.17        30.00
       50.16        30.00
       55.14        32.00
       60.13        34.00
     2 11       63.00        2.00       -1.00
       20.00        26.00
       25.00        25.00
       30.00        24.00
       35.00        22.00
       45.00        20.00
       50.00        17.00
       55.00        15.00
       65.00        13.00
       70.00        12.00
       75.00         9.00
       80.00         4.00

     type subroutine: QUIT
```

Table 15.--(a) Printout of file test.org before corrections by subroutine FUDGE
(b) Execution of subroutine FUDGE
(c) Printout of file test.org after corrections.....

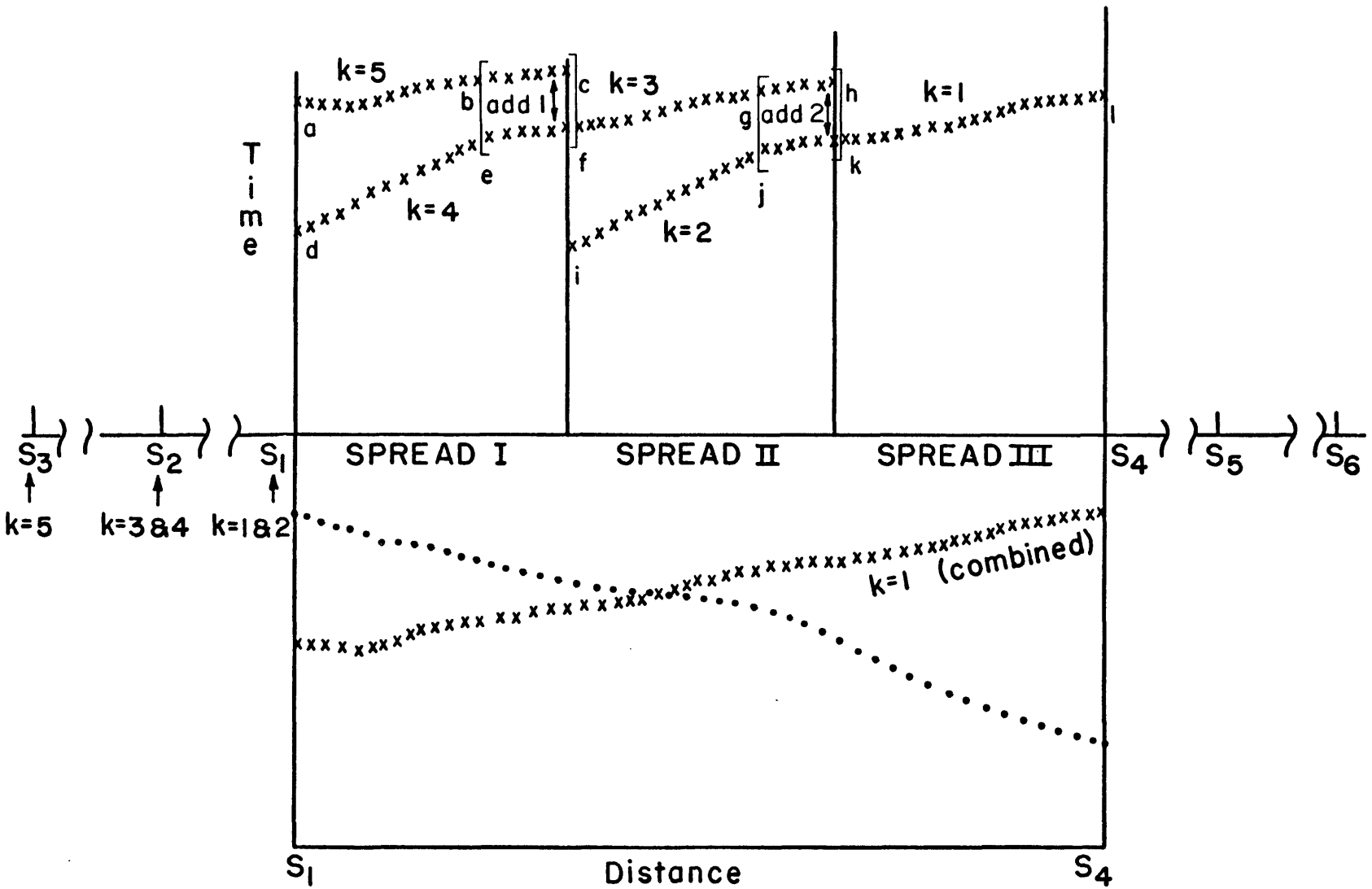

Figure 41.--Sketch illustrating the manner by which arrival-time data from three shotpoint locations S_1, S_2 and S_3 are combined to represent arrival-times from a single shotpoint S_1.....

If the portions of two data sets to be combined overlap, such as in combining the data between points e and f (fig. 41) for k=4 with those between a and c for k=5, an option may be selected which automatically calculates the correction to be applied to the subsidiary set. On the other hand, an arbitrary correction value may be chosen by the user.

As an example, of execution of subroutine BUILD, figure 42 shows four sets of t-x data points (k=1 through 4) which have been input into file play.org (table 16a). Sets with indices k=1 (the control set) and k=2 (the subsidiary set) are to be combined and entered into file play.com with shot index k=5 using points 3 through 7 of k=2 and all the points, 1 through 7, of k=1. Because the sets do not overlap, the values chosen (option 2) to be added to the arrival-times of the subsidiary set is -9.2. The entry list is in Appendix A and table 17a tabulates the input to combine these sets.

Similarly, data sets with indices k=3 (control set) and k=4 (subsidiary set) are to be combined and entered into file play.com with shot index k=7. Points 1 through 8 of shot index k=3 and points 9 through 13 of k=4 are to be combined. Because of data overlap, the points within the brackets will be used (option 1) to determine a value to be added to the subsidiary data. Table 17b tabulates the input to combine these sets.

Execution of subroutine BUILD is illustrated in table 16b and the printout of file play.com after execution is shown in table 16c. The t-x plots for the combined sets are also shown by the triangles in figure 42.

In calculating the value to be added to the subsidiary set under option 1, least square straight lines are calculated through both sets of points within the interval of comparison (brackets in figure 42b) and the average difference between them determined. This difference is then applied to the arrival-times of the subsidiary set.

The program conditions for subroutine BUILD which may arise during execution are listed in Appendix A.

Subroutine PICK

Subroutine PICK is similar to subroutine BUILD and could well have been written as a branch of that program. It permits the user to select a subset of a single t-x data set from either filenam.org or filenam.com, add a constant to the arrival-times and store that result with a separate shot index in either of the files. The shotpoint parameters are the same as those of the initial data set.

An example is given in table 18. The entry list and program conditions are in Appendix A. Table 18a shows the data file horse.com from which a subset is to be extracted from shot index k=2, 10 added to the arrival-times and the result entered into file horse.org with shot index k=17. Execution is shown as two stops in table 18b and the result is printed in table 18c.

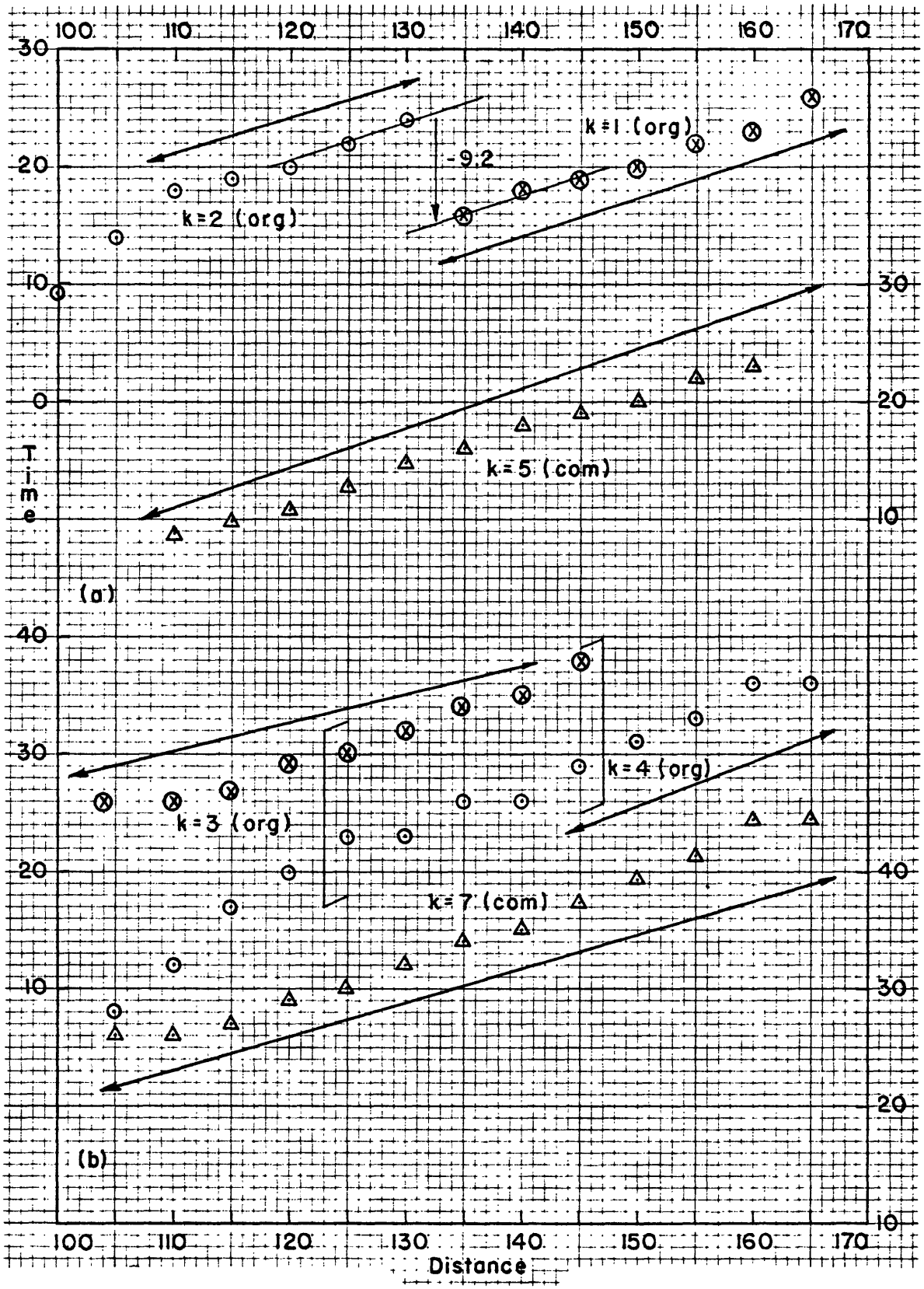

Figure 42.--T-$\bar{x}$ data sets for example illustrating execution of subroutine BUILD as described in text.....

(a)

```
seismc

type input file name: play

input complete from 3 dsk files named     play
type subroutine: term
type input filename and extension: play.org

                    play.org


 1  7     100.00        5.00       -1.00
    135.00       16.00
    140.00       18.00
    145.00       19.00
    150.00       20.00
    155.00       22.00
    160.00       23.00
    165.00       26.00
 2  7      80.00        7.00       -1.00
    100.00        9.00
    105.00       14.00
    110.00       18.00
    115.00       19.00
    120.00       20.00
    125.00       22.00
    130.00       24.00
 3  9       0.00        3.00       -1.00
    105.00       26.00
    110.00       26.00
    115.00       27.00
    120.00       29.00
    125.00       30.00
    130.00       32.00
    135.00       34.00
    140.00       35.00
    145.00       38.00
 4 13      90.00        4.00       -1.00
    105.00        8.00
    110.00       12.00
    115.00       17.00
    120.00       20.00
    125.00       23.00
    130.00       23.00
    135.00       26.00
    140.00       26.00
    145.00       29.00
    150.00       31.00
    155.00       33.00
    160.00       36.00
    165.00       36.00
```

(b)

```
type subroutine: build
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2, opt
if opt=1; xsmal,xlarg or if opt=2; add
*5 1 org 131 170 2 org 109 132 2 -9.2;
combined shot index    5 has    12 jugs
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2, opt
if opt=1; xsmal,xlarg or if opt=2; add
*7 3 org 100 142 4 org 142 170 1 121 146;
constant    8.40 is added to shot    4 org
 to combine it with shot    3 org
combined shot index    7 has   13 jugs
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2, opt
if opt=1; xsmal,xlarg or if opt=2; add
*0;
output of 3 files to dsk complete
type subroutine: term
type input filename and extension: play.com
```

(c)

```
                    play.com


 5 12     100.00        5.00       -1.00
    110.00        8.80
    115.00        9.80
    120.00       10.80
    125.00       12.80
    130.00       14.80
    135.00       16.00
    140.00       18.00
    145.00       19.00
    150.00       20.00
    155.00       22.00
    160.00       23.00
    165.00       26.00
 7 13        0.00        3.00       -1.00
    105.00       26.00
    110.00       26.00
    115.00       27.00
    120.00       29.00
    125.00       30.00
    130.00       32.00
    135.00       34.00
    140.00       35.00
    145.00       37.40
    150.00       39.40
    155.00       41.40
    160.00       44.40
    165.00       44.40


type subroutine: QUIT
```

Table 16.--(a) Printout of file play.org before combining data sets by subroutine BUILD
(b) Execution of subroutine BUILD
(c) Printout of file play.com after combining data sets.....

BUILD input form

File name play

	k	control set k1	ext1 org/com	xmin1	xmax1	subsidiary set k2	ext2 org/com	smin2	xmax2	opt(1) opt(2)	xsmal add	xlarg
(a)	5	1	org	131	170	2	org	109	132	2	-9.2	
(b)	7	3	org	100	142	4	org	142	170	1	121	146

Table 17.--Input form with input values for execution of subroutine BUILD.....

Table 18.--(a) Printout of file horse.com before execution
of subroutine PICK
(b) Execution of subroutine PICK
(c) Printout of file horse.org after execution
of subroutine PICK.....

(a)

```
seismc

type input file name: horse

input complete from 3 dsk files named     horse
type subroutine: term
type input filename and extension: horse.com

                    horse.com


 2 13       0.00        3.00       -1.00
    105.00       26.00
    110.00       26.00
    115.00       27.00
    120.00       29.00
    125.00       30.00
    130.00       32.00
    135.00       34.00
    140.00       35.00
    145.00       37.40
    150.00       39.40
    155.00       41.40
    160.00       44.40
    165.00       44.40
```

(b) 1.

```
type subroutine: pick
pick entry 1; shot1*,ext1,xmin,xmax,shot2,ext2,add
*2 com 117 145 17 org 10;
new data set: extension=org  index=17   njugs=  6
```

2.

```
pick entry 1; shot1*,ext1,xmin,xmax,shot2,ext2,add
*0;
output of 3 files to dsk complete
type subroutine: term
type input filename and extension: horse.org
```

(c)

```
                    horse.org


 17  6       0.00        3.00       -1.00
    120.00       39.00
    125.00       40.00
    130.00       42.00
    135.00       44.00
    140.00       45.00
    145.00       47.40


type subroutine: QUIT
```

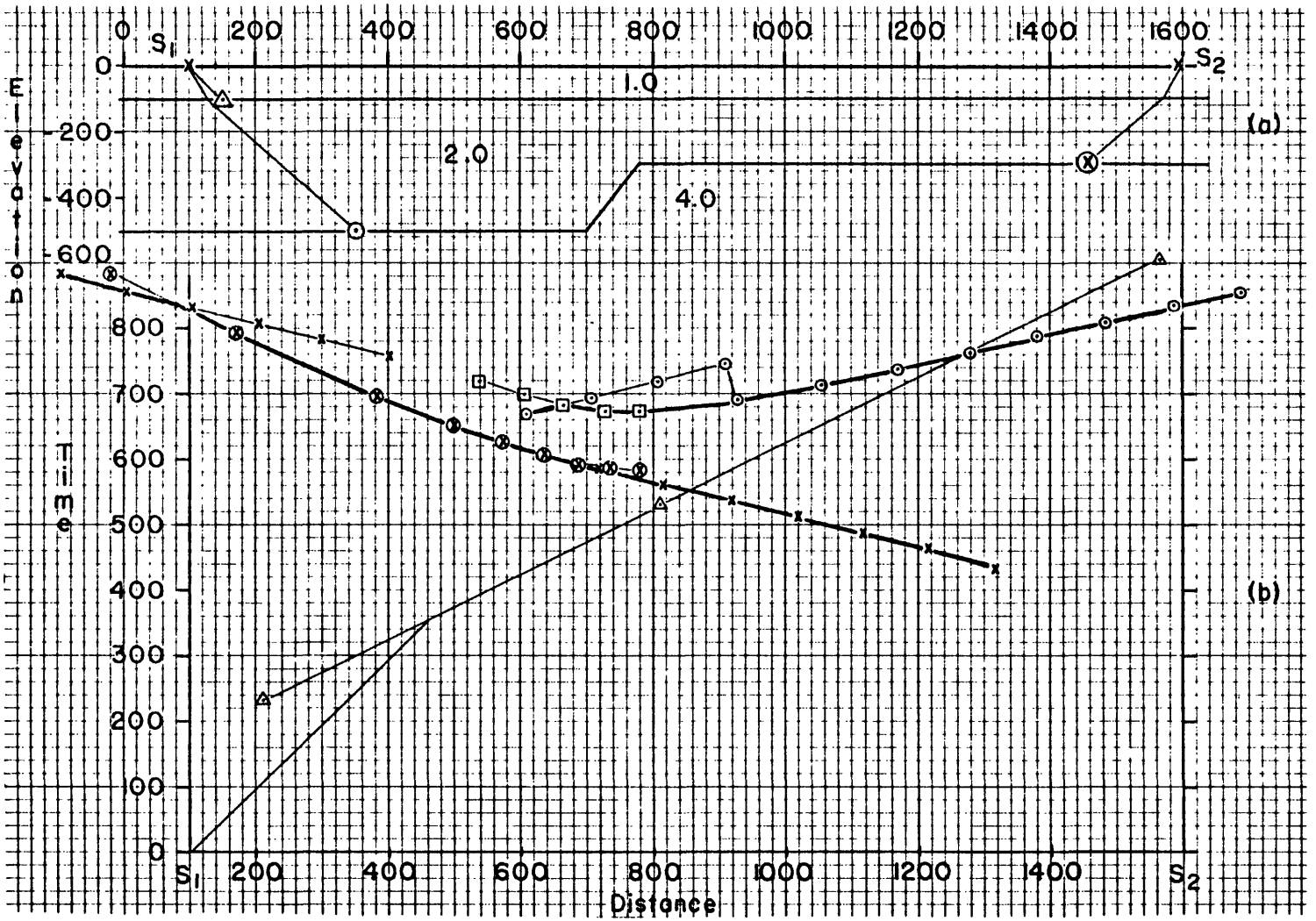

Figure 43.--First example used to illustrate execution of subroutine RAYTRACE.....

(a)

```
print raytr1.lay

                        raytr1.lay

  1  2  -90000.00        0.00        1.00
   90000.00        0.00        1.00
  2  2  -90000.00     -100.00        2.00
   90000.00     -100.00        2.00
  3  4  -90000.00     -500.00        4.00
      700.00     -500.00        4.00
      780.00     -300.00        4.00
    90000.00     -300.00        4.00
```

(b)

```
seismc

type input file name: raytr1

1. input complete from 3 dsk files named     raytr1
   type subroutine: raytr
2. raytrace entry 1                          file=    raytr1
   layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
   *2 100 0 100 1600 150;
3. raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
   *0;
4. critical angle exceeded in subroutine angle
   normal exit for raydown
   the near-critical subsurface paramaters beneath the shot-point are
     horizon       x-coord.     elevation       time      refrac. angle
        2         157.73       -100.00        115.47        -89.94

5. do you want to continue? yes or no:yes
   arrival time data for shot at x=     100.00 depth=    0.00
     from off horizon    2
    x-refractor coord.    x-surface coord.   arrival time
            157.74              215.46        230.94
            307.74              365.47        305.94
            457.74              515.47        380.94
            607.74              665.47        455.94
            757.74              815.47        530.94
            907.74              965.47        605.94
           1057.74             1115.47        680.94
           1207.74             1265.47        755.94
           1357.74             1415.47        830.94
           1507.74             1565.47        905.94
6. all done. another shot!
   raytrace entry 1                          file=    raytr1
   layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
   *3 100 0 100 1600 100;
7. raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
   *1 780 - ;
8. critical angle exceeded in subroutine angle
   normal exit for raydown
   the near-critical subsurface paramaters beneath the shot-point are
     horizon       x-coord.     elevation       time      refrac. angle
        3         355.01       -500.00        333.78        -84.13

9. do you want to continue? yes or no:yes
   arrival time data for shot at x=     100.00 depth=    0.00
     from off horizon    3
    x-refractor coord.    x-surface coord.   arrival time
            355.01              611.75        668.00
            455.01              711.75        693.00
            555.01              811.75        718.00
            655.01              911.75        743.00
            755.01              515.95        729.21
            855.01              934.44        689.47
            955.01             1060.07        711.70
           1055.01             1173.50        735.32
           1155.01             1280.91        759.36
           1255.01             1385.29        783.60
           1355.01             1488.06        807.98
           1455.01             1589.90        832.47
           1555.01             1691.19        857.04
10. diffraction results from above shot:
    diffractions from subsurface point at x=     780.00
     emergence angle    x-surface coord.   arrival time
              0.00              780.00        673.88
             -5.00              758.14        674.36
            -10.00              736.02        675.80
            -15.00              713.36        678.26
            -20.00              689.85        681.80
            -25.00              665.12        686.53
            -30.00              638.71        692.63
            -35.00              610.02        700.34
            -40.00              578.24        710.03
            -45.00              542.20        722.21
            -50.00              500.19        737.71
            -55.00              449.47        757.84
            -60.00              385.55        784.82
            -65.00              300.27        822.68
            -70.00              177.28        879.54
            -75.00              -21.56        974.45
            -80.00             -410.82       1164.65
11. all done. another shot!
    raytrace entry 1                          file=    raytr1
    layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
    *3 1600 0 1600 0 100;
    raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
    *1 780 + ;
    critical angle exceeded in subroutine angle
    normal exit for raydown
    the near-critical subsurface paramaters beneath the shot-point are
      horizon       x-coord.     elevation       time      refrac. angle
         3        1459.66       -300.00        218.51         84.13
```

Table 19.--(a) Printout of velocity model in file raytr1.lay of Figure 43a.
(b) Execution of subroutine RAYTRACE for velocity model shown in Figure 43a.....

Table 19.--Continued

```
12. do you want to continue? yes or no:yes
    arrival time data for shot at x=   1600.00 depth=   0.00
      from off horizon   3
     x-refractor coord.   x-surface coord.  arrival time
            1459.66            1318.38       437.26
            1359.66            1218.38       462.26
            1259.66            1118.38       487.26
            1159.66            1018.38       512.26
            1059.66             918.38       537.26
             959.66             818.38       562.26
             859.66             718.38       587.26
             759.66            -808.48      1274.99
             659.66             402.91       759.20
             559.66             302.91       784.20
             459.66             202.91       809.20
             359.66             102.91       834.20
             259.66               2.91       859.20
             159.66             -97.09       884.20
              59.66            -197.09       909.20
13. diffraction results from above shot:
    diffractions from subsurface point at x=    780.00
      emergence angle   x-surface coord.  arrival time
                0.00            780.00       588.43
               -5.00            758.14       588.90
              -10.00            736.02       590.35
              -15.00            713.36       592.80
              -20.00            689.85       596.34
              -25.00            665.12       601.08
              -30.00            638.71       607.18
              -35.00            610.02       614.89
              -40.00            578.24       624.57
              -45.00            542.20       636.75
              -50.00            500.19       652.26
              -55.00            449.47       672.39
              -60.00            385.55       699.37
              -65.00            300.27       737.23
              -70.00            177.28       794.09
              -75.00            -21.56       889.00
              -80.00           -410.82      1079.19
    all done. another shot!
    raytrace entry 1                    file=    raytr1
    layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
    *QUIT
```

Subroutine PLOTX

Subroutine PLOTX is used to write t-x data sets from either filnam.org or filnam.com onto a 9-track tape for later plotting. Because plotter routines are system oriented, details regarding the programs are not presented here. The entry list, however, is given in Appendix A.

Subroutine RAYTRACE

Given a velocity model in filnam.lay and a shotpoint distance coordinate and depth, subroutine RAYTRACE calculates the t-x coordinates for an arrival-time plot of critically refracted or diffracted rays from any horizon other than the first (the surface horizon). Execution of RAYTRACE thus represents a reverse process of calculating a velocity model from arrival-time data and hence is helpful in checking the results of such calculations.

If the shot is deeper than the horizon to be calculated, but shallower than the next underlying one, the arrival-times at all but close distances are approximately equal to those obtained if the shot were placed on the horizon being calculated. Hence, a provision is included for calculating arrival-times with the shot placed on the upper surface of the layer being calculated.

During execution, RAYTRACE calls subroutines COMP, RAYD, and RAYUP.

Three examples of the execution of RAYTRACE follow. The entry list and condition statements are in Appendix A.

Figure 43 shows a 3-layer problem, with horizon 3 offset by a fault near the center of the profile. The velocities of each layer are indicated. Table 19a is a printout of this velocity model, contained in file raytr1.lay.

The shotpoints for which t-x values are to be calculated are shown in figure 43a as S_1 and S_2 at distances of 100 and 1600 respectively. Table 19b shows printout during execution of RAYTRACE for this problem.

Thirteen steps are illustrated as follows:

1. Execute SEISMC, request file raytr1, and execute subroutine RAYTR.
2. Input data for entry 1 (see entry list). Entry indicates that t-x data are to be calculated from horizon 2. The shot distance coordinate is 100 and its depth is 0. Arrivals are to be calculated between subsurface distance coordinates 100 and 1600 and the calculation interval is 150. The default options are used for the remaining input variables.
3. Input data for entry 2. Because horizon 2 is flat, the entry of 0 commands that no diffraction will be calculated.
4. Printout shows the results for a ray which propogated to horizon 2 and intersected it at approximately the critical angle. The first line is a message from subroutine ANGLE indicating that a trial ray impinged upon some horizon beyond the critical angle. The message, "normal exit for raydown", indicates that reflection beyond the

critical angle occurred on horizon layd (2) and that it did not occur on the first trial ray used. The next two lines show the distance and elevation coordinates (157.73,-100) on horizon 2 at which approximate critical refraction occurs, and the travel-time (115.47) from the shotpoint to this critical refraction point. The value -89.94 would actually be -90 if the exact critical refraction point had been found. (In practice, the absolute value of this variable generally falls between 80 and 90. Values closer to 90 can be achieved, if desired, by decreasing the value of dbeta in entry 1. However, actual arrival-times at the surface are quite insensitive to whether or not the critical point is accurately determined). At this point, the user has the option to stop further calculations by entering "no" to the indicated question.

5. This step shows the computed results. The second and third columns show the distance and corresponding time values, respectively, of refracted arrivals at points on the ground surface. The first column shows the x coordinates at which these arrivals emanate from horizon 2. Some of these t-x values have been plotted (triangles in figure 43b) and the line through these points shows the arrival-time curve of horizon 2 for the shotpoint on the surface at distance coordinate 100 as input in step 2.
6. Entry in this step commands that refracted arrivals be calculated for a shot at coordinate 100 from horizon 3 using the parameters indicated.
7. This step contains input data for entry 2. The entry commands that one diffraction point on horizon 3 be calculated at the distance coordinate 780. The minus (-) indicates that diffracted arrivals emanating from this diffraction point are to be calculated in the direction toward the shotpoint. Diffractions in the + direction were not calculated because they are all secondary arrivals.
8. Near-critical refraction occurred on horizon 3 at coordinates (355.01, -500) with travel-time 333.78. These values were calculated at a refraction angle of -84.13 degrees (instead of 90).
9. The refraction results are printed out as in Step 9. They are plotted as circles in figure 43b.
10. This step shows the diffraction results. The first column contains the diffraction angles and the next two columns show the corresponding computed t-x values. These are plotted with the square symbols in figure 43b.
11. This step is a repetition of steps 6, 7, and 8 for a shotpoint at distance coordinate 1600.
12. Refracted t-x results obtained from step 11 are plotted with the symbol x in figure 43b.
13. Diffraction t-x results are plotted with the circled symbol x.

The arrival-time curves for horizon 3 are determined, as shown in figure 43b, by selecting the minimum values of the refracted and diffracted arrivals. These minimum values have been connected by the heavy lines which then represent the arrival-time curves for horizon 3.

Note that arrival-times for diffracted and refracted ray paths at S_1 for the shotpoint at S_2 are practically the same. If the vertical offset on horizon 3 had been closer to S_1 or had its displacement been larger, the diffracted arrival-time at S_1 would have been less than the refracted. This would have posed a problem in calculating the travel-time curve for the shotpoint at S_1 because the minimum time paths would not pass through the the point of critical refraction at (355.01, -500) (see step 8).

Another example of RAYTRACE is shown in figure 44. The velocity model consists of two layers (shown in file raytr2.lay in table 20a) with shotpoint at distance coordinate 50. Horizon 2 dips steeply beneath the shotpoint. Execution of RAYTRACE (table 20b) results in condition 7 (see conditions statements in Appendix A). Condition 7 occurred because the default option of an initial vertical down-going ray, incremented in negative steps (e.g. toward the right) could never intersect horizon 2 at the critical angle. As seen in figure 44, the ray that strikes horizon 2 at the critical angle must propagate to the left from the shotpoint rather than to the right. This condition is overcome by using a positive non-zero (15^o) value for betal in entry 1 in keeping with the sign convention. The t-x results from table 20b are plotted in figure 44b. The plot shows arrival-times along two segments, one corresponding to the dipping part and the other to the horizontal part of horizon 2. Care must be exercized not to honor the point shown as A in figure 44b as a first-arrival from this horizon. Point A, as seen from the rays sketched in figure 44a, represents an arrival from the horizontal portion of horizon 2, whereas an arrival from the dipping portion occurs at the same distance at less time.

Figure 45a represents a rather complex velocity model of four layers, each varying laterally in velocity. This model is contained in file raytr3.lay (table 21a). Figures 45b and c are plots of the t-x values calculated by subroutine RAYTRACE for horizons 2, 3, and 4 with shotpoints at the distance coordinates of 0 and 2000. Execution and printout of the resulting t-x points is given in table 21b.

The last 3 examples were executed using the default option "no" for variable 6 in entry 1 of the entry list. Entry of "no" allows the ray paths to cross lateral velocity boundaries without bending (without being refracted). It has been our experience that this option gives better results. The reason for this is that, in nature, lateral velocity changes probably are better represented by smooth continuous functions rather than the sharp discontinuities represented by the velocity cells shown in figure 45.

Subroutines CRIT (xicrt,etacrt,taucrt,xcrt,tcrt) and
ACROSS (xicrt,etacrt,taucrt,xcrt,tcrt,id,xfar,x,t,nobs,curv,al,bl,cl,nval,
nstar,xstar,xprim,tstar,tprim,xir,etar,bend2)

Given a velocity model in filnam.lay consisting of id layers and one-way t-x data from horizon (id+1) in either filnam.org or filenam.com, subroutine CRIT calculates the parameters for reflection at the critical angle from horizon (id+1). These parameters are the subsurface point of critical reflection with coordinates (xicrt,etacrt), the travel-time from the shotpoint to the critical reflection point (taucrt), the distance

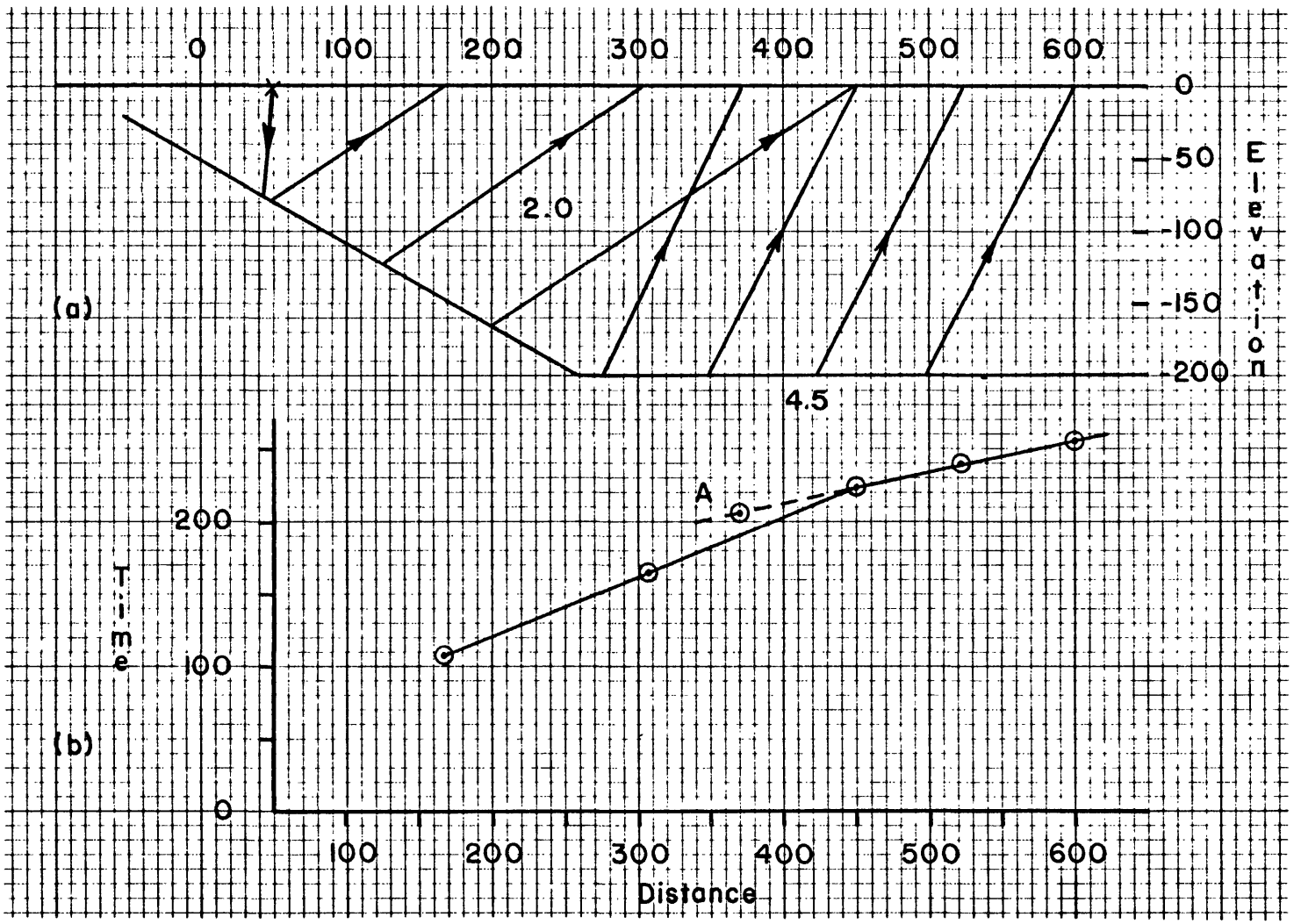

Figure 44.--Second example used to illustrate execution of subroutine RAYTRACE.....

Table 20.--(a) Printout of velocity model in file raytr1.lay of Figure 44a.
(b) Execution of subroutine RAYTRACE for the velocity model shown in Figure 44a.....

```
seismc
```

(a)

```
type input file name: raytr2

input complete from 3 dsk files named     raytr2
type subroutine: term
type input filename and extension: raytr2.lay

                    raytr2.lay


  1  2  -90000.00        0.00        2.00
   90000.00        0.00        2.00
  2  4  -90000.00      -20.00        4.50
     -50.00      -20.00        4.50
     260.00     -200.00        4.50
   90000.00     -200.00        4.50
```

(b)

```
type subroutine: raytr
raytrace entry 1                          file=    raytr2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*2 50 3 50 500 75;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
raytrace condition 7 in layer    2
type subroutine: raytr
raytrace entry 1                          file=    raytr2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*2 50 3 50 500 75 no 0 15;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation      time     refrac. angle
     2         45.20       -75.28         36.22       -86.73

do you want to continue? yes or no:yes
arrival time data for shot at x=      50.00 depth=   3.00
  from off horizon   2
 x-refractor coord.   x-surface coord.  arrival time
          50.00             168.07       108.22
         125.00             308.93       166.98
         200.00             449.80       225.73
         275.00             374.22       206.38
         350.00             449.22       223.04
         425.00             524.22       239.71
         500.00             599.22       256.38
all done. another shot!
raytrace entry 1                          file=    raytr2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*quit
```

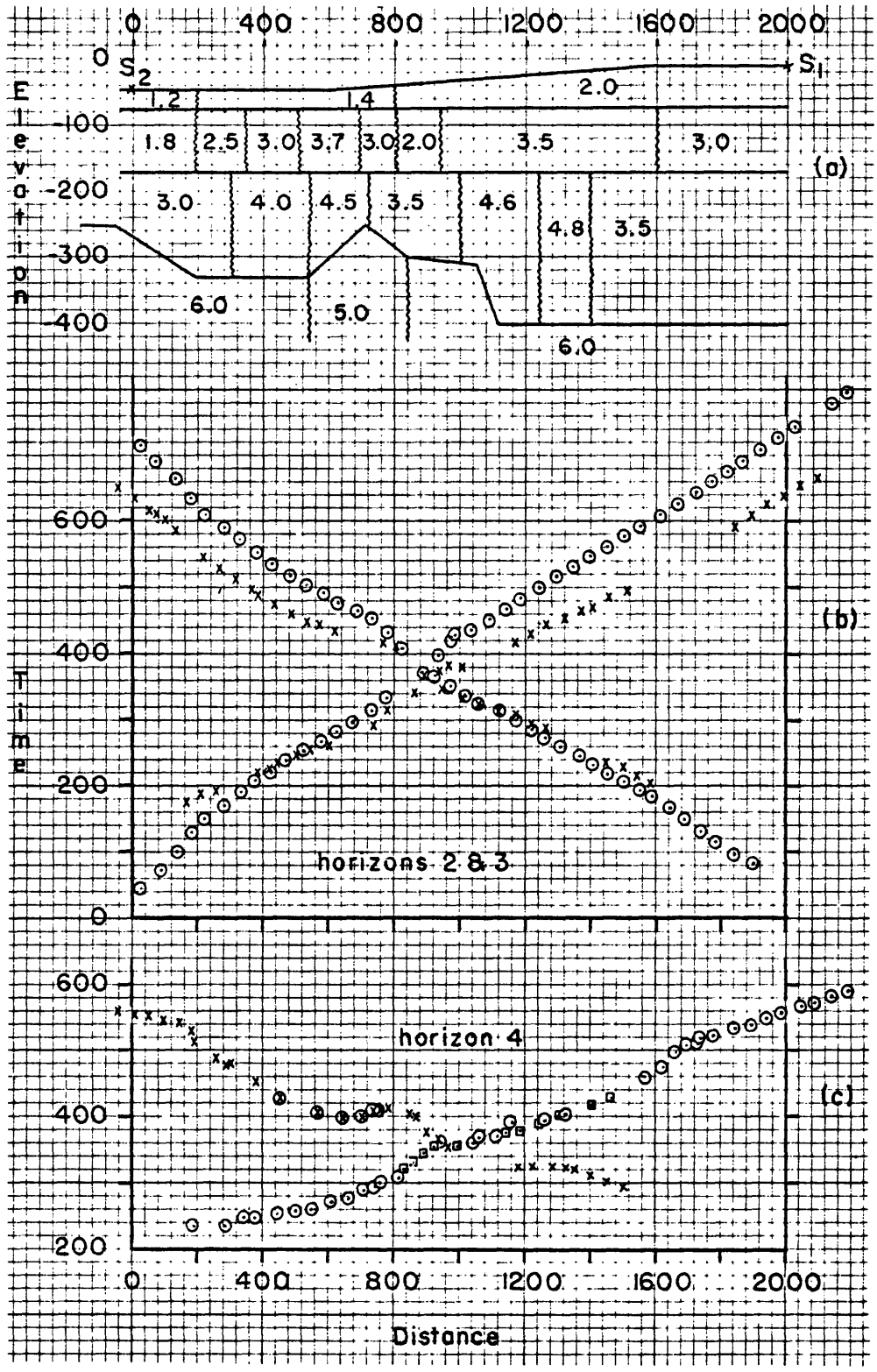

Figure 45.--Third example used to illustrate execution of subroutine RAYTRACE.....

Table 21.--(a) Printout of velocity model in file raytr3.lay of Figure 45a.
(b) Execution of subroutine RAYTRACE for horizon 2 of the velocity model shown in Figure 44a.
(c) Execution of subroutine RAYTRACE for horizon 3.
(d) Execution of subroutine RAYTRACE for horizon 4.....

(a)

```
seismo
type input file name: raytr3

input complete from 3 dsk files named     raytr3
type subroutine: term
type input filename and extension: raytr3.lay

                    raytr3.lay


 1  7  -90000.00      -50.00          1.20
    200.00      -50.00        1.40
    600.00      -50.00        1.40
    800.00      -41.00        1.80
   1400.00      -18.00        2.00
   1600.00      -10.00        2.00
  90000.00      -10.00        2.00
 2  9  -90000.00      -70.00          1.80
    200.00      -70.00        2.50
    340.00      -70.00        3.00
    500.00      -70.00        3.70
    700.00      -70.00        3.00
    800.00      -70.00        2.00
    940.00      -70.00        3.50
   1600.00      -70.00        3.00
  90000.00      -70.00        3.00
 3  8  -90000.00     -170.00          3.00
    300.00     -170.00        4.00
    540.00     -170.00        4.50
    720.00     -170.00        3.50
   1000.00     -170.00        4.60
   1240.00     -170.00        4.80
   1400.00     -170.00        3.50
  90000.00     -170.00        3.50
 4  9  -90000.00     -250.00          6.00
    -50.00     -250.00        6.00
    300.00     -330.00        6.00
    540.00     -330.00        5.00
    710.00     -250.00        5.00
    840.00     -300.00        6.00
   1060.00     -310.00        6.00
   1120.00     -400.00        6.00
  90000.00     -400.00        6.00
```

(b)

```
type subroutine: raytr
raytrace entry 1                          file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*2 0 0 0 2000 100;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
     2          17.88        -70.00         22.36        -88.85

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=    0.00
  from off horizon   2
 x-refractor coord.    x-surface coord.   arrival time
         17.88               35.77          44.72
        117.88              135.77         100.27
        217.88              231.40         147.93
        317.88              331.40         187.93
        417.88              428.43         221.65
        517.88              526.06         253.13
        617.88              626.55         281.08
        717.88              731.56         313.82
        817.88              884.42         370.27
        917.88              993.02         425.58
       1017.88             1040.81         437.27
       1117.88             1143.16         468.38
       1217.88             1245.51         499.50
       1317.88             1347.86         530.61
       1417.88             1455.64         559.83
       1517.88             1558.50         590.91
       1617.88             1671.54         625.03
       1717.88             1771.54         658.37
       1817.88             1871.54         691.70
       1917.88             1971.54         725.03
all done. another shot!
raytrace entry 1                          file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*2 2000 0 2000 0 100
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
     2        1946.35        -70.00         40.24         88.85

do you want to continue? yes or no:yes
arrival time data for shot at x=    2000.00 depth=    0.00
  from off horizon   2
 x-refractor coord.    x-surface coord.   arrival time
       1946.35             1892.69          80.49
       1846.35             1792.69         113.82
       1746.35             1692.69         147.16
       1646.35             1592.94         180.30
       1546.35             1507.16         205.31
       1446.35             1409.87         231.52
       1346.35             1317.08         259.79
       1246.35             1219.32         285.93
```

Table 21.--Continued

```
      1146.35          1121.57       312.08
      1046.35          1023.82       338.22
       946.35           926.07       364.37
       846.35           787.64       429.51
       746.35           732.65       453.12
       646.35           637.49       478.83
       546.35           538.18       504.56
       446.35           435.80       535.69
       346.35           335.80       569.02
       246.35           232.83       609.69
       146.35           128.46       663.15
        46.35            28.46       718.71
```

(c)

```
all done. another shot!
raytrace entry 1                        file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*3 0 0 0 2000 100;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0 ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation      time      refrac. angle
     3          83.33       -170.00        87.50       -85.47

do you want to continue? yes or no:yes
arrival time data for shot at x=      0.00 depth=   0.00
  from off horizon   3
 x-refractor coord.    x-surface coord.   arrival time
          83.33              167.06        175.13
         183.33              265.46        190.32
         283.33              442.50        234.50
         383.33              504.18        246.19
         483.33              602.66        262.80
         583.33              738.67        299.26
         683.33              862.30        340.40
         783.33              967.10        391.12
         883.33              963.80        382.96
         983.33            11031.33       3677.45
        1083.33             1219.74        429.08
        1183.33             1321.40        453.17
        1283.33             1411.08        472.46
        1383.33             1515.83        494.65
        1483.33            11531.33       3815.34
        1583.33            11631.33       3848.68
        1683.33             1891.48        607.44
        1783.33             1991.48        636.01
        1883.33             2091.48        664.58
        1983.33             2191.48        693.15
all done. another shot!
raytrace entry 1                        file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*3 2000 0 2000 0 100;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0 ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation      time      refrac. angle
     3        1792.67       -170.00       101.03       87.13

do you want to continue? yes or no:yes
arrival time data for shot at x=   2000.00 depth=   0.00
  from off horizon   3
 x-refractor coord.    x-surface coord.   arrival time
        1792.67             1584.93        201.92
        1692.67             1494.94        222.21
        1592.67            -8418.80       5412.28
        1492.67            -8518.80       5467.83
        1392.67             1267.15        284.60
        1292.67             1168.68        303.18
        1192.67             1058.88        324.37
        1092.67              960.48        343.83
         992.67            -9018.80       5716.84
         892.67              805.53        409.24
         792.67              619.14        433.00
         692.67              541.68        447.31
         592.67              439.92        473.94
         492.67              371.82        496.20
         392.67              270.04        527.09
         292.67              128.58        584.41
         192.67              108.95        602.13
          92.67                8.95        635.46
```

(d)

```
all done. another shot!
raytrace entry 1                        file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 0 0 0 2000 100 no 0 0 .25 0 10 ;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*3 710 - 710 + 1060 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation      time      refrac. angle
     4          52.04       -273.32       109.25       -87.30

do you want to continue? yes or no:yes
arrival time data for shot at x=      0.00 depth=   0.00
  from off horizon   4
 x-refractor coord.    x-surface coord.   arrival time
          52.04              198.34        234.43
         152.04              344.72        247.37
         252.04              454.65        253.63
         352.04              556.72        260.35
         452.04              665.54        277.42
         552.04              743.77        294.00
         652.04              819.23        308.27
         752.04              954.63        360.90
```

Table 21.--Continued

```
        852.04          1043.82        359.78
        952.04          1120.93        369.04
       1052.04          1324.51        401.25
       1152.04          1574.79        459.85
       1252.04          1729.81        504.61
       1352.04          1783.70        521.92
       1452.04          1696.15        505.40
       1552.04          1796.15        522.07
       1652.04          1896.15        538.73
       1752.04          1996.15        555.40
       1852.04          2096.15        572.07
       1952.04          2196.15        588.73
diffraction results from above shot:
diffractions from subsurface point at x=    710.00
  emergence angle   x-surface coord.  arrival time
        0.00              710.00        294.62
      -10.00              680.19        291.47
      -20.00              649.20        292.26
      -30.00              615.45        294.36
      -40.00              576.53        298.85
      -50.00              528.66        306.39
      -60.00              462.97        321.07
      -70.00              383.90        344.79
      -80.00               99.60        449.77
diffractions from subsurface point at x=    710.00
  emergence angle   x-surface coord.  arrival time
        0.00              710.00        294.62
       10.00              740.99        301.35
       20.00              773.62        306.61
       30.00              816.00        310.48
       40.00              861.04        331.12
       50.00              867.57        338.66
       60.00              922.27        352.65
       70.00             1004.03        354.60
       80.00             1296.92        420.92
diffractions from subsurface point at x=   1060.00
  emergence angle   x-aurface coord.  arrival time
        0.00             1060.00        363.50
       10.00             1100.78        370.68
       20.00             1143.60        374.01
       30.00             1190.74        379.34
       40.00             1245.52        387.32
       50.00             1313.88        399.25
       60.00             1401.81        415.73
       70.00             1767.62        507.15
       80.00             2052.26        592.14
all done. another shot!
rsytrace entry 1                    file=    raytr3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 2000 0 2000 0 100 no 0 0 .15 0 10;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*3 1060 - 710 - 710 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation      time      refrac. angle
     4       1757.96       -400.00        150.87        83.73
```

```
do you want to continue? yes or no:yes
arrival time data for shot at x=   2000.00 depth=    0.00
  from off horizon    4
 x-refractor coord.   x-surface coord.  arrival time
       1757.96             1501.17        296.67
       1657.96             1402.56        311.24
       1557.96             1334.70        323.90
       1457.96             1235.55        325.14
       1357.96              963.01        356.15
       1257.96              788.48        427.11
       1157.96              821.82        412.79
       1057.96              939.23        361.08
        957.96              848.90        402.06
        857.96              724.99        403.32
        757.96              685.54        403.05
        657.96              293.54        475.15
        557.96              -38.27        604.09
        457.96              266.31        490.22
        357.96              141.93        538.80
        257.96              183.00        530.87
        157.96               98.35        546.39
         57.96                5.39        555.51
diffraction results from above shot:
diffractions from subsurface point at x=   1060.00
  emergence angle   x-surface coord.  arrival time
        0.00             1060.00        350.34
       10.00             1100.78        357.52
       20.00             1143.60        360.84
       30.00             1190.74        366.17
       40.00             1245.52        374.16
       50.00             1313.88        386.09
       60.00             1401.81        402.56
       70.00             1767.62        493.99
       80.00             2052.26        578.98
diffractions from subsurface point at x=    710.00
  emergence angle   x-surface coord.  arrival time
        0.00              710.00        400.00
       10.00              740.99        406.73
       20.00              773.62        411.99
       30.00              816.00        415.86
       40.00              861.04        436.50
       50.00              867.57        444.04
       60.00              922.27        458.03
       70.00             1004.03        459.98
       80.00             1296.92        526.30
diffractions from subaurface point at x=    710.00
  emergence angle   x-surface coord.  arrival time
        0.00              710.00        400.00
      -10.00              680.19        396.85
      -20.00              649.20        397.65
      -30.00              615.45        399.75
      -40.00              576.53        404.23
      -50.00              528.66        411.77
      -60.00              462.97        426.46
      -70.00              383.90        450.17
      -80.00               99.60        555.15
all done. another shot!
raytrace entry 1                    file=    raytr3
layd*,ahotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*QUIT
```

coordinate of the point of emergence of the critically reflected ray on the idth horizon (xcrt), and its arrival-time on the idth horizon (tcrt). Hence subroutine CRIT determines only one point on the horizon being calculated.

Because the t-x data point for critical reflection always occurs within the blind zone, CRIT contains a provision for extending data. Although it is usually necessary only to extend data in the direction of the shotpoint to the time intercept, data may be extended in both directions. On the other hand, if data sets have been combined, using subroutine BUILD to include the shotpoint, their extension may not be required. However, such is generally the exception rather than the rule for the execution of CRIT.

The provision to use only artificial data is also provided. This option is usually exercised if it is deemed advisable to approximate the t-x data points by a series of connected straight line segments, or if hidden layers are incorporated.

Subroutine CRIT calls Subroutines SELECT, XTEND, COMP, EQSPACE, either SPLIN1 and SPLINDT or LINDT, WAVED, RATHRU, HSECT, CSECT, and RAYUP.

After the execution of CRIT, the option to execute subroutine ACROSS is exercized. In essence, CRIT determines the critical reflection parameters corresponding to a single point on the t-x graph. Subroutine ACROSS calculates additional points (xir(i) etar(i)) on the (id+1)th horizon corresponding to the rest of the t-x data values, from an assumed velocity for this refracting horizon. The values in the argument for ACROSS, except for xir(i) and etar(i), are transferred from subroutine CRIT. The first five parameters xicrt,---,tcrt are the critical reflection parameters computed in CRIT; id, curv, nval, nstar, xstar, tstar, xprim, tprim and bend2 are described in the CRIT entry list (Appendix A); (x(i),t(i),i=1,nobs) are the original t-x data values defined on horizon 1 reduced to the idth horizon and al(i), bl(i) and cl(i) are cubic spline coefficients used if the option "spl" is exercized. The variable xfar sets the end point of the t-x data for making calculations.

Subroutine ACROSS calls either subroutines SPLIN1 and SPLINDT or LINDT, COMP, RATHRU and XISECT.

Examples of the use of the various options of subroutines CRIT and ACROSS are exhaustively illustrated in the solution of the t-x data set of figure 46. The numerous possibilities are presented not only to demonstrate these options (which are also available for subroutine RECIP) but to illustrate the variety of solutions possible for one-way data by varying the assumed velocity of the refracting horizon. These data are contained in file testcr.org and consist of four separate shot indices, k=1 through 4 (Table 22a). Shotpoints are located at distances 0 and 920 and shot depths have values of 3. For convenience in later operations, these four data sets are combined into two through the execution of subroutine BUILD (table 22b) and the resulting two t-x data sets are printed out in table 22c.

The data have been approximated by a series of straight line segments in two slightly different ways in figures 47a and 47b. Figure 47b will be used to demonstrate the use of artificial data. Interpretations will be

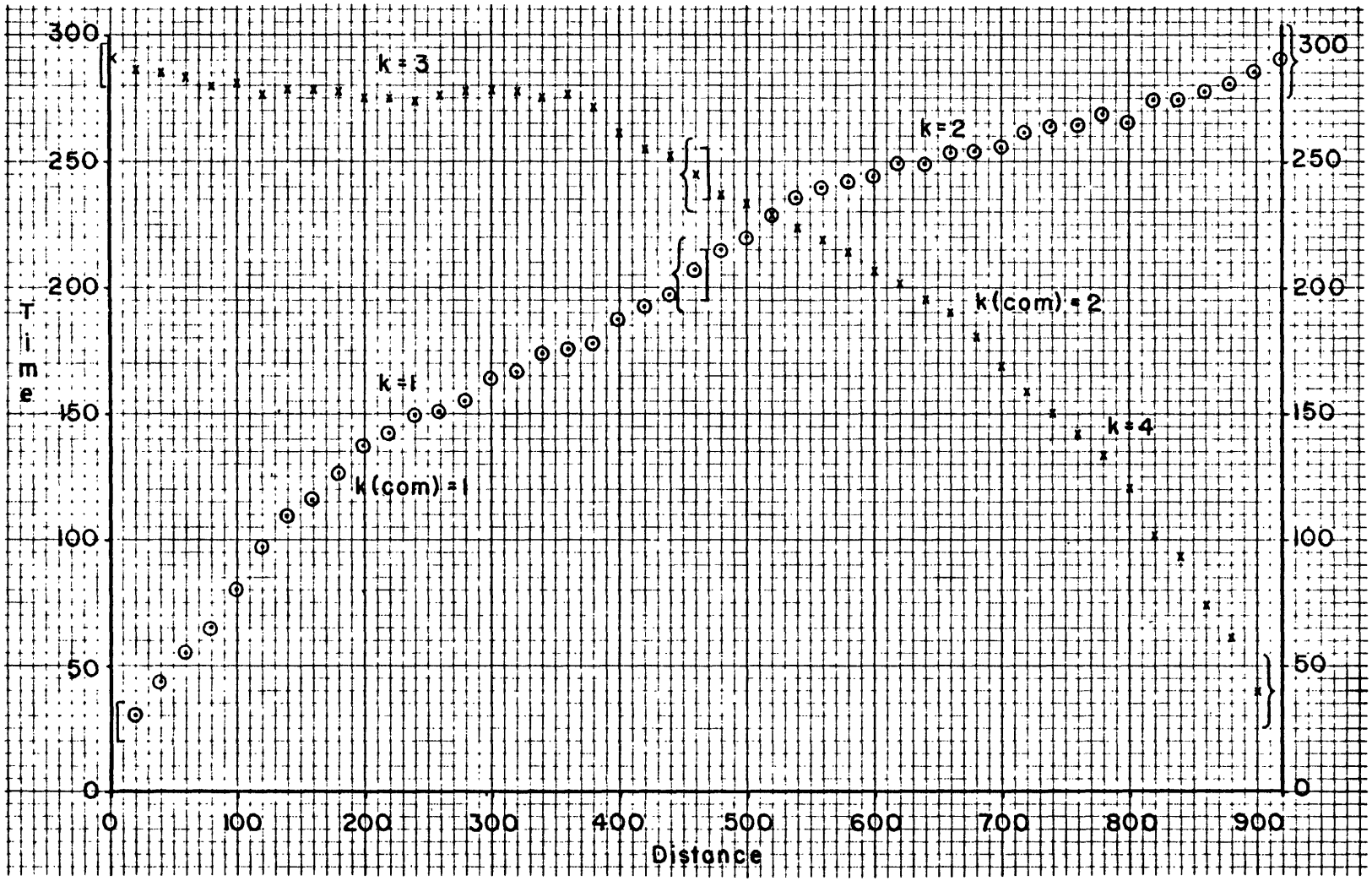

Figure 46.--T-x data set used to illustrate execution of subroutine CRIT.....

Table 22.--(a) Listing of t-x data points in file testcr.org of Figure 46.
(b) Execution of subroutine BUILD to combine the four data sets of (a) into two data sets.
(c) Listing of combined t-x data points.....

(a)

```
print testcr.org
                        testcr.org

 1 23       0.00          3.00        -1.00
     20.00       30.00
     40.00       44.00
     60.00       55.00
     80.00       65.00
    100.00       80.00
    120.00       97.00
    140.00      108.00
    160.00      115.00
    180.00      126.00
    200.00      137.00
    220.00      142.00
    240.00      148.00
    260.00      150.00
    280.00      155.00
    300.00      164.00
    320.00      166.00
    340.00      173.00
    360.00      175.00
    380.00      177.00
    400.00      187.00
    420.00      191.00
    440.00      197.00
    460.00      207.00
 2 24       0.00          3.00        -1.00
    460.00      207.00
    480.00      215.00
    500.00      219.00
    520.00      228.00
    540.00      235.00
    560.00      238.00
    580.00      241.00
    600.00      244.00
    620.00      248.00
    640.00      248.00
    660.00      253.00
    680.00      254.00
    700.00      255.00
    720.00      261.00
    740.00      263.00
    760.00      263.00
    780.00      268.00
    800.00      265.00
    820.00      273.00
    840.00      274.00
    860.00      277.00
    880.00      280.00
    900.00      285.00
    920.00      290.00
 3 24     920.00          3.00        -1.00
      0.00      290.00
     20.00      287.00
     40.00      285.00
     60.00      284.00
     80.00      280.00
    100.00      282.00
    120.00      277.00
    140.00      279.00
    160.00      279.00
    180.00      278.00
    200.00      275.00
    220.00      276.00
    240.00      274.00
    260.00      277.00
    280.00      278.00
    300.00      278.00
    320.00      277.00
    340.00      275.00
    360.00      277.00
    380.00      272.00
    400.00      261.00
    420.00      255.00
    440.00      252.00
    460.00      245.00
 4 23     920.00          3.00        -1.00
    460.00      245.00
    480.00      236.00
    500.00      234.00
    520.00      228.00
    540.00      223.00
    560.00      218.00
    580.00      213.00
    600.00      207.00
    620.00      202.00
    640.00      195.00
    660.00      190.00
    680.00      180.00
    700.00      168.00
    720.00      159.00
    740.00      150.00
    760.00      142.00
    780.00      133.00
    800.00      120.00
    820.00      102.00
    840.00       93.00
    860.00       74.00
    880.00       61.00
    900.00       40.00

seismo
```

(b)

```
type input file name: testor

input complete from 3 dsk files named     testor
type subroutine: build
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2, opt
if opt=1; xsmal,xlarg or if opt=2; add
*1 1 org 0 465 2 org 465 1000 2 0;
combined shot index    1 has   46 jugs
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2, opt
if opt=1; xsmal,xlarg or if opt=2; add
*2 4 org 445 1000 3 org -5 445 2 0;
combined shot index    2 has   46 jugs
build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2,'opt
if opt=1; xsmal,xlarg or if opt=2; add
*0;
output of 3 files to dsk complete
type subroutine: term
type input filename and extension: testcr.com

                    testcr.com
```

(c)

```
1 46       0.00       3.00      -1.00
    20.00      30.00
    40.00      44.00
    60.00      55.00
    80.00      65.00
   100.00      80.00
   120.00      97.00
   140.00     108.00
   160.00     115.00
   180.00     126.00
   200.00     137.00
   220.00     142.00
   240.00     148.00
   260.00     150.00
   280.00     155.00
   300.00     164.00
   320.00     166.00
   340.00     173.00
   360.00     175.00
   380.00     177.00
   400.00     187.00
   420.00     191.00
   440.00     197.00
   460.00     207.00
   480.00     215.00
   500.00     219.00
   520.00     228.00
   540.00     235.00
   560.00     238.00
   580.00     241.00
   600.00     244.00
   620.00     248.00
   640.00     248.00
   660.00     253.00
   680.00     254.00
   700.00     255.00
   720.00     261.00
   740.00     263.00
   760.00     263.00
   780.00     268.00
   800.00     265.00
   820.00     273.00
   840.00     274.00
   860.00     277.00
   880.00     280.00
   900.00     285.00
   920.00     290.00
2 46       920.00       3.00      -1.00
     0.00     290.00
    20.00     287.00
    40.00     285.00
    60.00     284.00
    80.00     280.00
   100.00     282.00
   120.00     277.00
   140.00     279.00
   160.00     279.00
   180.00     278.00
   200.00     275.00
   220.00     276.00
   240.00     274.00
   260.00     277.00
   280.00     278.00
   300.00     278.00
   320.00     277.00
   340.00     275.00
   360.00     277.00
   380.00     272.00
   400.00     261.00
   420.00     255.00
   440.00     252.00
   460.00     245.00
   480.00     236.00
   500.00     234.00
   520.00     228.00
   540.00     223.00
   560.00     218.00
   580.00     213.00
   600.00     207.00
   620.00     202.00
   640.00     195.00
   660.00     190.00
   680.00     180.00
   700.00     168.00
   720.00     159.00
   740.00     150.00
   760.00     142.00
   780.00     133.00
   800.00     120.00
   820.00     102.00
   840.00      93.00
   860.00      74.00
   880.00      61.00
   900.00      40.00

type subroutine: QUIT
```

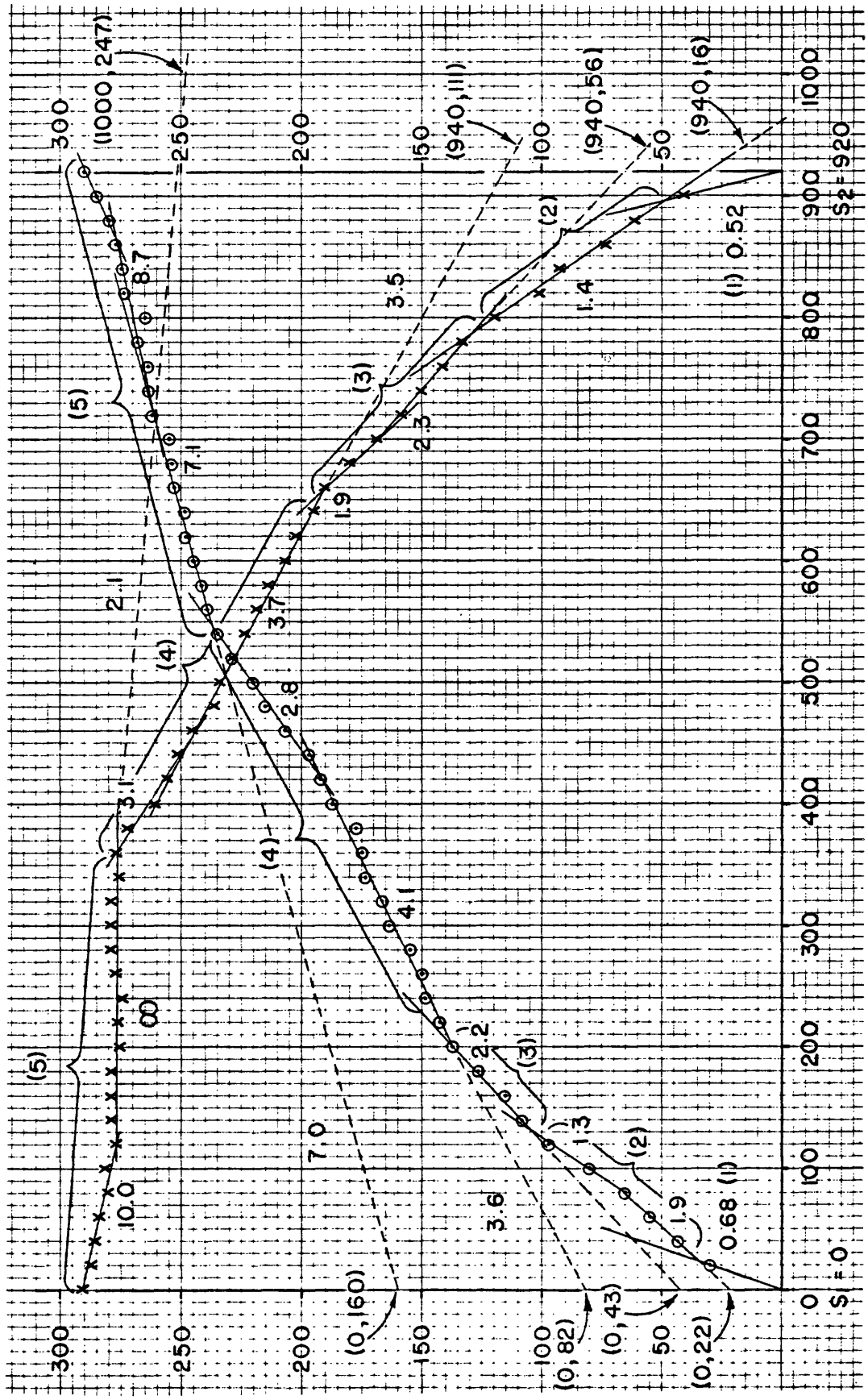

Figure 47.--(a) The t-x data set of figure 46 with points approximated by straight line segments as described in text

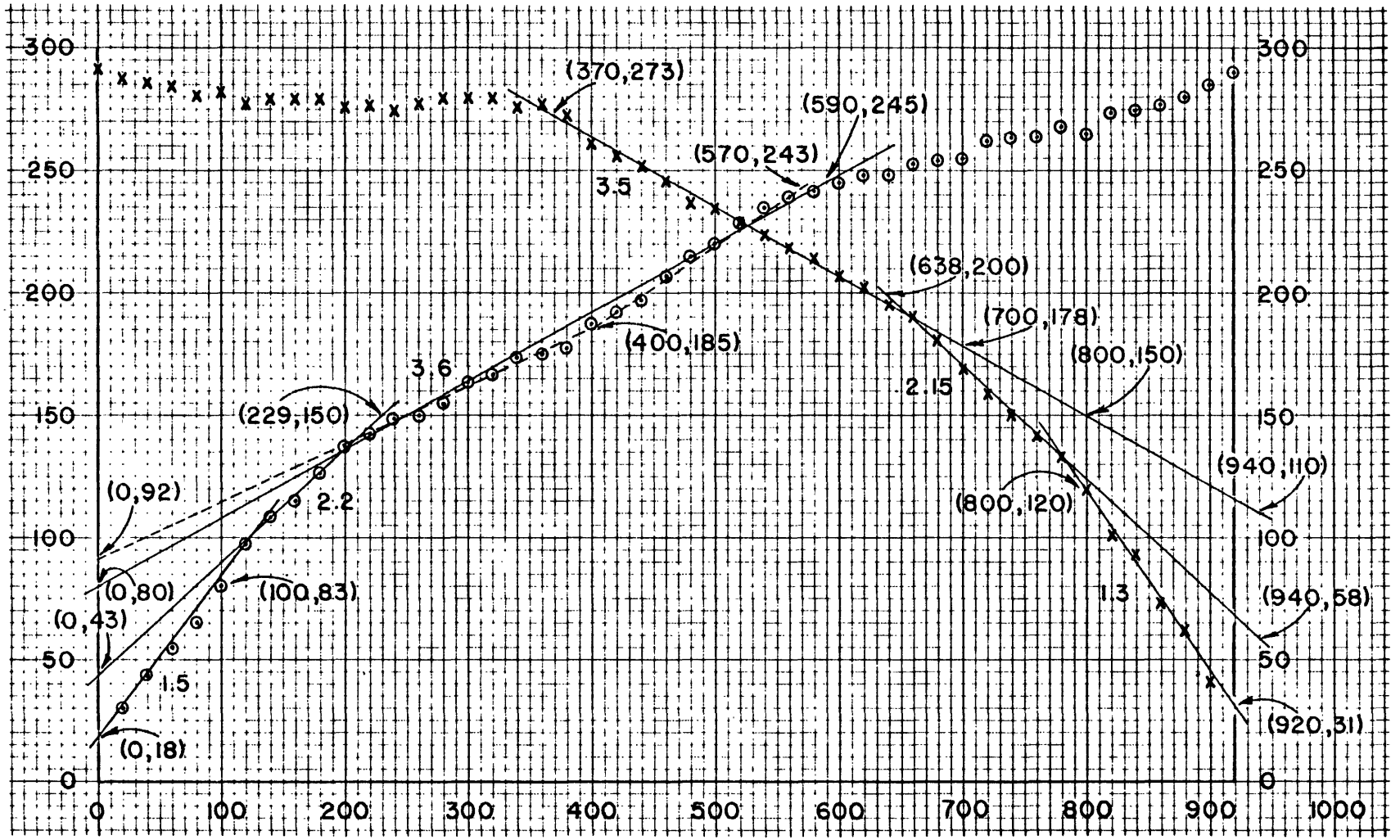

Fig. 47(b) The t-x data set of figure 46 with points approximated by straight line segments representing "artificial" data used for depth calculations.....

made in terms of five layers and in figure 47a the segments corresponding to arrivals from each of their corresponding horizons are indicated by numbered parentheses. In addition, the data for each horizon have been extended through the blind zone to intersect or extend beyond the time axis for each shotpoint. Because opposed or complimentary data are not available to constrain the extension of data, they have been extended as straight lines of approximately the same slope as the lines approximating the points. The t-x value assigned to the end of each of the extended data segments are shown in parentheses. The apparent velocity (inverse of the slope) of each line segment is also shown. These vary from 0.52 for the surface layer for shotpoint S_2 to infinity for a part of layer five also for shotpoint S_2 (fig. 47a).

The ground surface will be assumed to be the uniformly dipping horizon shown as horizon 1 in figures 48a and 48c. Because the velocity of layer 1 is assumed to vary from .65 to .55 based on data near the shotpoints. Execution of subroutine DATAIN to input the model for the first layer into file testcr.lay is shown in table 23a. The horizon has been projected well beyond the ends of the model to insure that all rays will remain confined within it. Printout of this file is shown in table 23b.

Table 24 shows the entry values for subroutines CRIT and ACROSS for eight examples to calculate horizon 2. The entry lists, input form and condition statements are in Appendix A. In examples 1 through 6 the intervals between xmin and xmax are chosen to include only the data points of horizon 2 as seen in figure 47a, and exclude arrivals from points representing other horizons. (Because critical reflection occurs in the blind zone, it need only be required that the interval include two of the points. For example, for k=1, xmin could be 30 and xmax 70. However, then execution of subroutine ACROSS would not continue beyond the limit 70 set by xmax.) In examples 7 and 8, xmin=xmax=0, implying that only artificial data are used in these cases. The straight line segments constructed through the data points in figure 47b represent the artificial data used in these examples. For the execution of CRIT, the value 1.6 was used for the velocity of layer 2 at shotpoints S_1 and 1.5 at S_2. Reasonably small values were used in all cases for delx, delxl and dint. In all examples, except 4 and 8, the value 5 was used for nval. The default option was used for the rest of the input parameters of entry 1. Execution of CRIT and ACROSS for examples 1 through 8 is shown in table 25 and a discussion of each follows.

In entry 1a for example 1, (tables 24 and 25) data are extended to the left using option 1 to the point with coordinates (0,22) (fig. 47a). Data are not extended to the right (entry 1b). The answers are printed out beneath entry 1b in table 25. The distance and elevation coordinates of the critical reflection points are (3.39,-8.95) and the travel time from the shotpoint to the critical reflection point is 10.54. The distance coordinates of the point of emergence of the critically reflected ray on the idth horizon, in this case horizon 1, is 6.43 with arrival-time 25.54. This value can be checked in figure 47a because id=1. It is possible, in unusual cases, that reflection at the critical angle may occur at more than one point. Therefore, the option is included to search for another solution. In this example, the option "yes" was exercized with the response that no

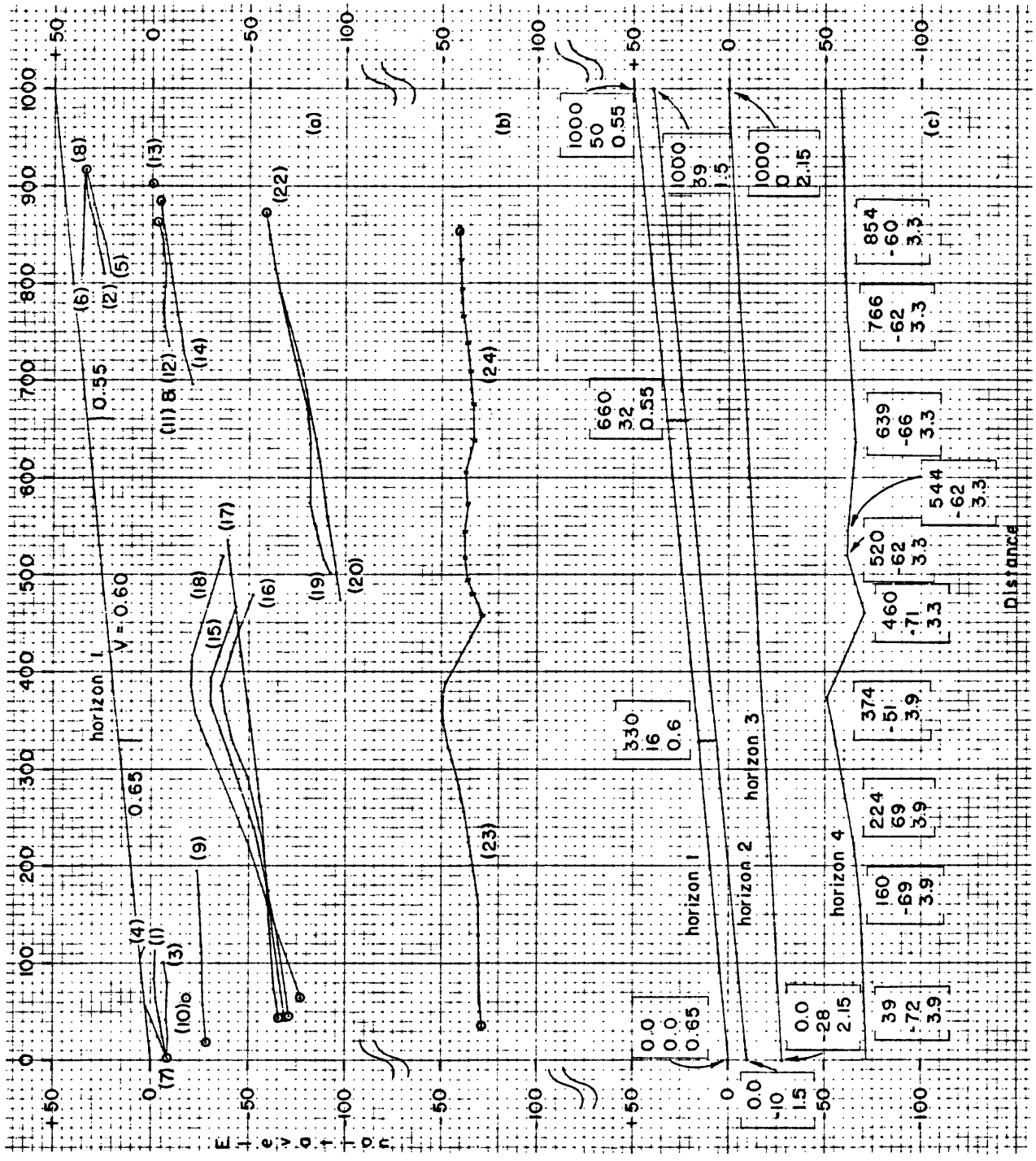

Figure 48.--(a) Velocity models illustrating the results of the various options of subroutine CRIT for interpreting the data shown in Figure 46.
(b) Velocity model for horizon 4 after velocity adjustments as computed in Table 33.
(c) Final velocity model as entered into file testcr.lay.....

Table 23.--(a) Execution of subroutine DATAIN to input velocity model for horizon 1 of Figure 48 into file testcr.lay.
(b) Printout of file testcr.lay.....

(a)

```
seismc

type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: datain
type file extension either org,com,or lay: lay
datain entry 2; i*,nc(i):1 6;
cx(i,j):-90000 0 330 660 1000 90000;
cy(i,j):0 0 16 32 50 50;
cv(i,j):.65 .65 .6 .55 .55 .55;
next layer please.  type 0; or -n; for return to mainline
datain entry 2; i*,nc(i):0;
output of 3 files to dsk complete
datain complete
type subroutine: term
```

(b)

```
type input filename and extension: testcr.lay
```

testcr.lay

```
 1  6  -90000.00          0.00          0.65
      0.00          0.00          0.65
    330.00         16.00          0.60
    660.00         32.00          0.55
   1000.00         50.00          0.55
  90000.00         50.00          0.55


type subroutine: QUIT
```

Table 24.--Input form with input values for eight different examples used to calculate horizon 2 in Table 25 as described in text.....

CRIT AND ACROSS input form File Name tester

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
1.	1	com	lin	1	30	130	1.6	15	15	15
2.	2	com	lin	1	795	895	1.5	-15	15	15
3.	1	com	lin	1	30	130	1.6	15	15	15
4.	1	com	lin	1	30	130	1.6	15	15	15
5.	2	com	lin	1	795	895	1.5	-15	15	15

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	del11)
1.	5					
2.	5					
3.	5					
4.	-					
5.	5					

Entry 1a for CRIT extend data to left

	opt11	±dx11	±fn11(i)
1.	1	1	0 22
2.	0		
3.	2	-50	1.6
4.	3	-50	2
5.	0		

Entry 1b for CRIT extend data to right

	opt1r	±dx1r	±fn1r(i)
1.	0		
2.	1	1	940 16
3.	0		
4.	0		
5.	2	40	-1.4

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
1.	-		
2.	-		
3.	-		
4.	-		
5.	-		

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
1.	1	0	1.6	10	
2.	1	0	1.5	-10	
3.	2	0 80	1.7 1.3	10	
4.	2	0 80	1.3 1.9	10	
5.	1	0	1.7	-10	

Tabel 24.--Continued

CRIT AND ACROSS input form File Name

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
6.	2	com	lin	1	795	895	1.5	-15	15	15
7.	1	com	lin	1	0	0	1.6	15	15	15
8.	2	com	lin	1	0	0	1.5	-15	15	15

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	delll)
6.	5					
7.	5					
8.	-					

Entry 1a for CRIT extend data to left

	optll	±dxll	±fnll(i)
6.	0		
7.	0		
8.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
6.	3	40	5
7.	0		
8.	0		

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
6.	-		
7.	2	0 100	18 83
8.	2	800 920	120 31

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
6.	1	0	1.2	-10	
7.	-				
8.	-				

```
seismc

type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: crit
(1) crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 1 30 130 1.6 15 15 15 5;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*1 1 0 22;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=     3.39 etacrt=     -8.95
critical subsurface arrival time taucrt=     10.54
critical distance on surface xcrt=     6.43
critical surface arrival time tcrt=     25.54
do you want to try for another solution? yes or no:yes
crit condition 10.   gxtry=   120.00 and xfar=   120.00
you are searching for a solution beyond the limits of the data
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 1.6 10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
            16.43                 13.57       -7.91
            26.43                 23.75       -6.87
            36.43                 33.93       -5.84
            46.43                 44.21       -4.67
            56.43                 54.25       -3.78
            66.43                 64.41       -2.80
            76.43                 74.07       -2.48
            86.43                 83.47       -2.30
            96.43                 93.20       -2.40
           106.43                102.94       -2.50
           116.43                112.67       -2.59
(2) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 1 795 895 1.5 -15 15 15 5;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*1 1 940 16;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   917.76 etacrt=     34.77
critical subsurface arrival time taucrt=     15.09
critical distance on surface xcrt=   912.30
critical surface arrival time tcrt=     36.74
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 1.5 -10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
           902.30                907.99       33.79
           892.30                897.95       32.98
           882.30                888.42       31.90
           872.30                878.65       30.92
           862.30                868.86       29.98
           852.30                858.68       29.23
           842.30                848.82       28.42
           832.30                839.23       27.43
           822.30                829.42       26.52
           812.30                819.61       25.61
           802.30                809.81       24.70
(3) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 1 30 130 1.6 15 15 15 5;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*2 -50 1.6;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=     2.62 etacrt=     -8.11
critical subsurface arrival time tsucrt=     8.83
critical distance on surface xcrt=     5.78
critical surface arrival time tcrt=     22.62
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*2 0 80 1.9 1.3 10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
            15.78                 12.40       -8.21
            25.78                 22.18       -8.31
            35.78                 32.26       -8.23
            45.78                 42.12       -8.13
            55.78                 52.08       -8.08
            65.78                 61.52       -8.29
            75.78                 71.31       -8.35
            85.78                 80.03       -8.86
            95.78                 90.25       -7.86
           105.78                100.49       -6.84
           115.78                110.73       -5.80
(4) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 1 30 130 1.6 15 15 15;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*3 -50 2;
```

Table 25.--Execution of subroutines CRIT and ACROSS for the eight examples listed in Table 24.....

Table 25.--Continued

```
crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
$0;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=     3.39 etacrt=    -8.95
critical subsurface arrival time taucrt=    10.54
critical distance on surface xcrt=     6.43
critical surface arrival time tcrt=    25.54
do you want to try for another solution? yes or no:yes
crit condition 10.   gxtry=   120.00 and xfar=   120.00
you are searching for a solution beyond the limits of the data
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2$,xv2(i),v2(i),+or-delx,<nval>
$2 0 80 1.3 1.9 10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
            16.43                 13.96      -6.73
            26.43                 24.53      -4.51
            36.43                 35.09      -2.30
            46.43                 45.66      -0.08
            56.43                 56.30       2.32
across condition 6 at xa=   66.43
across condition 6 at xa=   76.43
across condition 6 at xa=   86.43
across condition 6 at xa=   96.43
           106.43                106.35       5.02
           116.43                115.07       3.20
(5) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,dell>
$2 com lin 1 795 895 1.5 -15 15 15 5;
crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
$0;
crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
$2 40 -1.4;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   917.36 etacrt=    34.36
critical subsurface arrival time taucrt=    16.02
critical distance on surface xcrt=   912.12
critical surface arrival time tcrt=    38.16
do you want to try for another solution? yes or no:yes
crit condition 10.   gxtry=   800.00 and xfar=   800.00
you are searching for a solution beyond the limits of the data
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2$,xv2(i),v2(i),+or-delx,<nval>
$1 0 1.7 -10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
           902.12                907.64      33.23
           892.12                897.93      32.11
           882.12                888.70      30.79
           872.12                879.11      29.44
           862.12                869.54      28.20
           852.12                859.62      27.03
           842.12                849.98      25.77
           832.12                840.60      24.36
           822.12                831.02      23.02
           812.12                821.43      21.68
           802.12                811.84      20.34
(6) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,dell>
$2 com lin 1 795 895 1.5 -15 15 15 5;
crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
$0;
crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
$3 40 5;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   917.49 etacrt=    34.57
critical subsurface arrival time taucrt=    15.58
critical distance on surface xcrt=   912.24
critical surface arrival time tcrt=    37.38
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2$,xv2(i),v2(i),+or-delx,<nval>
$1 0 1.2 -10;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
           902.24                907.12      34.81
           892.24                896.74      35.06
           882.24                886.74      35.03
           872.24                876.46      35.04
           862.24                866.13      35.17
           852.24                855.59      35.44
           842.24                845.24      35.60
           832.24                835.05      35.56
           822.24                824.75      35.64
           812.24                814.44      35.72
           802.24                804.13      35.80
(7) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,dell>
$1 com lin 1 0 0 1.6 15 15 15 5;
crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
$0;
crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
$0;
crit entry 2; nobs$,x(i),t(i)
$2 0 100 18 83;
good news: the critical points have been found.they are
critical aubsurface coordinates xicrt=     2.30 etacrt=    -7.81
critical subsurface arrival time taucrt=     8.20
critical distance on surface xcrt=     5.60
critical surface arrival time tcrt=    21.64
```

Table 25.--Continued

```
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:no
do you wish to try anothor shot? yes or no:yes
(8) crit entry 1   file=testcr     ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 1 0 0 1.5 -15 15 15;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 800 920 120 31;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   917.72 etacrt=    34.78
critical subsurface arrival time taucrt=    15.10
critical distance on surface xcrt=   912.31
critical surface arrival time tcrt=    36.70
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:no
do you wish to try anothor shot? yes or no:no
type subroutine: QUIT
```

additional solution could be found (CRIT condition 10) and that the search ended at distance coordinate 120 on horizon id. The option to execute subroutine ACROSS is exercised. ACROSS entry 1 shows that a single velocity of 1.6 will be used for horizon 2 (tables 24 and 25) and that solutions will be determined at intervals of 10. The answers are printed in the columns beneath the input, (table 25) showing coordinates (xir(i),etar(i)) on horizon (id+1), which satisfy the data and the distance coordinate xr(i) of the corresponding point on the idth horizon. The critical reflection point and the values (xir(i), etar(i)) have been plotted in figure 48a(1). We note that initially the plotted horizon dips toward the surface, in response to the 1.9 apparent velocity for the initial part of the t-x data compared to the 1.6 velocity used. Thereafter, the horizon changes dip in response to the 1.3 apparent velocity. The use of nval=5 in CRIT entry 1 has served to filter the change in dip. Had nval been set to 2 or had dint been assigned a smaller value in ACROSS entry 1, the dips would probably have been steeper with more abrupt changes.

Example 2 is equivalent to example 1, only for shotpoint S_2. The results shown plotted in figure 48a(2) do not show the effects of scatter of the t-x data due to the filtering action of the use of 5 points for approximating the t-x data.

In example 3, the option to extend the data, using the velocity criteria, is exercised. The data are extended a distance of -50 with a velocity of 1.6, thus projecting them 10 distance units behind the shotpoint (e.g. the distance coordinate of the first t-x value used is 40 (fig. 47a) and the shotpoint is at 0). Data are not extended to the right. In executing subroutine ACROSS, a velocity of 1.9 was used between distance coordinate 0 and 80 and a velocity of 1.3 thereafter. This caused the computed horizon to approximately parallel the surface as seen in figure 48a(3). In effect, it was assumed that the change of apparent velocity of horizon 2 for shotpoint S_1 was caused by a change in velocity rather than a change in dip.

In example 4, the least squares option was used to extend the data beyond the intercept using only two points. This is approximately equivalent of extending the data using the first two points for horizon 2 at distances 40 and 60 in figure 47a. In executing subroutine ACROSS, a velocity of only 1.3 was used as far as distance coordinate 80 causing the calculated values for horizon 2 to intersect the surface (ACROSS condition 6) as shown in figure 48a(4). The use of 1.9 velocity beyond 80 causes the computed horizon to again assume values deeper than horizon 1.

The velocity criteria is used for extending the data in example 5. A negative velocity is used because, according to sign conventions, the slope of the t-x graph at S_2 is negative. Had a positive value been used, the slope of the t-x graph would have abruptly changed direction at point (880,61) and intersected the time axis at a time of approximately 90. A velocity of 1.7 in the execution of subroutine ACROSS has resulted in a steeper dip (fig. 48a(5) for the computed horizon than for example 2 for which 1.5 was used.

The least squares criteria is used for extending data in example 6. The velocity of 1.2 in subroutine ACROSS results in a computed horizon (fig. 48a(6) dipping rapidly toward the surface.

In examples 7 and 8 only artificial data are used (fig. 47b). In both cases, the data are approximated by straight line segments of our choosing. The coordinates used (0,18) and (100,83) in example 7, and (800,120) and (920,31) in example 8 are appropriately entered in CRIT entry 2, and the computed results shown in figure 48a(7 and 8). In neither case was subroutine ACROSS executed.

From the results of the various examples for horizon 2 shown in figure 48a, which at best provides a very limited insight into its nature, a decision must be made regarding its configuration between the two shotpoints S_1 and S_2. The decision made is shown in figure 48c which in effect simply connects the critical reflection points calculated below the two shotpoints. In practice, subroutine ACROSS is seldom executed for short arrival-time segments such as for horizon 2 in this problem. Instead, a considerable effort is made to determine a correct velocity and time intercept from opposed plots and artificial data used in the execution of CRIT to compute the critical reflection point. Horizon 2 is input into file testcr.lay through execution of subroutine DATAIN (table 26a) and this file is printed out in table 26b.

From the t-x plots in figures 47a and b, the velocity of layer 3 may be estimated to be roughly 2.2. Table 27 shows the entry values for subroutines CRIT and ACROSS for examples 9 through 14 demonstrating the calculation of this horizon.

Depth calculations below S_1 are shown by examples 9 and 10 (table 28) using the artifical data of figure 47b. In example 9, a 2.2 velocity is used for both CRIT and ACROSS and the results are shown in Figure 48a(9). The initial purpose of example 10 was to demonstrate the insensitivity of the results of CRIT to different values of velocity v2 used. In this case 1.7 was used instead of 2.2 and the comparison is shown plotted as a circle in figure 48a(10). In this instance the difference obtained between the two velocities appears to be significant which is the exception rather than the rule. The reason for this apparent anomalous result will not be investigated here. In example 9, the result of exercizing the option of printing the "star" points is shown in table 28. This printout shows the results of subroutine WAVED for ten equally spaced points on the extended t-x graph in reducing them from horizon 1 to horizon 2. For instance, the point with initial t-x coordinates (0,43) is translated to a point at (-2.7,27.1) on horizon 2. The "star" points can be used for analyzing which parts of an initial t-x graph arise from structure at depth.

Examples 11 through 14 calculate depths for horizon 3 below S^2. Example 11 extends the data using the specific point at t-x coordinates (940,56), (fig. 47a) a velocity of 1.9 in subroutine CRIT and 2.1 in ACROSS. Results are plotted in figure 48a(11). Example 12 is the same as 11, only a cubic spline function is used for approximating the data rather than least squares straight line segments.

```
seismc
type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: datain
type file extension either org,com,or lay: lay
datain entry 2; i*,nc(i):2 4;
cx(i,j):-90000 0 1000 90000;
cy(i,j):-10 -10 39 39;
cv(i,j):1.5 1.5 1.5 1.5;
next layer please.  type 0; or -n; for return to mainline
datain entry 2; i*,nc(i):0;
output of 3 files to dsk complete
datain complete
type subroutine: term
type input filename and extension: testcr.lay
```

(a)

(b)

testcr.lay

```
1  6  -90000.00        0.00        0.65
      0.00        0.00        0.65
    330.00       16.00        0.60
    660.00       32.00        0.55
   1000.00       50.00        0.55
  90000.00       50.00        0.55
2  4  -90000.00      -10.00        1.50
      0.00      -10.00        1.50
   1000.00       39.00        1.50
  90000.00       39.00        1.50
```

```
type subroutine: QUIT
```

Table 26.--(a) Execution of subroutine DATAIN to imput the interpreted velocity section of horizon 2 into file tcrit.lay.
(b) Printout of file testcr.lay showing two horizons.....

Table 27.--Input form with the input values for six different examples used to calculate horizon 3 in Table 28 as described in text.....

CRIT AND ACROSS input form File Name _____

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
9.	1	com	lin	2	0	0	2.2	10	10	10
10.	1	com	lin	2	0	0	1.7	10	10	10
11.	2	com	lin	2	650	795	1.9	-15	10	10
12.	2	com	spl	2	650	795	1.9	-15	10	10

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	delll)
9.	-					
10.	-					
11.	5					
12.	-					

Entry 1a for CRIT extend data to left

	optll	±dxll	±fnll(i)
9.	0		
10.	0		
11.	0		
12.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)	
9.	0			
10.	0			
11.	1	1	940	56
12.	1	1	940	56

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs		t(i) i = 1, nobs	
9.	2	0	229	43	150
10.	2	0	229	43	150
11.	-				
12.	-				

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
9.	1	0	2.2	20	
10.	-				
11.	1	0	2.1	-20	
12.	-				

Tabel 27.--Continued

CRIT AND ACROSS input form File Name tester

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
13.	2	com	lin	2	650	795	2.5	-15	10	10
14.	2	com	lin	2	0	0	2.15	-15	10	10

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	dell1)
13.	5					
14.	-					

Entry 1a for CRIT extend data to left

	optl1	±dxl1	±fnl1(i)
13.	0		
14.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
13.	1	1	940 56
14.	0		

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
13.	-		
14.	2	638 940	200 58

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
13.	-				
14.	1	0	2.15	-20	

Table 28.--Execution of subroutines CRIT and ACROSS for the six examples listed in Table 27.....

```
seismc

type input file name: tester

input complete from 3 dsk files named     tester
type subroutine: crit
(9) crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 2 0 0 2.2 10 10 10;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 0 229 43 150;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=    20.04 etacrt=   -28.00
critical subsurface arrival time taucrt=    28.07
critical distance on surface xcrt=    37.79
critical surface arrival time tcrt=    45.82
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 2.2 20;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
            57.79                 39.56     -27.57
            77.79                 59.07     -27.14
            97.79                 78.58     -26.70
           117.79                 98.09     -26.27
           137.79                117.60     -25.84
           157.79                137.11     -25.41
           177.79                156.62     -24.98
           197.79                176.13     -24.54
           217.79                195.64     -24.11
do you want the star points printed?yes
the path of the special xstar points from horizons 1 to id:
    x(1)  time(1)     x(id)  time(id)
     0.0      43.0      -2.7      27.1
    25.4      54.9      22.8      38.8
    50.9      66.8      48.2      50.7
    76.3      78.7      73.6      62.6
   101.8      90.6      99.1      74.5
   127.2     102.4     124.5      86.4
   152.7     114.3     150.0      98.3
   178.1     126.2     175.4     110.2
   203.6     138.1     200.9     122.1
   229.0     150.0     226.3     134.1
(10) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 2 0 0 1.7 10 10 10;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 0 229 43 150;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=    66.28 etacrt=   -18.19
critical subsurface arrival time taucrt=    53.46
critical distance on surface xcrt=    76.96
critical surface arrival time tcrt=    64.15
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:no
do you wish to try anothor shot? yes or no:yes
(11) crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 2 650 795 1.9 -15 10 10 5;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*1 1 940 56;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   863.61 etacrt=    -3.82
critical subsurface arrival time taucrt=    59.15
critical distance on surface xcrt=   823.15
critical surface arrival time tcrt=    94.45
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 2.1 -20;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)     etar(i)
           803.15                842.74      -5.22
           783.15                821.16      -6.18
           763.15                796.81      -6.28
           743.15                774.23      -5.80
           723.15                756.98      -6.00
           703.15                745.20      -7.08
           683.15                734.24      -9.15
do you want the star points printed?no
(12) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com spl 2 650 795 1.9 -15 10 10;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*1 1 940 56;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   863.70 etacrt=    -3.83
critical subsurface arrival time taucrt=    59.11
critical distance on surface xcrt=   823.16
```

Tabel 28.--Continued

```
critical surface arrival time tcrt=    94.45
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 2.1 -20;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
          803.16                 844.00       -5.28
          783.16                 821.77       -6.85
          763.16                 795.77       -7.03
          743.16                 773.03       -5.15
          723.16                 755.62       -5.31
          703.16                 745.85       -4.28
          683.16                 736.43       -9.81
do you want the star points printed?no
do you wish to try another shot? yes or no:yes
(13) crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 2 650 795 2.5 -15 10 10 5;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*1 1 940 56;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   903.23 etacrt=    -0.81
critical subsurface arrival time taucrt=    80.28
critical distance on surface xcrt=   863.93
critical surface arrival time tcrt=    74.55
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:no
(14) do you wish to try another shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 2 0 0 2.15 -15 10 10;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 638 940 200 58;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   885.31 etacrt=    -4.45
critical subsurface arrival time taucrt=    48.74
critical distance on surface xcrt=   844.81
critical surface arrival time tcrt=    84.79
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 2.15 -20;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
          824.81                 865.81       -5.87
          804.81                 846.32       -7.30
          784.81                 826.81       -8.72
          764.81                 807.32      -10.15
          744.81                 787.82      -11.57
          724.81                 768.32      -13.00
          704.81                 748.83      -14.42
          684.81                 729.34      -15.85
          664.81                 726.29      -15.76
          644.81                 697.99      -20.87
do you want the star points printed?no
do you wish to try another shot? yes or no:QUIT
```

Example 13 is the same as 11, except that a velocity of 2.5 is used in CRIT instead of 1.9. The results plotted in figure 48a(13), in this instance, demonstrate the insensitivity of the calculated depth of the critical reflection point to different values used for v2.

Example 14 uses the artificial line segments of figure 47b for depth calculations and a velocity of 2.15 in subroutine ACROSS. Results are shown in figure 47a(14).

As with horizon 2, because of the limited results, a decision must be made regarding the configuration of horizon 3 between the two shotpoints. The conclusion is shown in figure 48c. These results are shown as input into file testcr.lay and the file printed out in table 29.

The data of figures 47a and b could suggest a velocity of about 3.6 for layer 4. On the other hand, a straight line continuation of the data points representing this horizon to the reciprocal point would result in a considerable mistie, implying a considerably more complex velocity model than might be anticipated. Nevertheless, initial computations are made using a 3.6 velocity in both subroutines CRIT and ACROSS. Examples 15 through 21 show computation of horizon 4 using varying input parameters listed in table 30. Execution of these values is given in table 31. Examples 15 through 18 show results from shotpoint S_1.

In example 15, data are extended using option 1 and the t-x point with coordinates (0,82) (fig. 47a). Example 16 uses the least squares criteria of option 3, utilizing 16 points for the straight line approximation. For example 17, the artificial data consist of a single straight line segment with endpoint coordinates (0,80) and (590,245) (fig. 47b). The results of each example are shown in figure 48a(15) through (18) and each shows an arched feature with the exception of example 17, for which the data were approximated by a single straight line.

Examples 19 and 20 demonstrate results from shotpoint S_2, once again using a 3.6 velocity. Example 19 (fig. 47a) uses option 1 for extending data and example 20 is a straight line approximation using the artificial data shown in Figure 47b. Results for these examples are also given in figure 48a(19) and (20).

Table 31 shows the results of setting nprin=1 (example 16). The additional values printed out show the t-x values reduced to horizons 2 and 3. In examples 16, 17 and 18, the star points have also been printed out.

Inspection of the results from both shotpoints in the examples for horizon 4 (fig. 48a) shows a large mistie near distance coordinate 500. This could be the result of offset, possibly across a fault. However, a better explanation probably involves a change in the velocity value used in subroutine ACROSS. A better tie is produced by increasing the velocity of the refracting horizon for shotpoint S_1 and lowering it for S_2. Therefore, a value of 3.9 was used for S_1 and 3.3 for S_2. Examples 22 and 23 use these values (tables 32 and 33) and the results are shown as the continuous horizon in fig. 48b. The tie between shotpoints in this case is acceptable.

Table 29.--(a) Execution of subroutine DATAIN to input the interpreted velocity section of horizon 3 into file testcr.lay.
(b) Printout of file testcr.lay showing three horizons.....

```
seismc
(a) type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: datain
type file extension either org,com,or lay: lay
datain entry 2; i*,nc(i):3 4;
cx(i,j):-90000 0 1000 90000;
cy(i,j):-28 -28 0 0;
cv(i,j):2.15 2.15 2.15 2.15;
next layer please.  type 0; or -n; for return to mainline
datain entry 2; i*,nc(i):0;
output of 3 files to dsk complete
datain complete
type subroutine: term
(b) type input filename and extension: testcr.lay
```

testcr.lay

```
 1  6  -90000.00        0.00        0.65
       0.00        0.00        0.65
     330.00       16.00        0.60
     660.00       32.00        0.55
    1000.00       50.00        0.55
   90000.00       50.00        0.55
 2  4  -90000.00      -10.00        1.50
       0.00      -10.00        1.50
    1000.00       39.00        1.50
   90000.00       39.00        1.50
 3  4  -90000.00      -28.00        2.15
       0.00      -28.00        2.15
    1000.00        0.00        2.15
   90000.00        0.00        2.15
```

```
type subroutine: QUIT
```

Table 30.--Input form with input values for eight different examples used to calculate horizon 4 in Table 31 as described in text.....

CRIT AND ACROSS input form File Name tester

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
15.	1	com	lin	3	210	550	3.6	30	20	20
16.	1	com	lin	3	210	550	3.6	30	20	20
17.	1	com	lin	3	0	0	3.6	30	20	20
18.	1	com	lin	3	0	0	3.6	30	20	20
19.	2	com	lin	3	350	670	3.6	-30	20	20

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	del11)
15.	4					
16.	4	no	no	1		
17.	-					
18.	-					
19.	4					

Entry 1a for CRIT extend data to left

	optll	±dxll	±fnll(i)
15.	1	1	0 82
16.	3	-250	16
17.	0		
18.	0		
19.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
15.	0		
16.	0		
17.	0		
18.	0		
19.	1	1	940 111

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
15.	-		
16.	-		
17.	2	0 590	80 245
18.	3	0 400 570	92 185 243
19.	-		

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
15.	1	0	3.6	30	
16.	1	0	3.6	30	
17.	1	0	3.6	30	
18.	1	0	3.6	30	
19.	1	0	3.6	-30	

Table 30.--Continued

CRIT AND ACROSS input form File Name tester

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
20.	2	com	lin	3	0	0	3.6	-30	20	20
21.	2	com	lin	3	0	0	3.6	-30	20	20
22.	2	com	lin	3	0	0	3.6	-30	20	20

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	delll)
20.	4					
21.	4					
22.	4					

Entry 1a for CRIT extend data to left

	optll	±dxll	±fnll(i)
20.	0		
21.	0		
22.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
20.	0		
21.	0		
22.	0		

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
20.	2	370 940	273 110
21.	2	800 940	150 110
22.	2	700 940	178 110

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
20.	1	0			
21.	—				
22.	—				

Table 31.--Execution of subroutines CRIT and ACROSS for the eight examples listed in Table 30.....

```
seismo

type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: crit
(15) crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 3 210 550 3.6 30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*1 1 0 82;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=    48.63 etacrt=   -71.10
critical subsurface arrival time taucrt=    50.78
critical distance on surface xcrt=    77.41
critical surface arrival time tcrt=    75.73
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.6 30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
           xr(i)                 xir(i)     etar(i)
         107.41                  79.54      -68.81
         137.41                 110.55      -66.50
         167.41                 141.97      -64.01
         197.41                 175.20      -60.83
         227.41                 207.66      -56.93
         257.41                 240.14      -52.56
         287.41                 273.01      -47.60
         317.41                 306.46      -41.62
         347.41                 339.26      -35.25
         377.41                 369.93      -30.76
         407.41                 394.85      -30.88
         437.41                 418.59      -34.17
         467.41                 441.45      -38.93
         497.41                 466.70      -43.96
do you want the star points printed?no
(16) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 3 210 550 3.6 30 20 20 4 no no 1;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*3 -250 16;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
intermediate arrival times from sub waved
layer  distance time   distance time   distance time   distance time
   2
       -31.8   53.6      -11.8   59.2       8.6   64.8       28.6   70.4
        48.6   76.0       68.6   81.6      88.6   87.2      108.7   92.8
       128.7   98.4      148.7  104.0      168.7  109.6      188.7  115.2
       208.4  121.5      228.5  127.2      248.8  132.5      268.8  138.6
       288.7  143.6      308.6  149.1      329.2  154.1      349.4  156.4
       369.2  161.4      388.8  166.1      408.6  171.5      428.5  178.8
       448.1  185.9      468.3  192.8      488.5  200.6      508.4  207.7
       528.4  214.6      538.4  218.1
shot originally at x=   -30.0 now at x=   -31.8
   3
       -40.1   40.4      -20.1   45.9        0.8   51.5       20.6   56.7
        40.4   62.0       60.2   67.3       80.0   72.6       99.9   77.9
       119.7   83.2      139.5   88.5      159.3   93.8      178.6   99.2
       198.3  104.7      218.5  110.2      238.7  115.8      258.7  121.0
       278.8  126.1      299.7  131.1      321.2  135.4      341.6  139.6
       360.4  143.5      378.1  147.6      395.8  152.7      413.4  158.1
       432.2  164.4      451.7  171.3      471.5  178.1      491.8  185.1
       511.5  191.9      521.7  195.4
shot originally at x=   -30.0 now at x=   -40.1
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=    43.12 etacrt=   -66.35
critical subsurface arrival time taucrt=    47.38
critical distance on surface xcrt=    69.70
critical surface arrival time tcrt=    69.83
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.6 30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
           xr(i)                 xir(i)     etar(i)
          99.70                  73.71      -64.60
         129.70                 104.11      -62.91
         159.70                 133.77      -61.58
         189.70                 163.20      -60.56
         219.70                 193.77      -59.52
         249.70                 226.16      -57.63
         279.70                 258.95      -54.72
         309.70                 295.55      -48.95
         339.70                 328.87      -42.34
         369.70                 357.72      -38.03
         399.70                 385.05      -36.75
         429.70                 406.67      -38.99
         459.70                 430.07      -43.19
         489.70                 454.66      -47.93
         519.70                 479.07      -52.46
do you want the star points printed?yes
the path of the special xstar points from horizons 1 to id:
     x(1)  time(1)     x(id)  time(id)
    -30.0     69.2     -40.1     40.4
     33.3     86.9      24.3     57.7
     96.7    104.6      87.1     74.5
    160.0    122.3     149.8     91.3
    223.3    140.7     212.2    108.5
    286.7    158.1     276.0    125.4
```

Table 31.--Continued

```
     350.0      172.8      342.8      139.8
     413.3      189.2      399.5      153.8
     476.7      212.0      458.4      173.6
     540.0      234.6      521.7      195.4
(17) do you wish to try anothor shot? yes or no:yes
     crit entry 1   file=tester    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,$
      <nval,bend1,bend2,nprin,nstar,delll>
     $1 com lin 3 0 0 3.6 30 20 20;
     crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
     $0;
     crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
     $0;
     crit entry 2; nobs$,x(i),t(i)
     $2 0 590 80 245;
     good news: the critical points have been found.they are
     critical subsurface coordinates xicrt=    45.58 etacrt=   -69.46
     critical subsurface arrival time taucrt=    49.24
     critical distance on surface xcrt=    74.40
     critical surface arrival time tcrt=    73.53
     do you want to try for another solution? yes or no:no
     try across for solutions along t-x graph? yes or no:yes
     across entry 1; nv2$,xv2(i),v2(i),+or-delx,<nval>
     $1 0 3.6 30;
          the answers are:
          coordinate id horizon   coordinates of id+1 horizon
                xr(i)                xir(i)     etar(i)
              104.40                 76.15      -67.76
              134.40                106.72      -66.06
              164.40                137.29      -64.35
              194.40                167.86      -62.65
              224.40                198.43      -60.95
              254.40                229.08      -59.23
              284.40                261.40      -57.13
              314.40                293.79      -54.13
              344.40                323.75      -51.03
              374.40                353.64      -48.87
              404.40                384.21      -47.18
              434.40                414.77      -45.48
              464.40                445.34      -43.78
              494.40                475.91      -42.09
              524.40                506.47      -40.39
              554.40                537.04      -38.70
     do you want the star points printed?yes
     the path of the special xstar points from horizons 1 to id:
         x(1)  time(1)     x(id)  time(id)
          0.0      80.0      -9.6      51.3
         65.6      98.3      56.3      68.7
        131.1     116.7     121.3      86.1
        196.7     135.0     186.3     103.4
        262.2     153.3     251.3     120.8
        327.8     171.7     317.7     138.0
        393.3     190.0     381.4     154.4
        458.9     208.3     446.4     171.8
     524.4      226.7      511.4      189.1
     590.0      245.0      576.4      206.5
(18) do you wish to try anothor shot? yes or no:yes
     crit entry 1   file=tester    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
      <nval,bend1,bend2,nprin,nstar,delll>
     $1 com lin 3 0 0 3.6 30 20 20;
     crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
     $0;
     crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
     $0;
     crit entry 2; nobs$,x(i),t(i)
     $3 0 400 570 92 185 243;
     good news: the critical points have been found.they are
     critical subsurface coordinates xicrt=    65.09 etacrt=   -77.27
     critical subsurface arrival time taucrt=    58.33
     critical distance on surface xcrt=    91.06
     critical surface arrival time tcrt=    85.29
     do you want to try for another solution? yes or no:no
     try across for solutions along t-x graph? yes or no:yes
     across entry 1; nv2$,xv2(i),v2(i),+or-delx,<nval>
     $1 0 3.6 30;
          the answers are:
          coordinate id horizon   coordinates of id+1 horizon
                xr(i)                xir(i)     etar(i)
              121.06                 97.56      -71.50
              151.06                130.03      -65.74
              181.06                162.49      -59.98
              211.06                194.96      -54.22
              241.06                227.43      -48.45
              271.06                260.00      -42.61
              301.06                293.59      -35.93
              331.06                326.70      -28.61
              361.06                358.34      -22.77
              391.06                388.98      -20.15
              421.06                416.29      -21.39
              451.06                441.94      -24.97
              481.06                467.54      -28.77
              511.06                493.22      -32.47
              541.06                518.97      -36.10
     do you want the star points printed?no
(19) do you wish to try anothor shot? yes or no:yes
     crit entry 1   file=tester    ; k$,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
      <nval,bend1,bend2,nprin,nstar,delll>
     $2 com lin 3 350 670 3.6 -30 20 20 4;
     crit entry 1a; optll$,+or-dxll,+or-fnlr(i)
     $0;
     crit entry 1b; optlr$,+or-dxlr,+or-fnlr(i)
     $1 1 940 111;
     good news: the critical points have been found.they are
     critical subsurface coordinates xicrt=   870.85 etacrt=   -58.75
     critical subsurface arrival time taucrt=    71.52
     critical distance on surface xcrt=   822.41
     critical surface arrival time tcrt=   105.19
```

Table 31.--Continued

```
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.6 -30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
          792.41                842.47       -61.39
          762.41                814.09       -64.04
          732.41                786.14       -66.68
          702.41                759.82       -69.57
          672.41                735.02       -72.78
          642.41                706.40       -76.47
          612.41                670.83       -79.62
          582.41                637.88       -81.17
          552.41                604.68       -81.77
          522.41                575.18       -82.18
          492.41                552.41       -83.40
          462.41                528.33       -85.67
          432.41                514.71       -88.19
          402.41                500.69       -91.87
do you want the star points printed?no
(20) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 3 0 0 3.6 -30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 370 940 273 110;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   872.45 etacrt=   -57.96
critical subsurface arrival time taucrt=    70.82
critical distance on surface xcrt=   823.66
critical surface arrival time tcrt=   104.34
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.6 -30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
          793.66                844.36       -60.88
          763.66                816.27       -63.79
          733.66                790.25       -66.70
          703.66                764.36       -70.01
          673.66                737.34       -73.79
          643.66                707.38       -77.64
          613.66                676.60       -80.81
          583.66                646.79       -83.36
          553.66                618.12       -85.71
          523.66                589.45       -88.06
          493.66                560.78       -90.41
          463.66                532.11       -92.76
          433.66                503.44       -95.11
          403.66                474.77       -97.46
do you want the star points printed?no
(21) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 3 0 0 3.6 -30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 800 940 150 110;
crit condition 10.   gxtry=   800.00 and xfar=   821.07
you are searching for a solution beyond the limits of the data
try across for solutions along t-x graph? yes or no:no
(22) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=testcr    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 3 0 0 3.6 -30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
crit entry 2; nobs*,x(i),t(i)
*2 700 940 178 110;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=   871.94 etacrt=   -57.53
critical subsurface arrival time taucrt=    70.77
critical distance on surface xcrt=   824.25
critical surface arrival time tcrt=   103.79
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:no
do you wish to try anothor shot? yes or no:QUIT
```

CRIT AND ACROSS input form File Name tester

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
23.	1	com	lin	3	210	550	3.9	30	20	20
24.	1	com	lin	3	350	670	3.3	-30	20	20

Entry 1 continued (optional)

	(nval	bend1 yes/no	bend2 yes/no	nprin 0/1	nstar	dell1)
23.	4					
24.	4					

Entry 1a for CRIT extend data to left

	optl1	±dxl1	±fnl1(i)
23.	1	1	0 82
24.	0		

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
23.	0		
24.	1	1	940 111

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
23.	—		
24.	—		

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
23.	1	0	3.9	30	
24.	1	0	3.3	-30	

Table 32.--Input form with input values for two examples used to calculate horizon 4 in Table 32 as described in text.....

```
seismc

type input file name: tester

input complete from 3 dsk files named     tester
type subroutine: crit
(23) crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*1 com lin 3 210 550 3.9 30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*1 1 0 82;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*0;
good news: the critical points have been found.they are
critical subsurface coordinates xicrt=    39.32 etacrt=   -71.63
critical subsurface arrival time taucrt=    48.25
critical distance on surface xcrt=    68.26
critical surface arrival time tcrt=    73.35
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.9 30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
            98.26                 69.10       -71.10
           128.26                 98.89       -70.57
           158.26                129.29       -69.93
           188.26                160.30       -68.91
           218.26                192.97       -67.11
           248.26                224.07       -64.81
           278.26                256.54       -61.92
           308.26                289.39       -58.20
           338.26                321.30       -53.89
           368.26                349.72       -50.77
           398.26                374.04       -50.82
           428.26                390.01       -53.44
           458.26                410.66       -58.86
           488.26                435.85       -65.34
           518.26                459.26       -70.91
do you want the star points printed?no
(24) do you wish to try anothor shot? yes or no:yes
crit entry 1   file=tester    ; k*,ext,curv,id,xmin,xmax,v2,+or-delx,delxl,dint,
 <nval,bend1,bend2,nprin,nstar,delll>
*2 com lin 3 350 670 3.3 -30 20 20 4;
crit entry 1a; optll*,+or-dxll,+or-fnlr(i)
*0;
crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)
*1 1 940 111;
good news: the critical points hsve been found.they are
critical subsurface coordinates xicrt=   854.31 etacrt=   -59.71
critical subsurface arrival time taucrt=    76.34
critical distance on surface xcrt=   805.45
critical surface arrival time tcrt=   110.30
do you want to try for another solution? yes or no:no
try across for solutions along t-x graph? yes or no:yes
across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>
*1 0 3.3 -30;
      the answers are:
      coordinate id horizon   coordinates of id+1 horizon
            xr(i)                xir(i)      etar(i)
           775.45                824.18       -60.41
           745.45                793.92       -61.08
           715.45                766.11       -62.01
           685.45                738.41       -63.42
           655.45                708.94       -65.21
           625.45                676.44       -66.47
           595.45                639.34       -66.31
           565.45                605.74       -64.82
           535.45                573.86       -63.09
           505.45                544.10       -61.89
           475.45                519.64       -62.03
           445.45                493.90       -63.64
           415.45                480.02       -65.75
           385.45                456.10       -71.23
do you want the star points printed?no
do you wish to try anothor shot? yes or no:QUIT
```

Table 33.--Execution of subroutines CRIT and ACROSS for the two examples listed in Table 32.....

The final configuration of horizon 4 is shown in figure 48c. This configuration is input into file testcr.org and the printout is shown in tables 34a and b.

In order to test the validity of this result, arrived at perhaps through devious means, subroutine RAYTRACE is invoked for both shotpoints S_1 and S_2. Execution is shown in table 35 and the results are plotted with the symbol x which are compared with the points representing the original data in figure 49. Except for the results near distance coordinates 350 to 400 for S_2, the results of raytrace agree fairly well with the original arrival-times from horizon 4 and thus, horizon 4 as shown in figure 48c is approximately correct mathematically. It must be pointed out, however, that a great many solutions can be found which fit the data equally well. For instance, extension of the data from the two shotpoints to reciprocal tie points can be done in an infinite number of ways, and by invoking subroutine RECIP (to be discussed under the next subheading) any number of solutions for horizon 4 can be found. The only way to overcome such problems is to collect adequate field data so that reciprocal times and intercept times can be pinned down for all the important refracting horizons. It is never feasible to collect complete data from all horizons, notably the shallower ones and hence subroutine CRIT must be invoked to determine depths for these. As we have gained experience with these programs however, we tend to lean less heavily on subroutine ACROSS but instead extend data, when necessary, in both directions and use subroutine RECIP.

For the reasons just discussed concerning horizon 4, it would be ill-advised to begin solving for horizon 5 from the data given in figure 47 using subroutines CRIT and ACROSS. We will return to this horizon, however, during the discussion of subroutine RECIP.

Subroutine RECIP (nisect,xi,eta,vinst,v,tau,delt,dell,xminus,xplus)

Given a velocity model in filnam.lay consisting of id layers, and reversed (left and right) t-x data from horizon (id+1) in either filnam.org and/or filnam.com, subroutine RECIP calculates elevations, velocities and other parameters of the (id+1)th horizon. These parameters are nisect, the number of depth points calculated, (xi(i),eta(i)), the distance and depth coordinates of these depth points where (i=1,nisect), vinst(i), the velocity of the (id+1)th horizon at the points (xi(i),eta(i)) assuming refraction at the critical angle; v(i), the interval refraction velocity between the (i-1)th and ith depth point; tau(i), the travel time from the left shotpoint to the ith depth point; delt(i), the travel time in the interval between the (i-1)th and ith depth point; dell(i), the distance between these successive depth points; and xminus(i) and xplus(i), the distance coordinates from the right and left shotpoints respectively at which emerging rays from the point (xi(i),eta(i)) intersect the idth horizon.

A basic assumption for program execution is that the reciprocal time is known (e.g., the travel time between the left and right shotpoints in the (id+1)th layer). But because the reciprocal time is not always available from the t-x data, the program provides the option to extend data according

```
seismc
(a) type input file name: testcr

input complete from 3 dsk files named     testcr
type subroutine: datain
type file extension either org,com,or lay: lay
datain entry 2; i*,nc(i):4 12;
cx(i,j):-100 39 160 224 374 460 520 544 639 766 854 1000;
cy(i,j):-72 -72 -69 -65 -51 -71 -62 -62 -66 -62 -60 -58;
cv(i,j):3.9 3.9 3.9 3.9 3.9 3.9 3.3 3.3 3.3 3.3 3.3 3.3;
next layer please.  type 0; or -n; for return to mainline
datain entry 2; i*,nc(i):0;
output of 3 files to dsk complete
datain complete
type subroutine: term
(b) type input filename and extension: testcr.lay

                    testcr.lay


1  6  -90000.00        0.00        0.65
      0.00        0.00        0.65
    330.00       16.00        0.60
    660.00       32.00        0.55
   1000.00       50.00        0.55
  90000.00       50.00        0.55
2  4  -90000.00      -10.00        1.50
      0.00      -10.00        1.50
   1000.00       39.00        1.50
  90000.00       39.00        1.50
3  4  -90000.00      -28.00        2.15
      0.00      -28.00        2.15
   1000.00        0.00        2.15
  90000.00        0.00        2.15
4 12    -100.00      -72.00        3.90
     39.00      -72.00        3.90
    160.00      -69.00        3.90
    224.00      -65.00        3.90
    374.00      -51.00        3.90
    460.00      -71.00        3.90
    520.00      -62.00        3.30
    544.00      -62.00        3.30
    639.00      -66.00        3.30
    766.00      -62.00        3.30
    854.00      -60.00        3.30
   1000.00      -58.00        3.30


type subroutine: QUIT
```

Table 34.--(a) Execution of subroutine DATAIN to input the interpreted velocity section of horizon 4 of figure 48c into file testcr.lay.
(b) Printout of file testcr.lay showing four horizons.....

```
seismo

type input file name: tester

input complete from 3 dsk files named     tester
type subroutine: raytr
raytrace entry 1                    file=    tester
layd#,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
#4 0 3 0 800 30 no ;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
#0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time       refrac. angle
     4          37.80        -72.00         48.39         -82.12

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=    3.00
  from off horizon    4
 x-refractor coord.    x-surface coord.   arrival time
          37.80              77.51          103.81
          67.80             105.75          111.44
          97.80             136.06          119.62
         127.80             166.37          127.80
         157.80             196.68          135.99
         187.80             223.20          142.77
         217.80             252.82          150.34
         247.80             279.65          156.92
         277.80             308.80          164.01
         307.80             337.81          172.34
         337.80             366.96          179.43
         367.80             396.10          186.51
         397.80             455.59          205.54
         427.80             494.75          219.28
         457.80             533.75          233.01
         487.80             522.01          227.78
         517.80             550.47          234.05
         547.80             612.11          252.58
         577.80             644.97          263.65
         607.80             677.42          276.23
         637.80             709.95          287.48
         667.80             731.00          294.17
         697.80             761.24          303.89
         727.80             791.47          313.52
         757.80             821.70          323.34
         787.80             853.19          333.54
all done. another shot!
raytrace entry 1                    file=    tester
layd#,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
#4 920 3 920 0 30 no ;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
#0;
```

```
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time       refrac. angle
     4          854.96        -59.99         76.24         79.00

do you want to continue? yes or no:yes
arrival time data for shot at x=     920.00 depth=    3.00
  from off horizon    4
 x-refractor coord.    x-surface coord.   arrival time
         854.96             786.45          154.53
         824.96             755.85          163.15
         794.96             726.35          171.47
         764.96             695.79          180.09
         734.96             666.05          188.57
         704.96             636.15          195.80
         674.96             606.39          204.52
         644.96             576.63          213.23
         614.96             556.69          218.58
         584.96             528.59          226.05
         554.96             500.50          233.52
         524.96             467.66          242.57
         494.96             433.88          252.23
         464.96             400.79          261.75
         434.96             408.69          258.55
         404.96             381.70          262.24
         374.96             354.72          265.94
         344.96             304.47          276.53
         314.96             272.74          284.90
         284.96             241.23          293.27
         254.96             209.84          301.64
         224.96             178.50          310.02
         194.96             150.34          317.26
         164.96             119.84          325.10
         134.96              93.06          331.66
         104.96              63.39          338.88
          74.96              33.72          346.09
          44.96               4.06          353.31
          14.96             -23.30          360.42
all done. another shot!
raytrace entry 1                    file=    tester
layd#,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
#0;
type subroutine: QUIT
```

Table 35.--Execution of subroutine RAYTRACE to check accuracy of results for horizon 4 shown in figure 48c.....

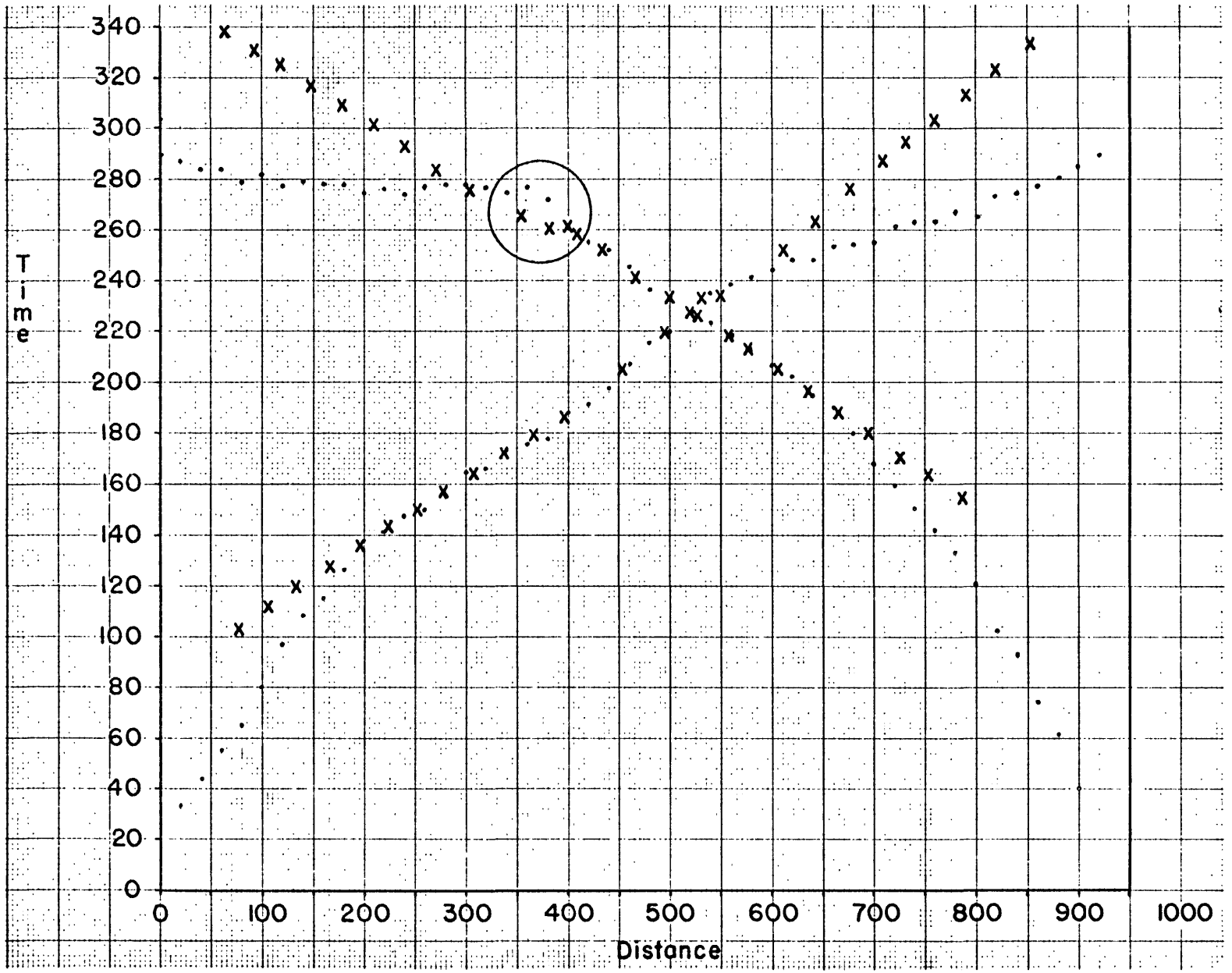

Figure 49.--Plot of the results of subroutine RAYTRACE (Table 35) plotted with the symbol x as compared to the t-x data of figure 47 from which the velocity model was derived.....

to several criteria to intersect the reversed shotpoints. The methods used for extending data are identical to those described in subroutine CRIT and hence are not repeated here. In lieu of extending data to obtain the reciprocal time, the option is also available to simply input this value. In this case, uphole times must also be input, whereas they are otherwise calculated within the program. Artificial data may also be used.

Subroutine RECIP calls subroutines SELECT, COMP, XTEND, INTERP, WAVED, EQSPACE, either SPLIN1 and SPLINDT or LINDT, RATHRU and RANGE.

Several examples of the execution of RECIP follow for which the program entry list, the input form and the condition statements are included in Appendix A.

Figure 50a shows reversed t-x plots for shotpoints S_1 and S_2. For S_1 there are five data points and for S_2 four; each are plotted with the symbol x. These t-x values are input into file tcip1.org using subroutine DATAIN and listed in table 36a. Shotpoint S_1 is at distance coordinate 0 and S_2 at 8000 with shots at zero depth. These plots which define straight lines could of course be solved quite easily without the aid of a computer. They will first be solved as a two layer problem, with the first horizon horizontal at zero elevation with velocity 2 (fig. 50b). The values for this uppermost layer are input into file tcip1.lay using subroutine DATAIN and are listed in table 36b.

The entry values for this problem are posted in table 37(1). Under entry 1, the deepest known horizon is id=1, the lin approximation is used, and the calculation interval delx, is 500. (In addition to choosing delx small enough to include all significant perturbations of the t-x graph, it should also be less than the thickness of the layer being calculated to avoid missing points which could otherwise be calculated. The use of too large a value for delx is becomes evident only after calculations are made and a change in the value of delx involves repeating all the calculations. The default options are used for the remaining values of entry 1. Under entry 2, the shot index kl for the left shotpoint is 1, and its data is in the file with extension org. The values -1000 and 10,000 for xminl and xmaxl insures that all the data points associated with the left shotpoint are included in the calculations. The value 100 is arbitrarily selected for delxl and dintl. We usually select them to be less than delx, but for straight line curves as shown in this example, they could be considerably larger without affecting the results. The default option is used for nvall, nprinl and nstarl. Entries 2a and 2b are zero because the curves for the left shotpoint need not be extended; because the t-x points clearly include the distance coordinate of the reversed shotpoint. Because artificial data are not used (xminl≠xmaxl and xminr≠xmaxr) entries 2c and 3c are not used. Execution of subroutine RECIP is shown in table 38 as four steps.. After the entry values are input (step 1), the program prints out the calculated reciprocal times from both shotpoints for inspection (step 2). Because the shotpoint is on the surface, these are identical to the reciprocal times of the reversed curves in figure 50a. Should they not agree within the accuracy of the data, further calculation may be halted. The option is also provided at this step to print out the reduced t-x data points. Usually this option is declined, unless it is necessary to inspect an arrival-time curve reduced to the idth horizon.

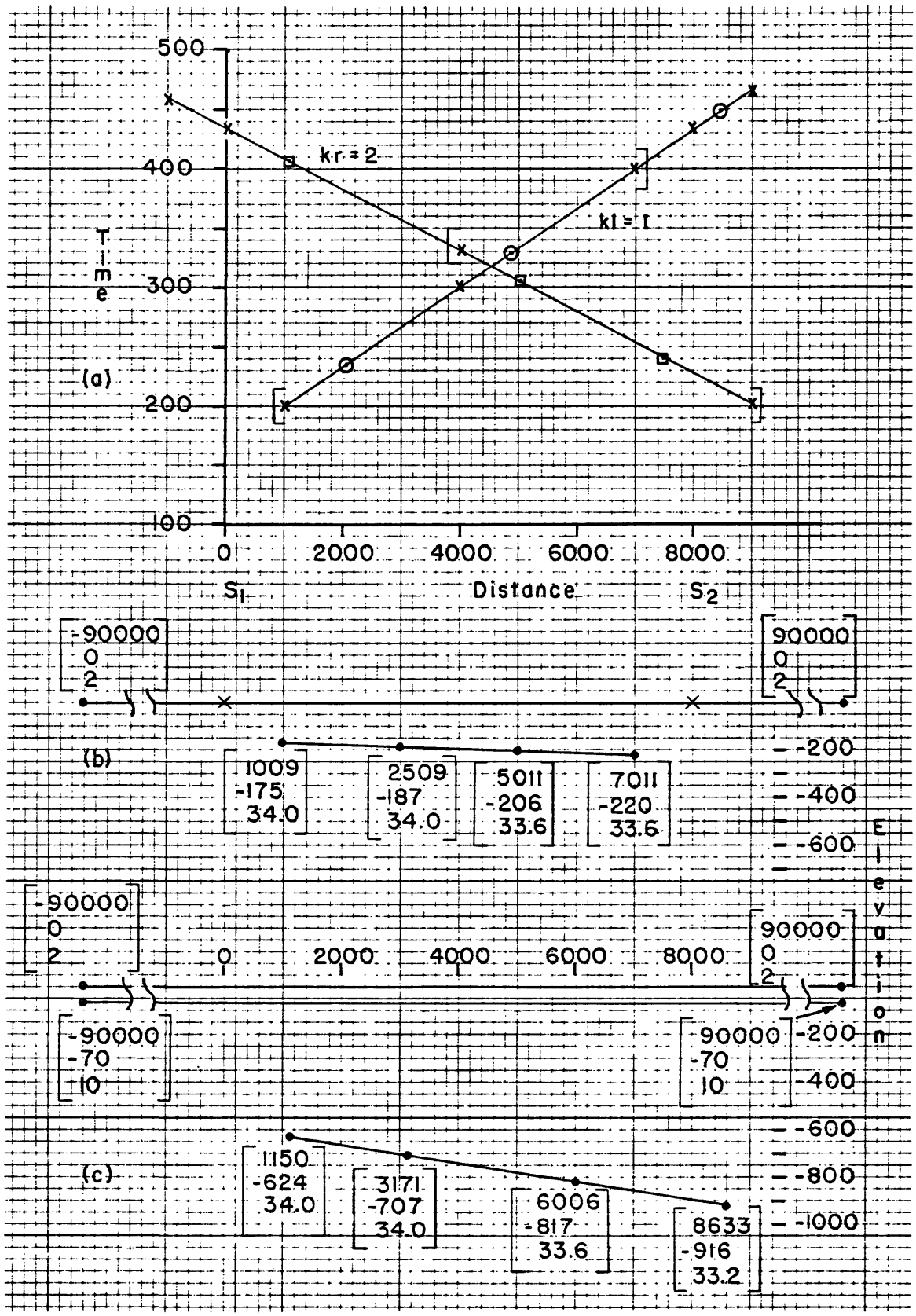

Figure 50.--First example used to illustrate execution of subroutine RECIP as described in text.....

(a) print tcip1.org

tcip1.org

```
1  5        0.00          0.00        -1.00
  1000.00      200.00
  4000.00      300.00
  7000.00      400.00
  8000.00      433.00
  9000.00      467.00
2  4     8000.00          0.00        -1.00
 -1000.00      458.00
     0.00      433.00
  4000.00      331.00
  9000.00      200.00
```

(b) print tcip1.lay

tcip1.lay

```
1  2  -90000.00          0.00         2.00
 90000.00         0.00         2.00
```

Table 36.--(a) Printout of file tcip1.org showing the t-x data points of Figure 50(a).
(b) Printout of file tcip1.lay showing horizon 1 of Figures 50(b) and (c).....

Table 37.--Input form with input values for three examples used to interpret the reversed arrival-time curves shown in Figure 50(a).....

RECIP input form — File name tcip1

(optional parameters in parentheses)

Entry 1

	id	curv lin/spl	delx	(bend1 yes/no	bend2 yes/no	iu	mycip yes/no	tss	tuh1	tuhr)
1.	1	lin	500							
2.	2	lin	200							
3.	2	lin	200	no	no	1	yes	433		

Entry 2 (left shotpoint)

	kl	extl org/com	xminl	xmaxl	delxl	dintl	(nval1	nprin1 0/1	nstarl)
1.	1	org	-1000	10000	100	100			
2.	1	org	-1000	10000	100	100			
3.	1	org	500	7500	100	100			

Entry 2a extend data to left

	optl1	±dxl1	±fnl1(i)
1.	0		
2.	0		
3.	0		

Entry 2b extend data to right

	optlr	±dxlr	±fnlr(i)
1.	0		
2.	0		
3.	0		

Entry 3 (right shotpoint)

	kr	extr org/com	xminr	xmaxr	delxr	dintr	(nvalr	nprinr 0/1	nstarr)
1.	2	org	-2000	10000	100	100			
2.	2	org	-2000	10000	100	100			
3.	2	org	3000	10000	100	100			

Entry 3a extend data to left

	optr1	±dxrl	±fnrl(i)
1.	0		
2.	0		
3.	0		

Entry 3b extend data to right

	optrr	±dxrr	±fnrr(i)
1.	0		
2.	0		
3.	0		

Entry 2c (optional artificial; if xmin = xmaxr)

	nl	x1(i) i = 1, nl	t1(i) i = 1, nl
1.			
2.			
3.			

Entry 3c (optional artificial; if xminr = xmaxr

	nr	xr(i) i = 1, nv	tr(i) i = 1, nv
1.			
2.			
3.			

```
seismc

type input file name: tcip1

1.input complete from 3 dsk files named     tcip1
  type subroutine: recip
  recip entry 1                      file=    tcip1
  id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
  *1 lin 500;
  recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
  *1 org -1000 10000 100 100;
  recip entry 2a; optll*,+or-dxll,+or-fnll(i)
  *0;
  recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
  *0;
  recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
  *2 org -2000 10000 100 100;
  recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
  *0;
  recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
  *0;
2.recip times adding uphole;  left to right, right to left, and average
       433.0       433.0       433.0
  do you want to continue? yes or no:yes
  print t-x data on horizon    1 ? yes or no:no
3.The reciprocal time solutions are as follows
              point values          incremental parameters     surface coordinates
        xi        eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                                    right sp left sp
    1008.9     -174.9     34.0      0.0   113.05     0.00     0.00   1000.0   1020.6
    1509.1     -178.8     34.0     34.0   127.77    14.72   500.22   1500.0   1521.1
    2009.3     -182.7     34.0     34.0   142.48    14.72   500.22   2000.0   2021.5
    2509.5     -186.7     34.0     34.0   157.20    14.72   500.22   2500.0   2522.0
    3009.7     -190.6     34.0     34.0   171.91    14.72   500.22   3000.0   3022.5
    3509.9     -194.5     34.0     34.0   186.63    14.72   500.22   3500.0   3522.9
    4010.3     -198.4     33.8     34.0   201.36    14.73   500.35   4000.0   4023.5
    4510.6     -202.0     33.6     33.6   216.24    14.88   500.34   4500.0   4524.1
    5010.8     -205.6     33.6     33.6   231.13    14.89   500.20   5000.0   5024.5
    5511.0     -209.2     33.6     33.6   246.02    14.89   500.20   5500.0   5525.0
    6011.2     -212.7     33.6     33.6   260.91    14.89   500.20   6000.0   6025.4
    6511.4     -216.3     33.6     33.6   275.80    14.89   500.20   6500.0   6525.7
    7011.5     -219.9     33.7     33.6   290.68    14.88   500.20   7000.0   7026.1
    7511.7     -223.3     33.8     33.8   305.49    14.81   500.19   7500.0   7526.7
    8011.9     -226.7     33.5     33.7   320.33    14.83   500.19   8000.0   8027.4
    8512.1     -230.6     33.2     33.3   335.37    15.04   500.22   8500.0   8527.8
4.for the right shotpoint, no solutions occur along the
  reduced t-x graph in the intervals:
    -1000.0 to    1000.0  and    8500.0 to    9000.0
  and for the left shotpoint, in the intervals:
     1000.0 to    1020.6  and    8527.8 to    9000.0
  do you wish the xstar points? yes or no
  no
  type subroutine: QUIT
```

Table 38.--Execution of subroutine RECIP for the example in Figure 50(a) assuming a two-layer problem.....

```
seismc
```

(a)
```
type input file name: tcip1

input complete from 3 dsk files named     tcip1
type subroutine: datain
type file extension either org,com,or lay: lay
datain entry 2; i*,nc(i):2 6;
cx(i,j):-491 1009 2509 5011 7011 9011;
cy(i,j):-163 -175 -187 -206 -220 -234;
cv(i,j):34 34 34 33.6 33.6 33.6;
next layer please.  type 0; or -n; for return to mainline
datain entry 2; i*,nc(i):0;
output of 3 files to dsk complete
datain complete
type subroutine: term
type input filename and extension: tcip1.lay

                    tcip1.lay


  1  2  -90000.00        0.00        2.00
   90000.00        0.00        2.00
  2  6     -491.00     -163.00       34.00
    1009.00     -175.00       34.00
    2509.00     -187.00       34.00
    5011.00     -206.00       33.60
    7011.00     -220.00       33.60
    9011.00     -234.00       33.60
```

(b)
```
type subroutine: raytr
raytrace entry 1                        file=    tcip1
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*2 0 0 0 9000 400;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
     2          8.46       -167.00         83.60        -84.79

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=   0.00
  from off horizon   2
 x-refractor coord.    x-surface coord.  arrival time
          8.46              19.64        167.29
        408.46             419.85        180.66
        808.46             820.07        194.03
       1208.46            1220.28        207.40
       1608.46            1620.50        220.76
       2008.46            2020.71        234.13
       2408.46            2420.93        247.50
       2808.46            2821.06        260.81
       3208.46            3221.26        274.09
       3608.46            3621.46        287.38
       4008.46            4021.66        300.67
       4408.46            4421.86        313.96
       4808.46            4822.07        327.24
       5208.46            5222.28        340.54
       5608.46            5622.47        353.85
       6008.46            6022.66        367.16
       6408.46            6422.84        380.47
       6808.46            6823.03        393.77
       7208.46            7223.22        407.08
       7608.46            7623.40        420.39
       8008.46            8023.59        433.70
       8408.46            8423.78        447.01
       8808.46            8823.96        460.31
all done. another shot!
raytrace entry 1                        file=    tcip1
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*0;
type subroutine: QUIT
```

Table 39.--(a) Execution of subroutine DATAIN to input the interpreted velocity section of horizon 2 into file tcip1.lay and subsequent printout.
(b) Execution of subroutine RAYTRACE for horizon 2.....

The answers are printed out in step 3. "Point values" are the coordinates (xi,eta) of the successively calculated depth points on the (id+1)th horizon, and the refraction velocity at these points assuming critical refraction. "Incremental parameters" are calculated using successive depth points. The velocity between successive depth points is obtained by dividing the distance delta-x by the time interval delta-t. This value is generally approximately equal to the velocity listed under "point values" and, from a personal point of view, is usually preferred. The value of the first incremental velocity is arbitrarily set equal to zero. The delta-t and delta-x are printed because this permits the user to analyze velocities over larger intervals. The value tau, seldom used, represents the travel-time from the left shotpoint to the point (xi,eta). The values of x-minus and x-plus are used for determining which points on the original t-x graphs reduce to the depth point (xi,eta). For example, the first line of the printout shows that the rays emerging from the point (1008.9,-174.9) produce an arrival for the reduced right shotpoint at distance coordinates 1000.0 and for the reduced left shotpoint at distance coordinate 1020.6. Unfortunately, these distance coordinates refer to the reduced t-x data. To relate them to the t-x data actually recorded on the surface (horizon 1) requires use of the "xstar" points.

Under step 4, the printout indicates which intervals of the reduced t-x graph did not produce solutions because of the absence of corresponding points on the reversed t-x graph. These intervals occur near the end of the data list. For example, for the right shotpoint, the program did not compute solutions for arrivals between -1000 to 1000 and between 9500 and 9000; for the left shotpoint, solutions for arrivals in the intervals 1000 to 1020.6 and 8527.8 and 9000 were not computed. Usually these values indicate intervals for which reversed t-x data is simply not available. However, in this case, some of the intervals could have been reduced by using a smaller value of delx as described earlier, reducing the probability of missing end points.

Several values of computed (xi,eta) have been plotted on the velocity section of figure 50b.

In order to check the result, values for horizon 2 are input into file tcipl.lay using subroutine DATAIN (table 39a). Note that this horizon has been extended beyond the calculation interval of 1008.9 to 8512.1 (column 1 in table 38b) to include depths below each shotpoint. Subroutine RAYTRACE is then executed for shotpoint S_1 (table 39b). Some of the solutions have been plotted with the symbol o in the initial t-x graph of figure 50a. Exact agreement is expected, especially for a simple problem such as this one. However, detailed comparison shows slight discrepancies. The reason for the discrepancies is not that depth calculations in subroutine RECIP are wrong, but that extension of the refracting horizon below the shotpoint introduced slight errors. The same results would have accrued if RAYTRACE had been executed for shotpoint S_2.

The same arrival-time curve may be computed assuming a three-layer problem. (Note that because we have not presented arrival-times for shallower layers, we may within limits, assume any overlying configuration we wish.) The assumed velocity distribution is input into file tcipl.lay shown in table 40a and sketched in figure 50c; e.g., two horizontal layers

the upper with a velocity of 2 and the lower 70 units deep with velocity of 10 were input. Input values are given in table 37(2). The method of entry is identical to that described in the first problem except that id=2 and delx was decreased from 500 to 200. (The problem was first run with delx=500, but only two depth points, shown in table 40b resulted.) Execution of subroutine RECIP is shown in table 41. Some of the answers are plotted in the velocity section for figure 50c. Table 42a shows this horizon input into file tcip1.lay, and table 42b shows the execution of RAYTRACE for shotpoint S_2 to check the results. These checks are plotted with the square symbol in figure 50a.

Tables 37(3) and 43 illustrate a third execution of RECIP using the above 3 layer example but for the case in which the data do not include the reciprocal point (e.g., mycip="yes"). T-x values used in this example are shown in brackets in figure 50a. The answers in table 43 are the same as those in 41c except that the extent of the answers is reduced due to fewer t-x data values.

A second example of subroutine RECIP is the solution of the t-x plot of figure 51. This graph is the same as that for horizons 4 and 5 of the example used to illustrate subroutine CRIT. The file name has been changed from testcr to tcip2 and the t-x data file used for computation is given in tcip2.com shown in table 44a. The solutions obtained for the first three layers from subroutine CRIT describe the overlying velocity distribution; they are sketched in figure 52 and shown in table 44b in file tcip2.lay.

The t-x data points representing horizon 4 are shown as dots within the brackets in figure 51. In order to find a solution which fits these points using reciprocal methods, it is necessary to find a reciprocal time. Because this time was not recorded and constraining complementary data are not available, the data must be extended to intersect the reciprocal time axis, (-50,377) and (975,383) in figure 51. The data are also extended to intersect the shotpoints at (0,88) and (920,118). Because data extension is not unique, an infinite number of solutions exist for the bracketed points. The entry values for this example are given in table 45(1) and program execution is shown in table 46. The computed horizon 4 is plotted in figure 52. The special "star" points are also printed out in table 46, and their use for helping reconstruct ray paths is illustrated as follows. The depth point with coordinates (444.6, -57.7) is circled in table 46. From the table we see that emerging rays from this point intersect horizon 3 at 398.7 for the right shotpoint and 489.8 for the left shotpoint. These intersection points are shown by the emerging rays sketched from horizon 4 to horizon 3 in figure 52. From the "star" points in table 46, rays from these intersections are seen to emerge at the surface at approximately 381 and 508, which are also sketched on horizon 1 in figure 52. To relate these values to the t-x graph, these points are shown as the heavy circles in figure 51. These two points on the arrival-time curve then are those which combined to give the subsurface point (444.6,-57.7). This procedure is a bit laborious, but at present is the only method for relating t-x data points with subsurface depth points.

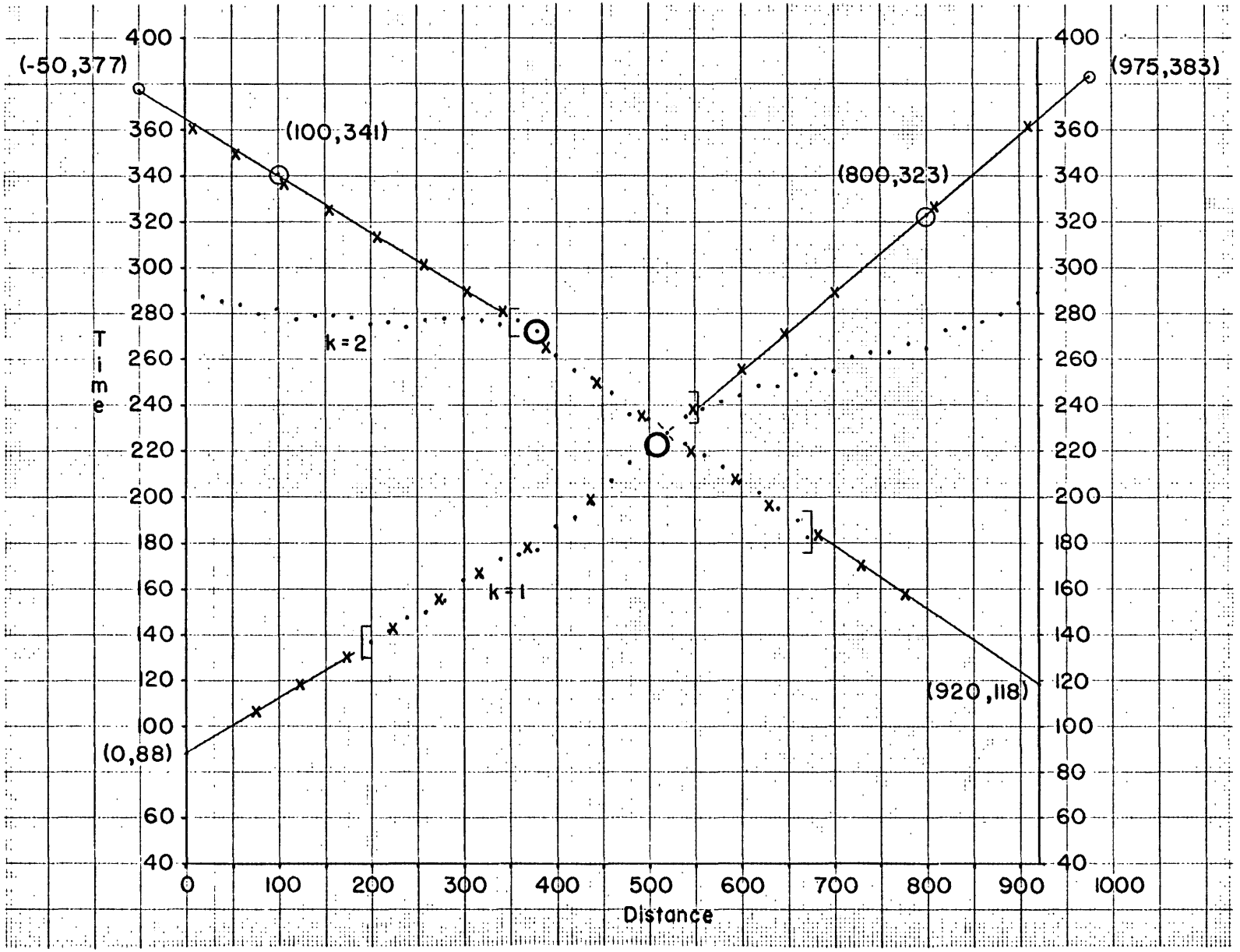

Figure 51.--Second example used to illustrate execution of subroutine RECIP as described in text.....

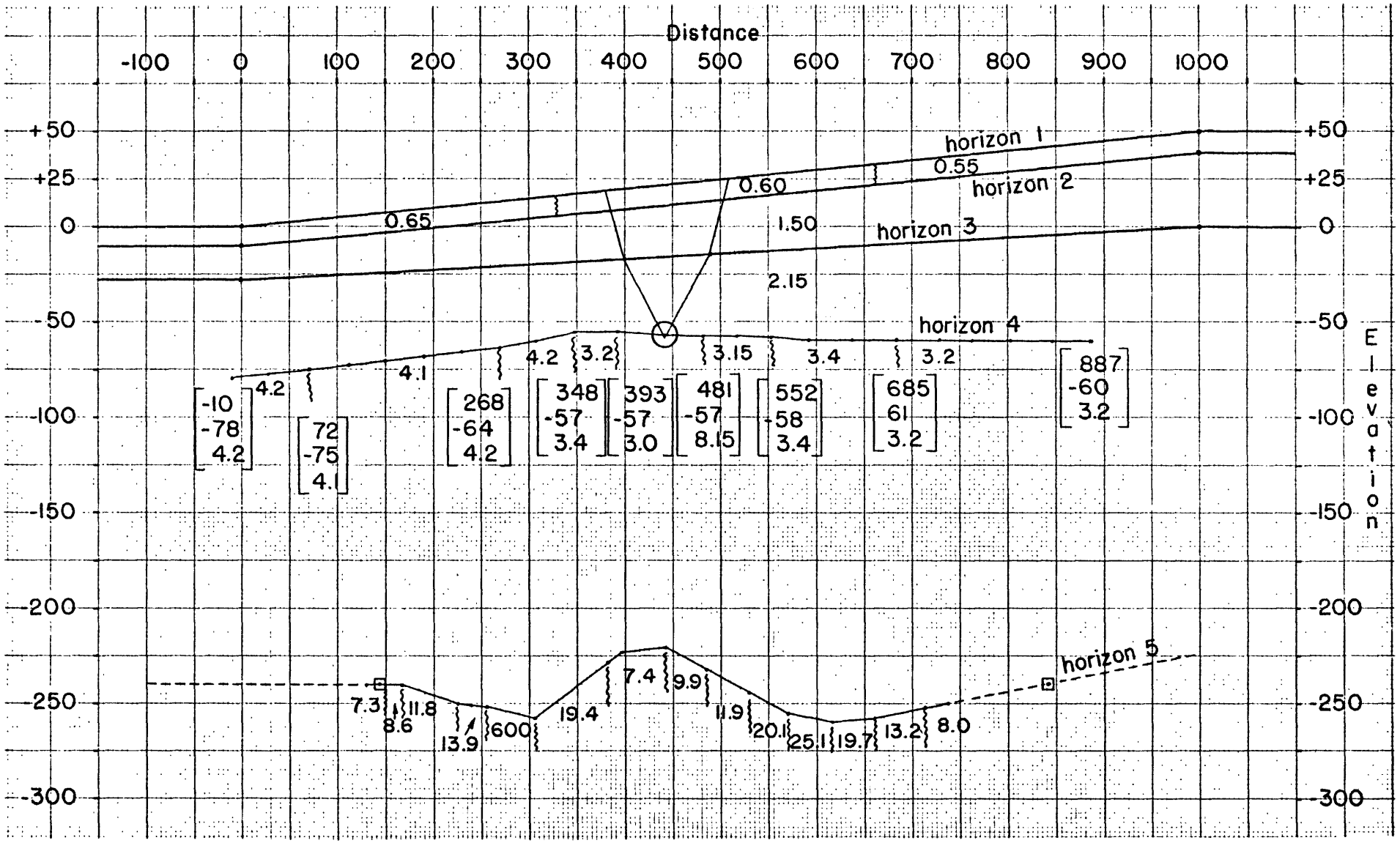

Figure 52.--Velocity model interpreted for hozisons four and five from subroutine RECIP from the arrival-time plots of Figure 51.....

Table 40.--(a) Printout of file tcip1.lay showing horizons 1 and 2 of Figure 50(c).
(b) First execution of subroutine RECIP for the example in Figure 50(a) assuming a three-layer problem.....

```
seismc
(a) type input file name: tcip1

 input complete from 3 dsk files named     tcip1
 type subroutine: term
 type input filename and extension: tcip1.lay

                    tcip1.lay


  1  2  -90000.00        0.00        2.00
   90000.00        0.00        2.00
  2  2  -90000.00      -70.00       10.00
   90000.00      -70.00       10.00


(b) type subroutine: recip
 recip entry 1                          file=    tcip1
 id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
 *2 lin 500;
 recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
 *1 org -1000 10000 100 100;
 recip entry 2a; optll*,+or-dxll,+or-fnll(i)
 *0;
 recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
 *0;
 recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
 *2 org -2000 10000 100 100;
 recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
 *0;
 recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
 *0;
 recip times adding uphole;  left to right, right to left, and average
      433.0       433.0       433.0
 do you want to continue? yes or no:yes
 print t-x data on horizon    2 ? yes or no:no
 The reciprocal time solutions are as follows
              point values          incremental parameters     surface coordinates
        xi        eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                                   right sp left sp
    7723.2    -879.3    33.5      0.0  312.60     0.00      0.00   7503.5   8010.9
    8228.8    -899.8    33.2     33.2  327.82    15.22    505.95   8003.5   8528.8
 for the right shotpoint, no solutions occur along the
 reduced t-x graph in the intervals:
    -996.5 to    7503.5  and    8003.5 to    9003.7
 and for the left shotpoint, in the intervals:
     995.3 to    8010.9  and    8528.8 to    8995.2
 do you wish the xstar points? yes or no
 no
 type subroutine: QUIT
```

Table 41.--Second execution of subroutine RECIP for the example in Figure 50(a) assuming a three-layer problem.....

```
seismo

type input file name: tcip1

input complete from 3 dsk files named     tcip1
type subroutine: recip
recip entry 1                    file=    tcip1
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*2 lin 200;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 org -1000 10000 100 100;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 org -2000 10000 100 100;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
recip times adding uphole;  left to right, right to left, and average
     433.0      433.0      433.0
do you want to continue? yes or no:yes
print t-x data on horizon    2 ? yes or no:no
The reciprocal time solutions are as follows
             point values          incremental parameters     surface coordinates
       xi        eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                                   right sp left sp
   1149.6    -624.0     34.0      0.0  117.84     0.00      0.00   1003.5   1345.5
   1351.8    -632.3     34.0     34.0  123.79     5.96    202.36   1203.5   1550.6
   1554.0    -640.6     34.0     34.0  129.75     5.96    202.36   1403.5   1755.7
   1756.2    -648.9     34.0     34.0  135.71     5.96    202.35   1603.5   1960.8
   1958.3    -657.1     34.0     34.0  141.67     5.96    202.36   1803.5   2165.9
   2160.5    -665.4     34.0     34.0  147.62     5.96    202.36   2003.5   2371.1
   2362.7    -673.7     34.0     34.0  153.58     5.96    202.36   2203.5   2576.2
   2564.9    -682.0     34.0     34.0  159.54     5.96    202.37   2403.5   2781.3
   2767.1    -690.3     34.0     34.0  165.50     5.96    202.36   2603.5   2986.4
   2969.3    -698.6     34.0     34.0  171.45     5.96    202.34   2803.5   3191.5
   3171.5    -706.9     34.0     34.0  177.41     5.96    202.38   3003.5   3396.7
   3373.7    -715.2     34.0     34.0  183.37     5.96    202.34   3203.5   3601.8
   3575.9    -723.5     34.0     34.0  189.33     5.96    202.36   3403.5   3806.9
   3778.0    -731.8     34.0     34.0  195.29     5.96    202.35   3603.5   4012.0
   3980.2    -740.1     34.0     34.0  201.24     5.96    202.37   3803.5   4217.1
   4185.0    -748.3     33.8     33.9  207.29     6.05    204.98   4003.5   4424.9
   4389.8    -756.1     33.6     33.7  213.38     6.09    204.89   4203.5   4632.4
   4591.8    -763.7     33.6     33.6  219.40     6.02    202.20   4403.5   4837.1
   4793.9    -771.2     33.6     33.6  225.43     6.02    202.17   4603.5   5041.8
   4995.9    -778.8     33.6     33.6  231.45     6.02    202.20   4803.5   5246.5
   5198.0    -786.4     33.6     33.6  237.47     6.02    202.18   5003.5   5451.2
   5400.0    -793.9     33.6     33.6  243.49     6.02    202.21   5203.5   5656.0
   5602.1    -801.5     33.6     33.6  249.52     6.02    202.19   5403.5   5860.7
   5804.1    -809.0     33.6     33.6  255.54     6.02    202.18   5603.5   6065.4
   6006.2    -816.6     33.6     33.6  261.56     6.02    202.20   5803.5   6270.1
   6208.2    -824.1     33.6     33.6  267.59     6.02    202.20   6003.5   6474.9
   6410.3    -831.7     33.6     33.6  273.61     6.02    202.22   6203.5   6679.6
   6612.3    -839.2     33.6     33.6  279.63     6.02    202.14   6403.5   6883.5
   6814.3    -846.6     33.7     33.7  285.63     6.01    202.16   6603.5   7086.7
   7016.3    -853.8     33.8     33.7  291.62     5.99    202.06   6803.5   7290.3
   7218.3    -861.0     33.8     33.8  297.61     5.99    202.13   7003.5   7494.8
   7420.2    -868.2     33.8     33.8  303.59     5.98    202.03   7203.5   7699.2
   7622.2    -875.5     33.8     33.7  309.59     6.00    202.15   7403.5   7906.7
   7824.3    -883.3     33.5     33.5  315.64     6.05    202.27   7603.5   8116.6
   8026.5    -891.5     33.2     33.2  321.73     6.09    202.37   7803.5   8323.4
   8228.8    -899.8     33.2     33.2  327.82     6.10    202.43   8003.5   8528.8
   8431.0    -908.1     33.2     33.2  333.92     6.10    202.43   8203.5   8733.9
   8633.3    -916.3     33.2     33.2  340.02     6.10    202.41   8403.5   8939.2
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
   -996.5 to    1003.5  and    8403.5 to    9003.7
and for the left shotpoint, in the intervals:
    995.3 to    1345.5  and    8939.2 to    8995.2.
do you wish the xstar points? yes or no
no
type subroutine: QUIT
```

Table 42.--(a) Printout of file tcip2.lay showing the three horizons of Figure 50(c).
(b) Execution of subroutine RAYTRACE for horizon 3 of figure 50c.....

```
seismc
(a) type input file name: tcip1
input complete from 3 dsk files named     tcip1
type subroutine: term
type input filename and extension: tcip1.lay
                    tcip.lay
  1  2  -90000.00        0.00        2.00
   90000.00        0.00        2.00
  2  2  -90000.00      -70.00       10.00
   90000.00      -70.00       10.00
  3  6  -90000.00     -535.00       34.00
   -1000.00     -535.00       34.00
     947.00     -615.00       34.00
    4491.00     -760.00       33.60
    8633.00     -916.00       33.20
   90000.00     -916.00       33.20
(b) type subroutine: raytr
raytrace entry 1                        file=    tcip1
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*3 8000 0 8000 0 500;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon        x-coord.     elevation       time      refrac. angle
      3       7710.43       -881.25        121.06         83.36

do you want to continue? yes or no:yes
arrival time data for shot at x=   8000.00 depth=   0.00
  from off horizon   3
 x-refractor coord.    x-surface coord.  arrival time
        7710.42             7487.00        240.16
        7210.42             6992.10        253.10
        6710.42             6497.20        266.04
        6210.42             6002.30        278.98
        5710.42             5507.40        291.92
        5210.42             5012.50        304.86
        4710.42             4517.61        317.80
        4210.42             4028.07        330.42
        3710.42             3533.46        343.02
        3210.42             3038.85        355.62
        2710.42             2544.24        368.23
        2210.42             2049.63        380.83
        1710.42             1555.02        393.43
        1210.42             1060.41        406.03
         710.42              565.92        418.63
         210.42               71.32        431.22
all done. another shot!
raytrace entry 1                        file=    tcip1
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*0;
type subroutine: QUIT
```

Table 43.—Third execution of subroutine RECIP for the example in Figure 50(a) using the option mycip="yes".....

```
seismc

type input file name: tcip1

input complete from 3 dsk files named     tcip1
type subroutine: recip
recip entry 1                    file=    tcip1
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*2 lin 200 no no 1 yes 433;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 org 500 7500 100 100;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 org 3000 10000 100 100;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
print t-x data on horizon    2 ? yes or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters    surface coordinates
      xi        eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                              right sp left sp
   4187.9    -748.6    33.6     0.0   207.36    0.00     0.00   4003.7   4427.8
   4390.0    -756.1    33.6    33.6   213.39    6.02   202.20   4203.7   4632.5
   4592.0    -763.7    33.6    33.6   219.41    6.02   202.20   4403.7   4837.3
   4794.1    -771.3    33.6    33.6   225.43    6.02   202.19   4603.7   5042.0
   4996.1    -778.8    33.6    33.6   231.45    6.02   202.18   4803.7   5246.7
   5198.2    -786.4    33.6    33.6   237.48    6.02   202.20   5003.7   5451.4
   5400.2    -793.9    33.6    33.6   243.50    6.02   202.21   5203.7   5656.2
   5602.3    -801.5    33.6    33.6   249.52    6.02   202.18   5403.7   5860.9
   5804.3    -809.0    33.6    33.6   255.55    6.02   202.20   5603.7   6065.6
   6006.3    -816.6    33.6    33.6   261.57    6.02   202.14   5803.7   6270.3
   6208.4    -824.1    33.6    33.6   267.59    6.02   202.23   6003.7   6475.1
   6410.5    -831.7    33.6    33.6   273.61    6.02   202.19   6203.7   6679.8
   6612.5    -839.2    33.6    33.6   279.64    6.02   202.18   6403.7   6884.5
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
   4003.7 to    4003.7  and    6403.7 to    9003.7
and for the left shotpoint, in the intervals:
    995.3 to    4427.8  and    6884.5 to    6995.3
do you wish the xstar points? yes or no:no
type subroutine: QUIT
```

Table 44.--(a) Printout of file tcip2.com showing the t-x data points of Figure 51.
(b) Printout of file tcipl.lay showing horizons 1, 2 and 3 of Figure 52.....

(a) print tcip2.com

tcip2.com

```
1 46        0.00        3.00       -1.00
    20.00       30.00
    40.00       44.00
    60.00       55.00
    80.00       65.00
   100.00       80.00
   120.00       97.00
   140.00      108.00
   160.00      115.00
   180.00      126.00
   200.00      137.00
   220.00      142.00
   240.00      148.00
   260.00      150.00
   280.00      155.00
   300.00      164.00
   320.00      166.00
   340.00      173.00
   360.00      175.00
   380.00      177.00
   400.00      187.00
   420.00      191.00
   440.00      197.00
   460.00      207.00
   480.00      215.00
   500.00      219.00
   520.00      228.00
   540.00      235.00
   560.00      238.00
   580.00      241.00
   600.00      244.00
   620.00      248.00
   640.00      248.00
   660.00      253.00
   680.00      254.00
   700.00      255.00
   720.00      261.00
   740.00      263.00
   760.00      263.00
   780.00      268.00
   800.00      265.00
   820.00      273.00
   840.00      274.00
   860.00      277.00
   880.00      280.00
   900.00      285.00
   920.00      290.00
2 46      920.00        3.00       -1.00
     0.00      290.00
    20.00      287.00
    40.00      285.00
    60.00      284.00
    80.00      280.00
   100.00      282.00
   120.00      277.00
   140.00      279.00
   160.00      279.00
   180.00      278.00
   200.00      275.00
   220.00      276.00
   240.00      274.00
   260.00      277.00
   280.00      278.00
   300.00      278.00
   320.00      277.00
   340.00      275.00
   360.00      277.00
   380.00      272.00
   400.00      261.00
   420.00      255.00
   440.00      252.00
   460.00      245.00
   480.00      236.00
   500.00      234.00
   520.00      228.00
   540.00      223.00
   560.00      218.00
   580.00      213.00
   600.00      207.00
   620.00      202.00
   640.00      195.00
   660.00      190.00
   680.00      180.00
   700.00      168.00
   720.00      159.00
   740.00      150.00
   760.00      142.00
   780.00      133.00
   800.00      120.00
   820.00      102.00
   840.00       93.00
   860.00       74.00
   880.00       61.00
   900.00       40.00
```

(b) print tcip2.lay

tcip2.lay

```
1  6  -90000.00          0.00          0.65
      0.00          0.00          0.65
    330.00         16.00          0.60
    660.00         32.00          0.55
   1000.00         50.00          0.55
  90000.00         50.00          0.55
2  4  -90000.00        -10.00          1.50
      0.00        -10.00          1.50
   1000.00         39.00          1.50
  90000.00         39.00          1.50
3  4  -90000.00        -28.00          2.15
      0.00        -28.00          2.15
   1000.00          0.00          2.15
  90000.00          0.00          2.15
```

Table 45.--Input form with input values for three examples used to interpret horizon 4 of the reversed arrival-time curves shown in Figure 51.....

RECIP input form — File name tcip 2

(optional parameters in parentheses)

Entry 1

	id	curv lin/spl	delx	(bend1 yes/no	bend2 yes/no	iu	mycip yes/no	tss	tuh1	tuhr)
1.	3	lin	40							
2.	3	lin	20	no	no	1	yes	370	6	6
3.	3	lin	40	no	no	1	yes	370	6	6

Entry 2 (left shotpoint)

	kl	extl org/com	xminl	xmaxl	delxl	dintl	(nvall	nprinl 0/1	nstarl)
1.	1	com	190	550	20	20			
2.	1	com	190	550	20	20			
3.	1	com	190	550	20	20			

Entry 2a extend data to left

	optll	±dxll	±fnll(i)
1.	1	1	0 88
2.	0		
3.	0		

Entry 2b extend data to right

	optlr	±dxlr	±fnlr(i)
1.	1	1	975 383
2.	0		
3.	1	1	800 323

Entry 3 (right shotpoint)

	kr	extr org/com	xminr	xmaxr	delxr	dintr	(nvalr	nprinr 0/1	nstarr)
1.	2	com	350	670	20	20			
2.	2	com	350	670	20	20			
3.	2	com	350	670	20	20			

Entry 3a extend data to left

	optrl	±dxrl	±fnrl(i)
1.	1	1	-50 377
2.	0		
3.	1	1	100 341

Entry 3b extend data to right

	optrr	±dxrr	±fnrr(i)
1.	1	1	920 118
2.	0		
3.	0		

Entry 2c (optional artificial; if xmin = xmaxr)

	nl	x1(i) i = 1, nl	t1(i) i = 1, nl
1.	-		
2.	-		
3.	-		

Entry 3c (optional artificial; if xminr = xmaxr

	nr	xr(i) i = 1, nv	tr(i) i = 1, nv
1.	-		
2.	-		
3.	-		

Table 46.--First execution of subroutine RECIP for horizon 4 for the example in Figure 51.....

```
seismo

type input file name: tcip2

input complete from 3 dsk files named     tcip2
type subroutine: recip
recip entry 1                        file=    tcip2
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*3 lin 40;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 190 550 20 20;
recip entry 2a; optli*,+or-dxll,+or-fnll(i)
*1 1 0 88;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*1 1 975 383;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 com 350 670 20 20;
recip entry 3a; optri*,+or-dxrl,+or-fnrl(i)
*1 1 -50 377;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*1 1 920 118;
recip times adding uphole;  left to right, right to left, and average
     368.9      370.3      369.6
do you want to continue? yes or no:yes
print t-x data on horizon    3 ? yes or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters     surface coordinates
       xi        eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                                   right sp left sp
    -10.5     -78.4      4.2      0.0   43.07      0.00      0.00   -41.3     17.2
     28.7     -77.3      4.3      4.3   52.15      9.09     39.24    -1.3     56.2
     72.2     -75.2      4.1      4.2   62.61     10.45     43.58     38.7     99.3
    111.7     -73.0      4.1      4.1   72.26      9.65     39.53     78.7    138.2
    150.9     -70.8      4.1      4.1   81.87      9.62     39.30    118.7    176.9
    190.1     -68.4      4.1      4.1   91.41      9.54     39.21    158.7    214.3
    229.2     -66.1      4.2      4.1  100.94      9.53     39.20    198.7    254.2
    268.4     -63.7      4.0      4.1  110.47      9.54     39.21    238.7    290.7
    307.2     -60.7      4.3      4.2  119.72      9.24     38.99    278.7    326.2
    347.9     -56.8      4.2      4.2  129.45      9.73     40.87    318.7    373.1
    392.9     -56.7      3.1      3.4  142.70     13.25     45.01    358.7    437.3
    444.6     -57.7      2.9      3.0  159.92     17.22     51.70    398.7    489.8
    481.5     -56.9      3.0      3.0  172.44     12.52     36.95    438.7    525.3
    517.8     -57.1      3.1      3.1  184.24     11.80     36.28    478.7    560.8
    552.4     -58.0      3.3      3.2  194.98     10.74     34.58    518.7    595.8
    594.2     -59.8      3.5      3.4  207.25     12.27     41.83    558.7    635.7
    638.6     -60.4      3.4      3.5  220.10     12.85     44.37    598.7    681.5
    685.2     -60.6      3.2      3.3  234.30     14.21     46.64    638.7    732.9
    730.0     -59.8      3.1      3.2  248.52     14.22     44.87    678.7    778.3
    764.6     -59.3      3.2      3.2  259.45     10.93     34.51    718.7    813.2
    804.8     -59.5      3.2      3.2  271.89     12.44     40.23    758.7    854.6
    846.0     -59.8      3.2      3.2  284.61     12.72     41.24    798.7    897.2
    887.3     -60.0      3.2      3.2  297.33     12.72     41.23    838.7    939.8
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
    -41.3 to     -41.3  and     838.7 to     941.8
and for the left shotpoint, in the intervals:
     -8.2 to      17.2  and     939.8 to     953.5
do you wish the xstar points? yes or no:yes
the path of special points (xstar) from horizon iu to id
left-hand shot-point
    x(iu)  time(iu)     x(id)  time(id)
    920.0     368.9     899.1     320.8
      0.0      92.6      -8.2      64.2
    108.3     119.2     100.2      89.3
    216.7     145.9     207.9     114.4
    325.0     173.2     317.6     139.6
    433.3     200.5     417.5     163.7
    541.7     240.1     523.4     200.6
    650.0     277.0     632.9     236.6
    758.3     313.9     739.3     269.6
    866.7     350.8     846.4     303.9
    975.0     387.6     953.5     338.2
the right-hand shot-point
      0.0     370.3      10.0     341.6
    -50.0     382.5     -41.3     354.0
     57.8     356.2      68.4     326.8
    165.6     329.9     177.2     298.9
    273.3     303.6     285.9     271.1
    381.1     275.0     399.9     238.0
    488.9     240.8     504.0     204.5
    596.7     213.7     614.4     174.5
    704.4     183.1     723.7     140.6
    812.2     153.3     832.7     108.3
    920.0     123.5     941.8      76.0
type subroutine: QUIT
```

508 ◄ ► 490

Table 45(2) shows the entry values for the same problem for which the data are not extended and the reciprocal time is entered from the terminal. The uphole times tuhl and tuhr, are estimated from the shot depth and the velocity of the first layer. The reciprocal time tss, is obtained by reading the reciprocal time from the t-x plot and adding the average uphole time. Execution is shown in table 47a. No solutions are given because the program could determine no ray intersections which satisfied the necessary conditions. However, from the location of the two heavy circles in figure 51, it is evident that some solutions for this problem should exist. They would have been found by decreasing the entry value of delx to around 5.

Table 45(3) shows entry values in which the reciprocal time is input from the terminal, and the data are extended to points (100,341) and (800,323) (fig. 51) to insure of solutions. Execution is shown in table 47b. The results differ a little from table 46 because the values of uphole time and reciprocal time differ slightly from those computed for this case as shown in table 46.

Values for layer four taken from figure 52 are entered into file tcip2.lay in table 48a. Execution of subroutine RAYTRACE to check these results is shown in tables 48b and c. The results are shown plotted by the symbol x in figure 51 and agreement with the original t-x values is good. Values depart somewhat on the upper left-hand part of the plot probably because the initial data extension was not quite correct. Because the uphole times at opposite shotpoints differ slightly, reciprocal times on the extended data should not agree precisely, but instead differ by the difference in uphole times. Furthermore, we note that, for this problem, agreement from RAYTRACE is satisfactory for the right shotpoint at distancecoordinates 350 to 400 whereas there was a discrepancy in the previous section using subroutines CRIT and ACROSS. In general, when data can be readily extended, we find it more satisfactory to use RECIP than CRIT and ACROSS. However, in this case there is basic agreement, as there should be, between the solutions obtained from the two methods. Occasionally, when errors are expected from RECIP, or simply as a check, both RECIP and CRIT are executed. The critical reflection point from CRIT should fall on the horizon calculated from RECIP.

Computation of horizon 5 is demonstrated with figure 53 and entry values in table 49. Data are extended beyond the intercept to coordinates (-50,157) and (1000,264). Without extensions there would be few, if any, depth points for horizon 5 common to the two plots. Data from complementary shotpoints offset 600 distance units or more from each end of the line would have provided constraints for extending these data. Program execution is shown in tables 50a, b, and c. Results of table 50a are shown plotted in figure 52 as horizon 5. These plotted values input into file tcip2.lay are shown in table 51a. Figure 52 shows strong velocity variations within layer 5; it varies from a low of 7.3 to a high of 600. One may expect strange answers when limited data are extended without constraints. However, the velocities shown are necessary to satisfy the data. Table 50b shows execution of RECIP without data extension; and not enough data were available to calculate solutions. Table 50c shows execution using the

Table 47.--(a) Second execution of subroutine RECIP for horizon 4 for the example in Figure 51.
(b) Third execution of subroutine RECIP for horizon 4 for the example in Figure 51.....

```
seismo
(a) type input file name: toip2

input complete from 3 dsk files named     toip2
type subroutine: recip
recip entry 1                    file=    toip2
id*,curv,delx,<bend1,bend2,iu,mycip,tas,tuhl,tuhr>
*3 iin 20 no no 1 yes 365 6 6;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 190 550 20 20;
recip entry 2a; optli*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 com 350 670 20 20;
recip entry 3a; optri*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
print t-x data on horizon    3 ? yes or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters    surface coordinates
      xi       eta velocity velocity    tau    delta-t delta-x x-minus  xplus
                                                              right sp left sp
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
    382.2 to       0.0  and       0.0 to     681.8
and for the left shotpoint, in the intervals:
    191.1 to       0.0  and       0.0 to     520.5
do you wish the xstar points? yes or no:no
(b) type subroutine: recip
recip entry 1                    file=    toip2
id*,curv,delx,<bend1,bend2,iu,mycip,tas,tuhl,tuhr>
*3 iin 40 no no 1 yes 370 6 6;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 190 550 20 20;
recip entry 2a; optli*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*1 1 800 323;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 com 350 670 20 20;
recip entry 3a; optri*,+or-dxrl,+or-fnrl(i)
*1 1 100 341;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
print t-x data on horizon    3 ? yes or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters    surface coordinates
      xi       eta velocity velocity    tau    delta-t delta-x x-minus  xplus
                                                              right sp left sp
    223.8     -68.7     4.1      0.0   99.96     0.00    0.00    191.1    249.2
    263.0     -66.5     4.1      4.0  109.67     9.71   39.29    231.1    288.7
    301.5     -63.2     4.0      4.2  118.77     9.10   38.61    271.1    320.6
    341.5     -59.5     4.4      4.2  128.24     9.47   40.16    311.1    364.3
    387.1     -59.2     3.2      3.5  141.34    13.10   45.64    351.1    431.6
    441.4     -60.3     2.9      3.0  159.41    18.07   54.31    391.1    489.2
    476.3     -59.2     3.0      2.9  171.35    11.94   34.95    431.1    522.1
    512.4     -59.1     3.1      3.1  183.12    11.77   36.04    471.1    557.2
    545.9     -59.6     3.3      3.2  193.63    10.50   33.51    511.1    589.9
    587.2     -61.3     3.4      3.4  205.75    12.12   41.36    551.1    630.2
    632.2     -62.1     3.4      3.5  218.72    12.97   45.03    591.1    676.2
    681.4     -62.2     3.2      3.3  233.58    14.86   49.18    631.1    729.9
    727.4     -60.6     3.1      3.1  248.36    14.78   46.04    671.1    775.8
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
    111.1 to     191.1  and     671.1 to     681.9
and for the left shotpoint, in the intervals:
    191.0 to     249.2  and     775.8 to     780.6
do you wish the xstar points? yes or no:no
type subroutine: QUIT
```

(a)
```
print tcip2.lay

                    tcip2.lay


1  6  -90000.00          0.00          0.65
      0.00          0.00          0.65
    330.00         16.00          0.60
    660.00         32.00          0.55
   1000.00         50.00          0.55
  90000.00         50.00          0.55
2  4  -90000.00        -10.00          1.50
      0.00        -10.00          1.50
   1000.00         39.00          1.50
  90000.00         39.00          1.50
3  4  -90000.00        -28.00          2.15
      0.00        -28.00          2.15
   1000.00          0.00          2.15
  90000.00          0.00          2.15
4 11  -90000.00        -78.00          4.20
    -10.00        -78.00          4.20
     72.00        -75.00          4.10
    268.00        -64.00          4.20
    348.00        -57.00          3.40
    393.00        -57.00          3.00
    481.00        -57.00          3.15
    552.00        -58.00          3.40
    685.00        -61.00          3.20
    887.00        -60.00          3.20
  90000.00        -60.00          3.20
```

```
seismo
```

(b)
```
type input file name: tcip2

input complete from 3 dsk files named     tcip2
type subroutine: raytr
raytrace entry 1                          file=    tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 0 3 0 1000 50;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation       time      refrac. angle
      4          39.97       -76.17         50.57       -84.32

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=    3.00
  from off horizon   4
 x-refractor coord.    x-surface coord.   arrival time
          39.97              75.30         106.54
          89.97             124.82         118.75
         139.97             174.40         130.91
         189.97             223.99         143.07
         239.97             273.57         155.23
         289.97             319.29         166.08
         339.97             368.03         178.40
         389.97             437.70         198.91
         439.97             501.36         221.09
         489.97             549.67         238.09
         539.97             602.38         256.15
         589.97             648.53         271.26
         639.97             701.17         290.10
         689.97             757.07         308.69
         739.97             808.72         326.25
         789.97             860.38         343.81
         839.97             912.03         361.38
         889.97             964.38         379.18
         939.97            1016.27         396.22
         989.97            1066.96         412.02
```

(c)
```
all done. another shot!
raytrace entry 1                          file=    tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 920 3 920 -500 50;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.    elevation       time      refrac. angle
      4          848.99       -60.19         78.11        83.42

do you want to continue? yes or no:yes
arrival time data for shot at x=     920.00 depth=    3.00
  from off horizon   4
 x-refractor coord.    x-surface coord.   arrival time
          848.98             778.26         156.91
          798.98             729.88         170.72
          748.98             681.50         184.52
          698.98             632.95         197.10
          648.98             593.87         207.68
          598.98             546.27         220.24
          548.98             491.42         234.90
          498.98             443.81         248.78
          448.98             389.84         265.24
          398.98             341.81         280.28
          348.98             303.65         289.41
          298.98             257.04         300.95
          248.98             207.06         313.27
          198.98             156.56         325.53
          148.98             106.05         337.80
           98.98              55.55         350.06
           48.98               8.18         361.28
           -1.02             -42.15         373.94
          -51.02             -89.26         385.26
         -101.02            -139.26         397.17
         -151.02            -189.26         409.07
         -201.02            -239.26         420.98
         -251.02            -289.26         432.88
         -301.02            -339.26         444.79
         -351.02            -389.26         456.69
         -401.02            -439.26         468.60
         -451.02            -489.26         480.50
all done. another shot!
raytrace entry 1                          file=    tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*QUIT
```

Table 48.--(a) Printout of file tcip2.lay showing the first four horizons of Figure 52.
(b) and (c) Execution of subroutine RAYTRACE for horizon 4.....

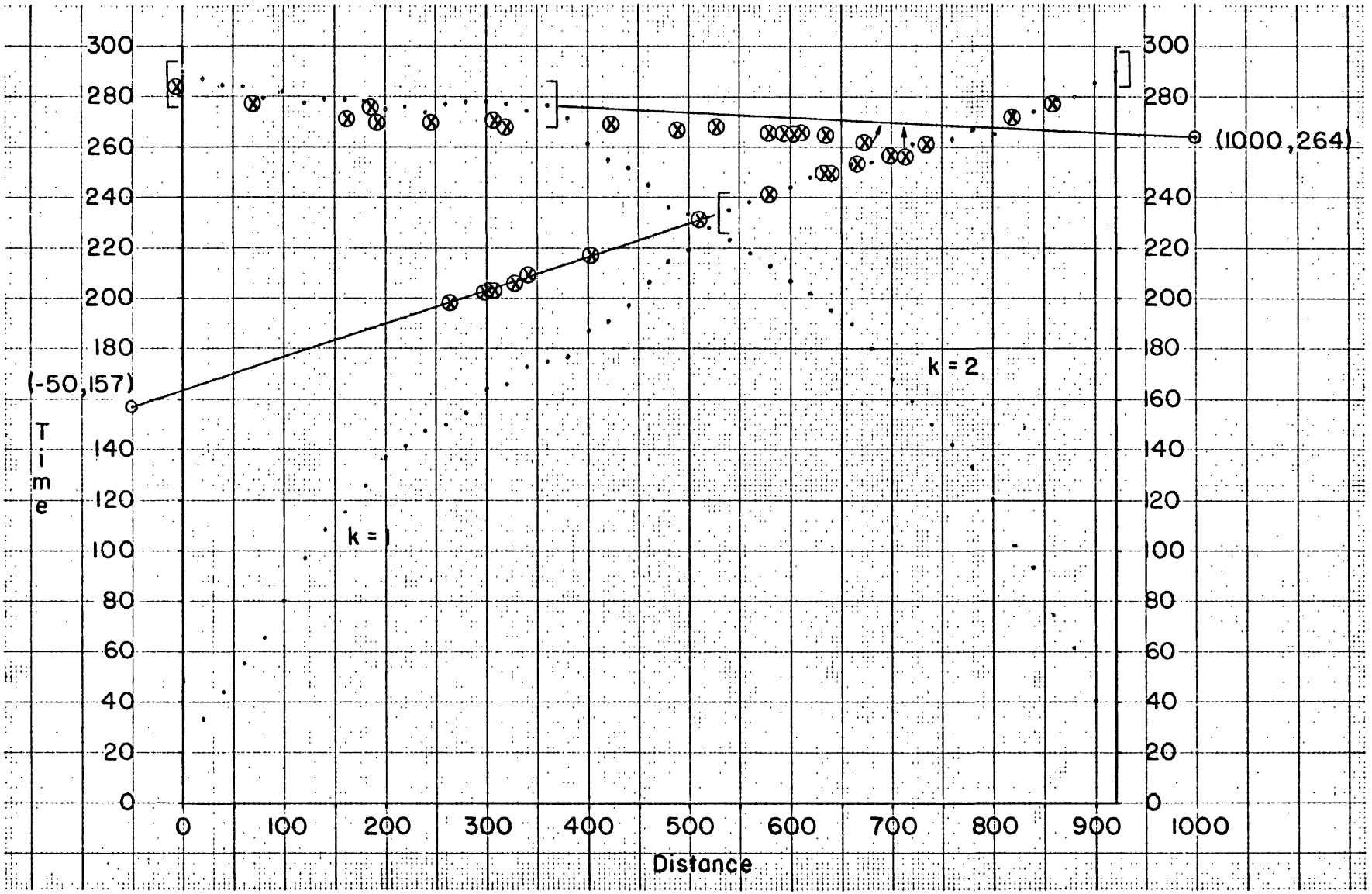

Figure 53.--Third example to illustrate execution of subroutine RECIP as described in text.....

Table 49.--Input form with input values for three examples used to interpret horizon 5 of the reversed arrival-time curves shown in Figure 53.....

RECIP input form — File name taip 2

(optional parameters in parentheses)

Entry 1

	id	curv lin/spl	delx	(bend1 yes/no	bend2 yes/no	iu	mycip yes/no	tss	tuh1	tuhr)
1.	4	lin	40							
2.	4	lin	40							
3.	4	lin	40	yes	yes					

Entry 2 (left shotpoint)

	kl	extl org/com	xminl	xmaxl	delxl	dintl	(nval1	nprinl 0/1	nstarl)
1.	1	com	530	1000	15	15			
2.	1	com	530	1000	15	15			
3.	1	com	530	1000	15	15			

Entry 2a extend data to left

	optll	±dxll	±fnll(i)
1.	1	1	-50 157
2.	0		
3.	1	1	-50 157

Entry 2b extend data to right

	optlr	±dxlr	±fnlr(i)
1.	0		
2.	0		
3.	0		

Entry 3 (right shotpoint)

	kr	extr org/com	xminr	xmaxr	delxr	dintr	(nvalr	nprinr 0/1	nstarr)
1.	2	com	-10	370	15	15			
2.	2	com	-10	370	15	15			
3.	2	com	-10	370	15	15			

Entry 3a extend data to left

	optrl	±dxrl	±fnrl(i)
1.	0		
2.	0		
3.	0		

Entry 3b extend data to right

	optrr	±dxrr	±fnrr(i)
1.	1	1	1000 264
2.	0		
3.	1	1	1000 264

Entry 2c (optional artificial; if xmin = xmaxr)

	nl	x1(i) i = 1, nl	t1(i) i = 1, nl
1.			
2.			
3.			

Entry 3c (optional artificial; if xminr = xmaxr

	nr	xr(i) i = 1, nv	tr(i) i = 1, nv
1.			
2.			
3.			

(a)

```
seismc
type input file name: tcip2

input complete from 3 dsk files named     tcip2
type subroutine: recip
recip entry 1                    file=    tcip2
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*4 lin 40;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 530 1000 15 15;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*1 1 -50 157;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 com -10 370 15 15;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*1 1 1000 264;
rscip times adding uphole;  left to right, right to left, and average
     294.6      295.5      295.0
do you want to continue? yes or no:yes
print t-x data on horizon     4 ? yes or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters     surface coordinates
       xi       eta velocity velocity     tau     delta-t delta-x x-minus  xplus
                                                               right sp left sp
    130.4    -241.5      7.8      0.0   99.02      0.00      0.00     21.5    229.7
    148.3    -240.2      8.6      7.3  101.49      2.46     17.95     61.5    249.1
    169.4    -241.0      9.4      8.6  103.96      2.47     21.16    101.5    275.3
    169.3    -241.1     12.3     -4.2  103.93     -0.03     -0.13    141.5    275.3
    227.7    -251.1     11.9     11.8  108.93      5.00     59.19    181.5    321.1
    256.0    -252.9     13.0     13.9  110.97      2.04     28.36    221.5    341.6
    239.6    -251.0     24.1    -17.3  110.02     -0.95    -16.45    261.5    328.3
    306.5    -258.8     20.3    599.6  110.13      0.11     67.31    301.5    379.5
    383.6    -228.8     12.2     19.4  114.39      4.25     82.73    341.5    441.5
    398.8    -223.1     14.1      7.3  116.61      2.22     16.24    381.5    454.4
    443.9    -222.8     13.3      7.5  122.60      5.99     45.13    421.5    504.0
    486.2    -232.2     11.2      9.7  127.07      4.47     43.37    461.5    563.9
    530.5    -245.0     12.0     11.9  130.93      3.86     46.10    501.5    601.3
    572.1    -255.0     12.9     20.1  133.07      2.13     42.80    541.5    623.6
    616.3    -259.0     13.5     25.1  134.83      1.77     44.36    581.5    684.9
    661.7    -258.6     13.5     19.7  137.14      2.30     45.39    621.5    722.0
    714.1    -253.3     11.2     13.2  141.12      3.98     52.72    661.5    752.5
    737.1    -251.8     11.1      8.0  143.97      2.85     22.96    701.5    825.5
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
     21.5 to      21.5  and     701.5 to    1007.6
and for the left shotpoint, in the intervals:
    -69.3 to     229.7  and     825.5 to     879.1
do you wish the xstar points? yes or no:yes
the path of special points (xstar) from horizon iu to id
left-hand shot-point
    x(iu)  time(iu)     x(id)  time(id)
    920.0     294.6     879.1     218.9
    -50.0     161.6     -69.3     109.7
     57.8     175.9      42.8     123.8
    165.6     190.1     150.8     137.8
    273.3     204.4     259.0     152.0
    381.1     218.6     367.5     165.7
    488.9     232.9     474.2     177.1
    596.7     248.3     577.2     188.4
    704.4     261.6     682.9     196.4
    812.2     273.9     788.9     205.2
    920.0     294.6     879.1     218.9
the right-hand shot-point
      0.0     295.4      21.5     243.6
      0.0     295.4      21.5     243.6
    111.1     284.4     124.2     233.6
    222.2     280.7     229.2     229.9
    333.3     281.7     343.5     231.0
    444.4     280.7     451.3     226.8
    555.6     278.5     563.1     221.1
    666.7     276.2     680.3     213.1
    777.8     274.0     787.9     207.2
    888.9     271.7     899.6     201.3
   1000.0     269.5    1007.6     195.4
```

(b)

```
type subroutine: recip
recip entry 1                    file=    tcip2
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*4 lin 40;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 530 1000 15 15;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 com -10 370 15 15;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
recip times adding uphole;  left to right, right to left, and average
     294.6      295.5      295.0
do you want to continue? yes or no:yes
print t-x data on horizon     4 ? yes or no:no
The reciprocal time solutions are aa follows
            point values          incremental parameters     surface coordinates
       xi       eta velocity velocity     tau     delta-t delta-x x-minus  xplus
                                                               right sp left sp
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
     21.4 to       0.0  and       0.0 to     364.9
```

Table 50.--(a) First execution of subroutine RECIP for horizon 5 for the example of Figure 52.
(b) Second execution of subroutine RECIP for horizon 5.
(c) Third execution of subroutine RECIP for horizon 5.....

Table 50.--Continued

```
and for the left shotpoint, in the intervals:
    521.8 to       0.0  and       0.0 to     878.9
(c)do you wish the xstar points? yes or no:no
type subroutine: recip
recip entry 1                          file=    tcip2
id*,curv,delx,<bend1,bend2,iu,myoip,tas,tuhl,tuhr>
*4 lin 40 yes yes ;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 com 530 1000 15 15;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*1 1 -50 157;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 oom -10 370 15 15;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*1 1 1000 264;
recip times adding uphole;  left to right, right to left, and average
     294.6       295.5       295.0
do you want to continue? yes or no:yes
print t-x data on horizon     4 ? yea or no:no
The reciprocal time solutions are as follows
            point values          incremental parameters     surface coordinates
       xi       eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                               right sp left sp
    134.8    -241.3      7.6      0.0   99.60       0.00     0.00    21.5   234.1
    156.6    -240.3      8.2      7.5  102.48       2.89    21.78    61.5   266.6
    169.4    -240.9      8.9      8.8  103.94       1.46    12.85   101.5   282.2
    169.3    -241.0     11.3     -4.1  103.91      -0.03    -0.12   141.5   282.1
    229.1    -247.3     11.2     12.7  108.63       4.72    60.11   181.5   319.2
    366.8    -207.0     17.5     73.8  110.58       1.94   143.43   301.5   319.9
    472.2    -218.1      4.5      5.9  128.56      17.98   106.06   341.5   567.4
    423.4    -223.2      7.5     -5.2  119.16      -9.40   -49.11   381.5   478.7
    443.9    -223.1     13.3      5.8  122.70       3.54    20.55   421.5   513.4
    474.6    -228.4      7.9      8.4  126.39       3.69    31.12   461.5   571.4
    527.8    -244.4      9.3     12.6  130.79       4.40    55.61   501.5   603.0
    533.4    -242.4     16.7      4.3  132.16       1.36     5.89   541.5   603.7
    618.3    -257.4     13.5     33.3  134.74       2.59    86.24   581.5   679.3
    673.6    -248.5     12.8     31.0  136.55       1.81    56.07   621.5   682.8
    734.2    -253.1      7.9      9.1  143.22       6.67    60.74   661.5   822.3
    737.1    -251.8     11.1      4.2  143.97       0.75     3.17   701.5   825.5
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
     21.5 to      21.5  and     701.5 to    1007.6
and for the left shotpoint, in the intervals:
    -69.3 to     234.1  and     825.5 to     879.1
do you wish the xstar points? yes or no:no
type subroutine: QUIT
```

option yes for the variables bend1 and bend2 in entry 1 (table 49(3). In general, the results are similar to those of table 50a. We have found that results tend to scatter if rays are permitted to bend at compartment boundaries.

Tables 51b and d show execution of subroutine RAYTRACE for horizon 5 for both shotpoints. (Tables 51c and 51e use the option yes for the variable bend.) The results from tables 51b and 51d are plotted in figure 53 with the circled symbol x. From 51b (left shotpoint), the RAYTRACE results agree quite well with the raw data, but for 51c they are about 5 units too small, except for the two far right points which indicate an even larger error. This apparent error is easily reconciled upon considering that the output shown in table 50a is for distances between 130.4 and 737.1 and values were arbitrarily extended beyond these limits, shown as the dashed line in figure 52. The execution of RECIP apparently did not calculate results for the end points of the left shotpoint. Execution of subroutine RAYTRACE (table 51d) projected the down-going ray to intersect horizon 5 at the distance coordinate 846.95 (also shown by a square in figure 52) which lies outside the range of solutions provided by subroutine RECIP. All the computed time values in table 51d could be increased to match the raw data by increasing the depth to horizon 5 on the right side, where it is shown dashed. Thus horizon 5 where shown as a solid line is correct and incorrect where shown dashed. Inspection of the last 4 t-x data points from the left shotpoint in figure 53 shows an abrupt decrease in apparent velocity, suggesting that the dip of the dashed line to the right in figure 52 should be reversed. Execution of subroutine CIP and ACROSS for the left shotpoint would undoubtedly demonstrate this quite clearly. The problem of arrival-times being in error by a constant did not occur for the right shotpoint because, as shown by the square symbol in figure 52, the down-going ray intersects horizon 5 within the range of results computed by RECIP.

The results from subroutine RECIP sometimes show considerable scatter in both the computed depth points and velocity values, particularly if the raw data is scattered. One way of overcoming this is to draw average lines through the data points and assume that these lines represent the true arrivals, or to increase the value of nval1 and nvalr in entries 2 and 3. However, this procedure may not help, particularly if the depth interval between horizon id and the one being calculated is large. In this case, inserting phantom horizons below the idth, but shallower than the one being calculated, may significantly reduce the scatter. These phantom horizons may have exactly the same lateral velocity variations as the overlying one, or the velocity change may be smoothed somewhat. For example, in figure 52, any number of horizons could be inserted between the elevations of -100 and -200. If two were inserted, these horizons would have to be input into file tcip2.lay and the horizon index for id on the entry form would also be increased by two. Inserting horizons has some filtering effect, but more importantly prevents rays from uncontrolled propagation over large distances.

(a)
```
print tcip2.lay

                    tcip2.lay


 1  6  -90000.00          0.00          0.65
      0.00          0.00          0.65
    330.00         16.00          0.60
    660.00         32.00          0.55
   1000.00         50.00          0.55
  90000.00         50.00          0.55
 2  4  -90000.00        -10.00          1.50
      0.00        -10.00          1.50
   1000.00         39.00          1.50
  90000.00         39.00          1.50
 3  4  -90000.00        -28.00          2.15
      0.00        -28.00          2.15
   1000.00          0.00          2.15
  90000.00          0.00          2.15
 4 11  -90000.00        -78.00          4.20
    -10.00        -78.00          4.20
     72.00        -75.00          4.10
    268.00        -64.00          4.20
    348.00        -57.00          3.40
    393.00        -57.00          3.00
    481.00        -57.00          3.15
    552.00        -58.00          3.40
    685.00        -61.00          3.20
    887.00        -60.00          3.20
  90000.00        -60.00          3.20
 5 18    -100.00      -241.00          7.30
    130.00       -241.00          7.30
    148.00       -240.00          8.60
    169.00       -241.00         11.80
    228.00       -251.00         13.90
    256.00       -253.00        600.00
    306.00       -259.00         19.40
    383.00       -229.00          7.40
    398.00       -223.00          7.40
    444.00       -223.00          9.70
    486.00       -232.00         11.90
    530.00       -245.00         20.10
    572.00       -255.00         25.10
    616.00       -259.00         19.70
    662.00       -259.00         13.20
    714.00       -253.00          8.00
    737.00       -252.00          8.00
   1000.00       -224.00          8.00

seismc

type input file name: tcip2
```

(b)
```
input complete from 3 dsk files named     tcip2
type subroutine: raytr
raytrace entry 1                        file=   tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*5 0 3 0 800 40;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
      5        144.74       -240.18        96.53        -81.01

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=   3.00
  from off horizon   5
 x-refractor coord.    x-surface coord.  arrival time
         144.74              265.12        198.80
         184.74              301.68        203.39
         224.74              347.69        209.92
         264.74              289.91        203.60
         304.74              330.96        206.20
         344.74              308.42        203.55
         384.74              405.11        217.94
         424.74              512.07        232.82
         464.74              578.82        242.18
         504.74              634.74        249.67
         544.74              639.28        249.13
         584.74              640.49        249.27
         624.74              667.90        253.39
         664.74              700.62        257.37
         704.74              736.17        262.56
         744.74              821.04        272.09
         784.74              860.06        277.04
```

(c)
```
all done. another shot!
raytrace entry 1                        file=   tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*5 0 3 0 800 40 yes ;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
      5        124.27       -241.00        93.76        -81.17

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=   3.00
  from off horizon   5
 x-refractor coord.    x-surface coord.  arrival time
         124.27              260.38        198.21
         164.27              283.24        201.31
```

Table 51.--(a) Execution of subroutine DATAIN to input the interpreted velocity section of horizon 5 into file tcip2.lay.
(b) through (e) Execution of subroutine RAYTRACE for horizon 5.....

Table 51.--Continued

```
          204.27              318.13        205.28
          244.27              314.81        205.13
          284.27              309.98        204.20
          324.27              286.89        204.22
          364.27              374.11        215.74
          404.27              490.02        229.84
          444.27              537.83        235.98
          484.27              588.10        242.74
          524.27              625.12        247.83
          564.27              664.38        253.79
          604.27              660.49        252.68
          644.27              687.49        255.16
          684.27              779.20        268.04
          724.27              817.66        271.60
          764.27              840.09        274.53
```

(d)

```
all done. another shot!
raytrace entry 1                    file=    tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*5 920 3 920 50 40;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
     5         846.95       -240.29        125.53         83.75

do you want to continue? yes or no:yes
arrival time data for shot at x=     920.00 depth=    3.00
  from off horizon   5
 x-refractor coord.    x-surface coord.   arrival time
          846.95              716.52        256.84
          806.95              674.80        261.95
          766.95              635.42        264.47
          726.95              611.81        266.19
          686.95              596.69        265.87
          646.95              604.06        265.79
          606.95              595.91        266.22
          566.95              582.28        266.25
          526.95              531.20        268.27
          486.95              490.94        266.89
          446.95              424.58        269.59
          406.95              320.00        268.82
          366.95              247.73        270.82
          326.95              187.91        276.24
          286.95              309.48        270.25
          246.95              194.33        270.82
          206.95              164.77        271.90
          166.95               70.03        276.78
          126.95               -4.14        284.43
           86.95              -42.50        289.51
all done. another shot!
raytrace entry 1                    file=    tcip2
```

(e)

```
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*5 920 3 920 50 40 yes ;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),......
*0;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
     5         846.95       -240.29        125.53         83.75

do you want to continue? yes or no:yes
arrival time data for shot at x=     920.00 depth=    3.00
  from off horizon   5
 x-refractor coord.    x-surface coord.   arrival time
          846.95              716.52        256.84
          806.95              674.80        261.95
          766.95              647.89        263.22
          726.95              637.48        264.16
          686.95              660.12        264.92
          646.95              604.06        265.79
          606.95              595.91        266.22
          566.95              582.28        266.25
          526.95              531.20        268.27
          486.95              490.94        266.89
          446.95              424.58        269.59
          406.95              469.24        281.69
          366.95              483.69        295.43
          326.95              182.14        276.96
          286.95              309.48        270.25
          246.95              194.33        270.82
          206.95              164.77        271.90
          166.95               70.03        276.78
          126.95                1.03        283.67
           86.95              -33.90        288.40
all done. another shot!
raytrace entry 1                    file=    tcip2
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*QUIT
```

Subroutine RECIP begins to fail in the case of large near-vertical offsets for which the basic program assumptions are violated. Such a problem is shown by horizon 3 represented by the heavy line in figure 54. This model is input into file tcip3.lay and listed in table 52a. The t-x data for horizon 3 are calculated from subroutine RAYTRACE (table 52b) and the minimum values of the combined refraction-diffraction results plotted with the symbol o in figure 55. These points have been connected by a smooth curve which then represent the t-x values and are input into file tcip3.org (table 53). The problem then is to execute subroutine RECIP and check how nearly the result approximates the original model in figure 54. (Note that in executing subroutine DATAIN in table 53, the value -1 was used for the variable offset. By so doing, actual distances could be input instead of distance intervals as is usually done.)

Execution of subroutine RECIP is shown in both tables 54 and 55. The input for both are identical except that a phantom horizon has been inserted into the velocity model at -500 elevation with velocity 3.0 for table 55. The results for both are similar and those of table 55 are plotted with the symbol o in figure 54. The shape of the model has been distorted in the vicinity of the vertical offset at a distance coordinate 6,000. This "revised" model is entered into file tcip3.lay (table 56a) and subroutine RAYTRACE re-executed (table 56b). Results are shown plotted by the symbol x in figure 55. At first glance, the results of raytrace appear to agree with the original model. However, close inspection shows local differences of approximately 30 time units. Using units of meters and km/sec, this difference would be about 30 milliseconds -- a larger difference than might be expected if the error were due to arrival-time picks.

Subroutine ANGLE (ap,vp,x1,y1,x2,y2,vq,aq,irefl,iflag)

Given an interface defined as a straight line by two points (x1,y1) and (x2,y2); an incident ray whose direction is defined by an angle, ap, with respect to the vertical and a velocity vp; and the velocity vq of the refracted ray; then subroutine ANGLE uses Snell's law to calculate the refraction angle, aq, with respect to the vertical.

Irefl and iflag normally equal zero. Irefl becomes one if the incident ray impinges on the interface at greater than the critical angle and iflag becomes one if the incident ray is defined by an angle whose absolute value exceeds ninety degrees. (In these programs, a ray angle must be defined between plus and minus $\pi/2$.)

Function subprogram ASIN(x)

ASIN(x) returns the value of the arc sin(x) in either the first or fourth quadrant as either a positive or negative angle.

Function subprogram ACOS(x)

ACOS(x) returns the value of the arc cos(x) in either the first or fourth quadrant as a positive or negative angle.

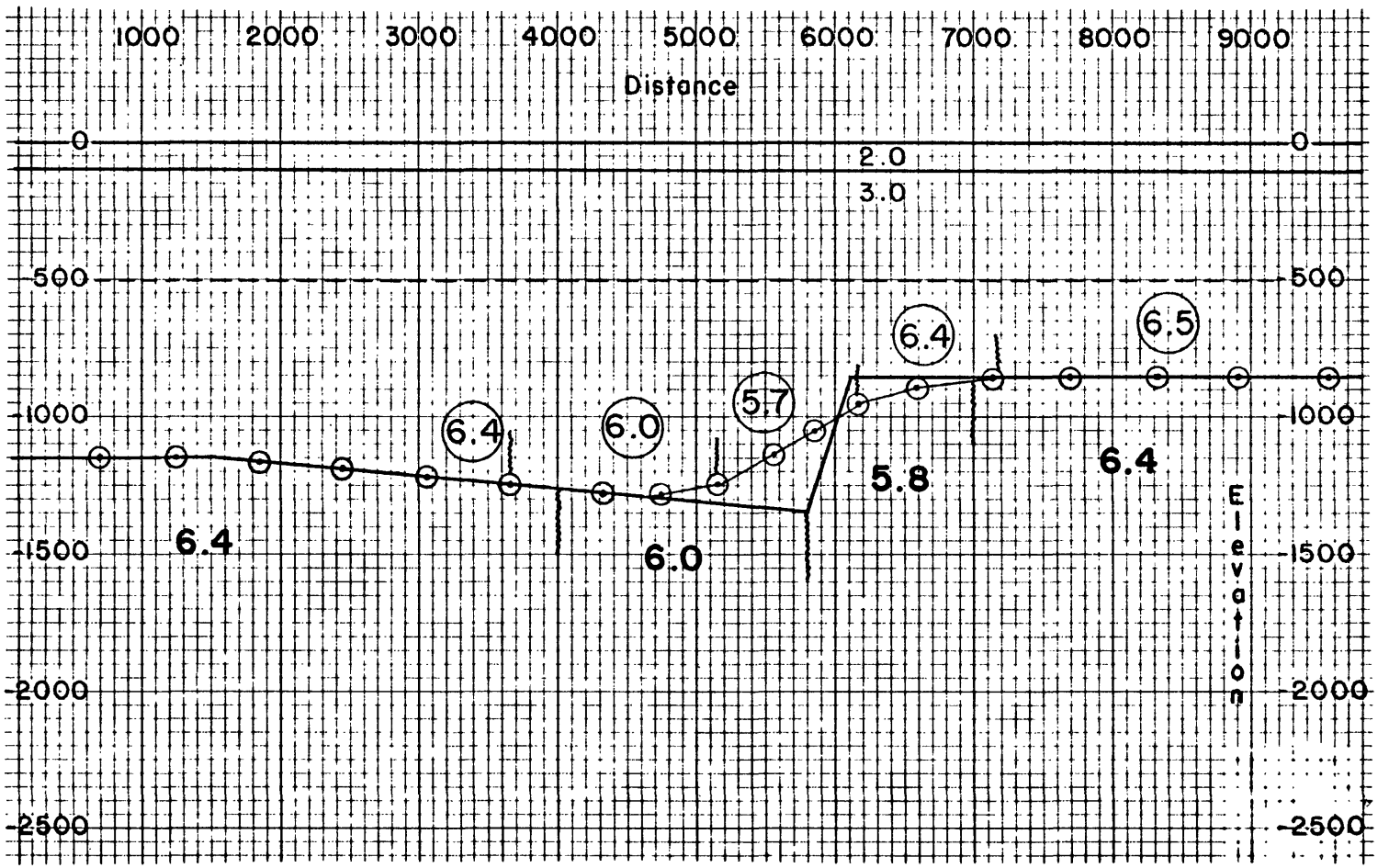

Figure 54.--Fourth example to illustrate execution of subroutine RECIP as described in text.....

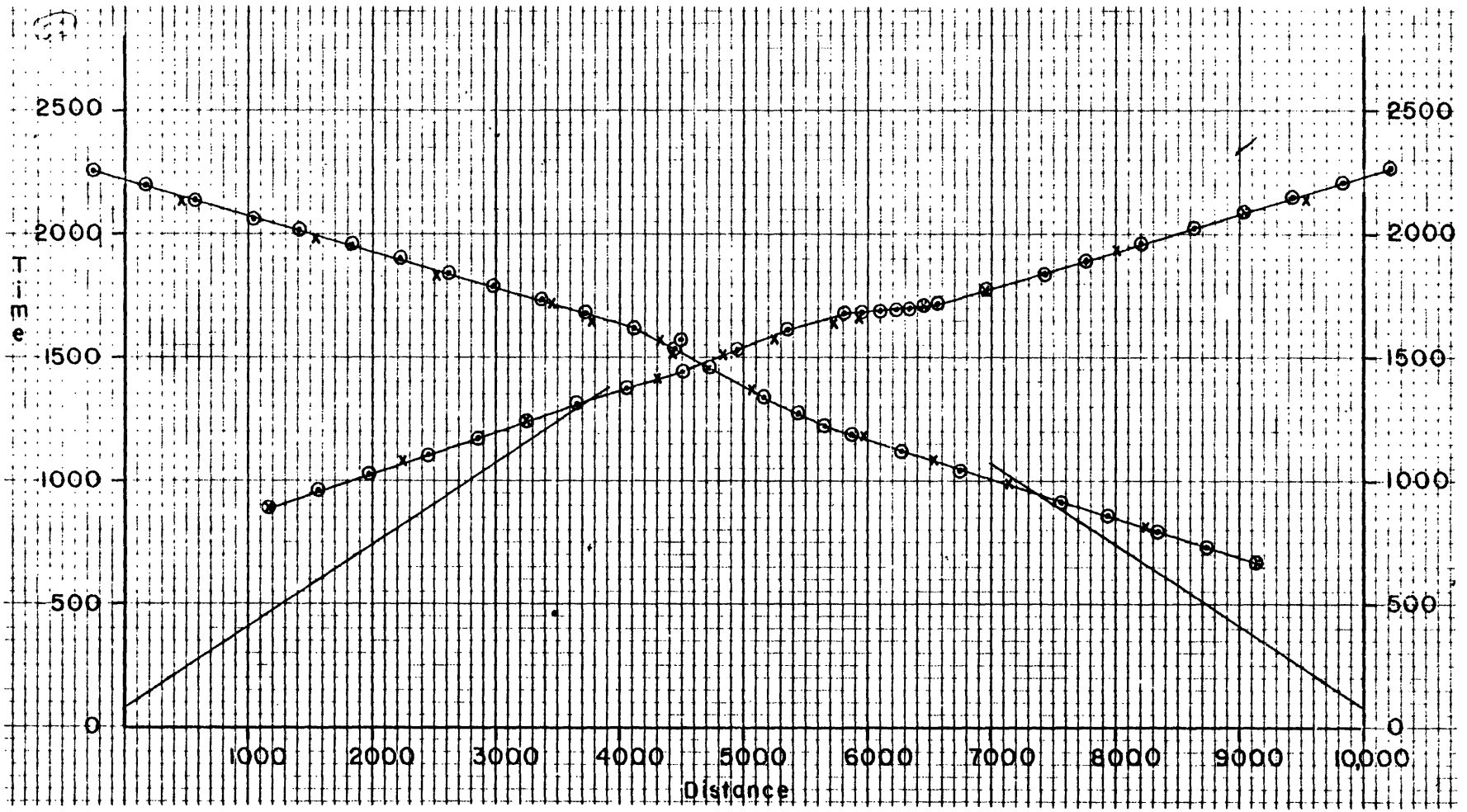

Figure 55.--Reversed arrival-time plots from horizon 3 of Figure 54 as described in the text.....

(a)

```
print tcip3.lay

                    tcip3.lay


 1  2  -90000.00        0.00        2.00
  90000.00        0.00        2.00
 2  2  -90000.00     -100.00        3.00
  90000.00     -100.00        3.00
 3  7  -90000.00    -1150.00        6.40
   1500.00    -1150.00        6.40
   4000.00    -1260.00        6.00
   5800.00    -1350.00        5.80
   6100.00     -850.00        5.80
   7000.00     -850.00        6.40
  90000.00     -850.00        6.40
```

(b)

```
seismc
type input file name: tcip3

input complete from 3 dsk files named     tcip3
type subroutine: raytr
raytrace entry 1                    file=    tcip3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*3 0 0 0 10000 400;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*2 6100 - 6100 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
      3        589.70      -1150.00        448.80        -88.14

do you want to continue? yes or no:yes
arrival time data for shot at x=       0.00 depth=    0.00
  from off horizon   3
 x-refractor coord.    x-surface coord.  arrival time
         589.70            1179.79        897.67
         989.70            1579.75        960.16
        1389.70            1979.75       1022.66
        1789.70            2450.19       1100.34
        2189.70            2860.80       1169.72
        2589.70            3271.28       1239.09
        2989.70            3681.71       1308.46
        3389.70            4092.11       1377.83
        3789.70            4502.50       1447.20
        4189.70            4983.63       1531.25
        4589.70            5396.59       1605.93
        4989.70            5809.53       1680.62
        5389.70            6222.48       1755.30
        5789.70            6635.42       1829.98
        6189.70            6461.45       1706.27
        6589.70            6982.36       1773.32
        6989.70            7428.06       1841.61
        7389.70            7794.41       1894.66
        7789.70            8203.06       1955.79
        8189.70            8608.03       2017.28
        8589.70            9011.13       2079.01
        8989.70            9413.19       2140.89
        9389.70            9814.63       2202.89
        9789.70           10215.67       2264.98
diffraction results from above shot:
diffractions from subsurface point at x=    6100.00
  emergence angle    x-surface coord.   arrival time
         0.00              6100.00        1682.81
       -10.00              5956.10        1687.00
       -20.00              5803.60        1700.21
       -30.00              5631.63        1724.52
       -40.00              5423.25        1764.50
       -50.00              5146.79        1829.90
       -60.00              4730.25        1944.05
       -70.00              3959.03        2177.91
       -80.00              1759.53        2888.78
diffractions from subsurface point at x=    6100.00
  emergence angle    x-surface coord.   arrival time
         0.00              6100.00        1682.81
        10.00              6243.90        1687.00
        20.00              6396.40        1700.21
        30.00              6568.37        1724.52
        40.00              6776.75        1764.50
        50.00              7053.21        1829.90
        60.00              7469.75        1944.05
        70.00              8240.97        2177.91
        80.00             10440.47        2888.78
all done. another shot!
raytrace entry 1                    file=    tcip3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*3 10000 0 10000 0 400;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*2 6100 - 6100 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time      refrac. angle
      3        9569.40       -850.00        335.61        88.14

do you want to continue? yes or no:yes
arrival time data for shot at x=  10000.00 depth=    0.00
  from off horizon   3
 x-refractor coord.    x-surface coord.  arrival time
         9569.39           9138.52        671.26
         9169.39           8738.52        733.76
         8769.39           8338.52        796.26
         8369.39           7938.52        858.76
```

Table 52.--(a) Printout of file tcip3.lay showing the input model of Figure 54.
(b) Execution of subroutine RAYTRACE for horizon 3 of Figure 54.....

Table 52.--Continued

```
        7969.39            7538.52        921.26
        7569.39            7138.52        983.76
        7169.39            6738.52       1046.26
        6769.39            6279.41       1122.21
        6369.39            5879.41       1191.18
        5969.39            2065.33       2294.70
        5569.39            4902.73       1504.16
        5169.39            4512.98       1563.42
        4769.39            4123.23       1622.67
        4369.39            3733.48       1681.93
        3969.39            3388.60       1734.30
        3569.39            2996.95       1790.37
        3169.39            2605.31       1846.43
        2769.39            2213.68       1902.50
        2369.39            1822.04       1958.56
        1969.39            1430.41       2014.62
        1569.39            1038.77       2070.69
        1169.39             580.35       2140.98
         769.39             180.22       2203.44
         369.39            -219.89       2265.91
diffraction results from above shot:
diffractions from subsurface point at x=   6100.00
  emergence angle   x-surface coord.  arrival time
         0.00              6100.00       1192.25
        10.00              6243.90       1196.45
        20.00              6396.40       1209.65
        30.00              6568.37       1233.96
        40.00              6776.75       1273.94
        50.00              7053.21       1339.34
        60.00              7469.75       1453.49
        70.00              8240.97       1687.35
        80.00             10440.47       2398.22
diffractions from subsurface point at x=   6100.00
  emergence angle   x-surface coord.  arrival time
         0.00              6100.00       1192.25
       -10.00              5956.10       1196.45
       -20.00              5803.60       1209.65
       -30.00              5631.63       1233.96
       -40.00              5423.25       1273.94
       -50.00              5146.79       1339.34
       -60.00              4730.25       1453.49
       -70.00              3959.03       1687.35
       -80.00              1759.53       2398.22
all done. another shot!
raytrace entry 1                    file=    tcip3
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*QUIT
```

```
s@iamc

type input file name: tcip3

input complete from 3 dsk files named     tcip3
type subroutine: datain
type file extension either org,com,or lay: org
enter  the word digit   if data digitized.
anything else if not: no
datain entry 1; k@,ng(k),sx(k),sd(k),offset(k)
@1 23 0 0 -1
gx(k,j):1180 1580 1980 2450 2861 3271 3681 4092 4502 4983 5396 5809 6244 6568 69
\c82 7428 7794 8203 8608 9011 9413 9814 10215;
gt(k,j):898 960 1023 1100 1170 1239 1308 1377 1447 1531 1606 1680 1687 1724 1773
\c 1842 1895 1956 2017 2079 2141 2202 2265;
for shot index=     1 xlast=    132065.00
next set of data please. 0; or -n; for return to mainline
datain entry 1; k@,ng(k),sx(k),sd(k),offset(k)
@2 24 10000 0 -1
gx(k,j):-220 180 580 1039 1430 1822 2214 2605 2997 3389 3733 4123 4415 5147 5631
\c 5879 6279 6739 7139 7539 7939 8339 8739 9139;
gt(k,j):2266 2203 2140 2071 2014 1959 1902 1846 1790 1734 1682 1623 1547 1339 12
\c34 1191 1122 1046 984 921 859 796 734 671;
for shot index=     2 xlast=    106816.00
next set of data please. 0; or -n; for return to mainline
datain entry 1; k@,ng(k),sx(k),sd(k),offset(k)
@0;
output of 3 files to dsk complete
datain complete
type subroutine: term
type input filename and extension: tcip3.org

                    tcip3.org


  1 23       0.00       0.00      -1.00
    1180.00     898.00
    1580.00     960.00
    1980.00    1023.00
    2450.00    1100.00
    2861.00    1170.00
    3271.00    1239.00
    3681.00    1308.00
    4092.00    1377.00
    4502.00    1447.00
    4983.00    1531.00
    5396.00    1606.00
    5809.00    1680.00
    6244.00    1687.00
    6568.00    1724.00
    6982.00    1773.00
    7428.00    1842.00
    7794.00    1895.00
    8203.00    1956.00
    8608.00    2017.00
    9011.00    2079.00
    9413.00    2141.00
    9814.00    2202.00
   10215.00    2265.00
  2 24   10000.00       0.00      -1.00
    -220.00    2266.00
     180.00    2203.00
     580.00    2140.00
    1039.00    2071.00
    1430.00    2014.00
    1822.00    1959.00
    2214.00    1902.00
    2605.00    1846.00
    2997.00    1790.00
    3389.00    1734.00
    3733.00    1682.00
    4123.00    1623.00
    4415.00    1547.00
    5147.00    1339.00
    5631.00    1234.00
    5879.00    1191.00
    6279.00    1122.00
    6739.00    1046.00
    7139.00     984.00
    7539.00     921.00
    7939.00     859.00
    8339.00     796.00
    8739.00     734.00
    9139.00     671.00

type subroutine: QUIT
```

Table 53.--Execution of subroutine DATAIN to input the t-x data values of Table 52(b) into file tcip3.org.....

```
seismo

type input file name· tcip3

input complete from 3 dsk files named      tcip3
type subroutine· recip
recip entry 1                    file=    tcip3
id*.curv.delx.<bend1.bend2.iu.myoip.tss.tuhl.tuhr>
*2 lin 300·
recip entry 2· kl*.extl.xminl.xmaxl.delxl.dintl.<nvall.nprinl.nstarl>
*1 org -1000 12000 200 200·
recip entry 2a: optll*.+or-dxll.+or-fnll(i)
*0·
recip entry 2b· optlr.+or-dxlr.+or-fnlr(i)
*0·
recip entry 3· kr*,extr.xminr,xmaxr,delxr.dintr,<nvalr,nprinr,nstarr>
*2 org -1000 12000 200 200·
recip entry 3a· optrl*.+or-dxrl.+or-fnrl(i)
*0·
recip entry 3b· optrr*.+or-dxrr.+or-fnrr(i)
*0·
recip times adding uphole·  left to right. right to left. and average
    2231.2     2231.4     2231.3
do you want to continue? yes or no·yes
print t-x data on horizon    2 ? yes or no:no
The reciprocal time solutions are as follows
          point values          incremental parameters     surface coordinates
      xi       eta velocity velocity     tau    delta-t delta-x x-minus  xplus
                                                              right sp left sp
   676.3   -1150.3      6.4      0.0  462.45     0.00     0.00    113.2   1230.6
   969.6   -1149.8      6.4      6.4  508.31    45.86   293.36    413.2   1528.5
  1259.7   -1151.3      6.4      6.4  553.61    45.30   290.06    713.2   1831.5
  1537.9   -1156.5      6.4      6.4  596.95    43.34   278.28   1013.2   2135.8
  1832.3   -1165.7      6.4      6.4  642.95    46.00   294.55   1313.2   2450.0
  2122.1   -1177.8      6.4      6.4  688.19    45.24   290.03   1613.2   2756.6
  2430.6   -1191.6      6.4      6.4  736.35    48.15   308.80   1913.2   3070.9
  2743.6   -1204.7      6.4      6.4  785.30    48.95   313.27   2213.2   3389.3
  3046.1   -1217.2      6.4      6.4  832.59    47.29   302.74   2513.2   3700.6
  3349.9   -1230.5      6.4      6.4  880.08    47.49   304.17   2813.2   4017.6
  3658.7   -1244.8      6.4      6.4  928.61    48.53   309.14   3113.2   4349.2
  3986.4   -1260.5      6.2      6.3  980.90    52.29   328.01   3413.2   4703.2
  4316.6   -1275.9      6.1      6.1 1034.95    54.06   330.58   3713.2   5070.6
  4828.9   -1275.4      5.4      6.0 1120.94    85.99   512.25   4013.2   5454.9
  5371.7   -1210.8      5.8      5.6 1217.70    96.76   546.70   4313.2   5719.0
  6005.2   -1001.8      5.4      5.7 1335.42   117.71   667.10   4613.2   6247.0
  6187.7    -933.4      5.1      5.1 1373.72    38.31   194.83   4913.2   6463.8
  6085.3    -962.2      5.9     -5.6 1354.73   -18.99  -106.39   5213.2   6340.8
  6198.2    -934.4      6.5      5.9 1374.38    19.65   116.28   5513.2   6479.8
  6381.7    -906.8      6.5      6.5 1403.08    28.71   185.56   5813.2   6708.0
  6589.3    -889.8      6.5      6.6 1434.75    31.66   208.37   6113.2   6955.5
  6868.7    -876.9      6.3      6.4 1478.51    43.76   279.70   6413.2   7252.4
  7137.2    -870.0      6.5      6.4 1520.49    41.98   268.58   6713.2   7517.7
  7423.4    -864.6      6.6      6.5 1564.41    43.92   286.26   7013.2   7800.5
  7718.1    -860.4      6.5      6.6 1609.38    44.97   294.64   7313.2   8101.2
  8014.9    -857.4      6.5      6.5 1654.91    45.53   296.82   7613.2   8400.1
  8314.3    -855.3      6.5      6.5 1701.03    46.12   299.43   7913.2   8705.8
  8614.4    -853.8      6.4      6.5 1747.54    46.51   300.12   8213.2   9006.8
  8911.4    -852.6      6.5      6.5 1793.57    46.03   296.96   8513.2   9301.2
  9212.2    -851.2      6.5      6.5 1840.14    46.58   300.88   8813.2   9602.3
  9513.7    -849.9      6.4      6.4 1887.07    46.93   301.51   9113.2   9911.6
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
   -186.8 to     113.2  and     9113.2 to    9172.2
and for the left shotpoint, in the intervals:
   1147.4 to    1230.6  and     9911.6 to   10181.9
do you wish the xstar points? yes or no:QUIT
```

Table 54.--Execution of subroutine RECIP for the arrival-time data in Table 53(b).....

Table 55.--Execution of subroutine RECIP for the arrival-time data in Table 53(b) with insertion of a phantom horizon in the velocity model.....

```
print tcip3.lay
                    tcip3.lay
  1  2  -90000.00        0.00          2.00
   90000.00          0.00          2.00
  2  2  -90000.00     -100.00          3.00
   90000.00     -100.00          3.00
  3  2  -90000.00     -500.00          3.00
   90000.00     -500.00          3.00
  4  7  -90000.00    -1150.00          6.40
    1500.00    -1150.00          6.40
    4000.00    -1260.00          6.00
    5800.00    -1350.00          5.80
    6100.00     -850.00          5.80
    7000.00     -850.00          6.40
   90000.00     -850.00          6.40


seismc

type input file name: tcip3

input complete from 3 dsk files named      tcip3
type subroutine: recip
recip entry 1                        file=    tcip3
id*,curv,delx,<bend1,bend2,iu,mycip,tss,tuhl,tuhr>
*3 lin 300;
recip entry 2; kl*,extl,xminl,xmaxl,delxl,dintl,<nvall,nprinl,nstarl>
*1 org -1000 12000 200 200;
recip entry 2a; optll*,+or-dxll,+or-fnll(i)
*0;
recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)
*0;
recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,<nvalr,nprinr,nstarr>
*2 org -1000 12000 200 200;
recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)
*0;
recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)
*0;
recip times adding uphole;  left to right, right to left, and average
    2231.2     2231.4     2231.3
do you want to continue? yes or no:yes
print t-x data on horizon     3 ? yes or no:no
The reciprocal time solutions are as follows
          point values          incremental parameters     surface coordinates
       xi        eta velocity velocity     tau     delta-t delta-x x-minus  xplus
                                                                   right sp left sp
    676.1    -1150.3      6.4      0.0  462.43        0.00     0.00    327.6    1020.2
    971.0    -1150.1      6.4      6.4  508.50       46.08   294.84    627.6    1318.3
   1265.7    -1152.0      6.4      6.4  554.52       46.02   294.76    927.6    1621.7
   1553.9    -1157.6      6.4      6.4  599.44       44.92   288.20   1227.6    1926.7
   1851.9    -1167.0      6.4      6.4  646.01       46.57   298.23   1527.6    2238.2
   2150.0    -1179.2      6.4      6.4  692.55       46.54   298.29   1827.6    2548.8
   2456.8    -1192.5      6.4      6.4  740.47       47.92   307.11   2127.6    2862.8
   2765.4    -1205.6      6.4      6.4  788.71       48.24   308.85   2427.6    3178.0
   3070.5    -1218.4      6.4      6.4  836.40       47.69   305.36   2727.6    3492.2
   3375.9    -1231.7      6.4      6.4  884.20       47.80   305.73   3027.6    3809.5
   3684.9    -1246.0      6.4      6.4  932.88       48.68   309.34   3327.6    4137.0
   4008.0    -1261.4      6.2      6.2  984.64       51.76   323.47   3627.6    4480.2
   4336.3    -1275.2      6.0      6.1 1038.70       54.05   328.57   3927.6    4821.0
   4753.4    -1277.4      5.6      5.9 1109.47       70.78   417.12   4227.6    5191.8
   5139.2    -1244.1      5.2      5.8 1176.73       67.26   387.22   4527.6    5430.9
   5575.7    -1151.9      5.3      5.7 1254.56       77.83   446.09   4827.6    5759.8
   5874.5    -1053.5      5.3      5.6 1310.70       56.14   314.59   5127.6    6007.9
   6084.2     -977.0      5.4      5.3 1352.86       42.16   223.24   5427.6    6216.9
   6177.3     -946.6      6.2      5.5 1370.59       17.73    98.01   5727.6    6312.2
   6343.3     -915.6      6.6      6.3 1397.56       26.98   168.81   6027.6    6503.1
   6568.0     -893.5      6.5      6.6 1431.68       34.12   225.79   6327.6    6746.9
   6850.4     -878.5      6.4      6.4 1475.62       43.94   282.75   6627.6    7034.1
   7132.3     -870.3      6.5      6.4 1519.51       43.89   282.02   6927.6    7316.4
   7424.7     -864.8      6.5      6.5 1564.56       45.05   292.46   7227.6    7605.8
   7719.5     -860.6      6.5      6.5 1609.64       45.08   294.83   7527.6    7901.2
   8017.3     -857.5      6.5      6.5 1655.32       45.68   297.86   7827.6    8199.5
   8316.3     -855.4      6.5      6.5 1701.39       46.07   299.02   8127.6    8500.2
   8615.6     -853.8      6.5      6.5 1747.72       46.33   299.25   8427.6    8799.4
   8914.6     -852.6      6.5      6.5 1794.05       46.33   299.00   8727.6    9097.6
   9214.4     -851.2      6.4      6.5 1840.53       46.48   299.80   9027.6    9397.4
   9514.3     -850.1      6.4      6.4 1887.22       46.69   299.98   9327.6    9699.4
for the right shotpoint, no solutions occur along the
reduced t-x graph in the intervals:
     27.6 to     327.6  and     9327.6 to    9385.8
and for the left shotpoint, in the intervals:
    936.6 to    1020.2  and     9699.4 to    9969.7
do you wish the xstar points? yes or no:yes
the path of special points (xstar) from horizon iu to id
left-hand shot-point
   x(iu)  time(iu)     x(id)  time(id)
 10000.0    2231.2    9754.7    2027.7
  1180.0     898.0     936.6     694.7
  2183.9    1056.4    1924:0     850.7
  3187.8    1225.0    2917.8    1017.4
  4191.7    1394.1    3918.1    1186.0
  5195.6    1569.6    4899.8    1357.6
  6199.4    1691.4    6119.3    1507.7
  7203.3    1807.2    6973.4    1606.5
  8207.2    1956.7    7974.4    1755.1
  9211.1    2109.9    8970.7    1907.0
 10215.0    2265.0    9969.7    2061.3
the right-hand shot-point
     0.0    2231.4     247.6    2027.4
  -220.0    2266.0      27.6    2062.0
   819.9    2103.9    1054.3    1902.2
  1859.8    1953.3    2079.2    1753.4
  2899.7    1803.9    3119.4    1604.1
  3939.6    1650.8    4197.3    1442.0
  4979.4    1387.6    5557.2    1112.7
  6019.3    1166.8    6297.2     958.1
  7059.2     996.3    7305.1     792.6
  8099.1     833.8    8345.2     630.0
  9139.0     671.0    9385.8     467.2
type subroutine: QUIT
```

(a)

```
print tcip4.lay
                    tcip4.lay
  1   2  -90000.00        0.00          2.00
  90000.00          0.00          2.00
  2   2  -90000.00     -100.00          3.00
  90000.00     -100.00          3.00
  3   2  -90000.00     -500.00          3.00
  90000.00     -500.00          3.00
  4   9  -90000.00    -1150.00          6.40
    1500.00    -1150.00          6.40
    4000.00    -1260.00          6.00
    4336.00    -1275.00          6.00
    5139.00    -1244.00          5.70
    6084.00     -977.00          6.40
    7132.00     -870.00          6.50
    9514.00     -850.00          6.40
   90000.00     -850.00          6.40
```

(b)

```
seismc
type input file name: tcip4

input complete from 3 dsk files named     tcip4
type subroutine: raytr
raytrace entry 1                        file=    tcip4
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 4 0 0 10000 500;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*1 6100 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time       refrac. angle
     4         593.70      -1150.00        448.80        -88.14

do you want to continue? yes or no:yes
arrival time data for shot at x=      4.00 depth=   0.00
  from off horizon   4
 x-refractor coord.    x-surface coord.   arrival time
         593.70             1183.79         897.67
        1093.70             1683.75         975.78
        1593.70             2248.89        1065.70
        2093.70             2762.27        1152.45
        2593.70             3275.39        1239.16
        3093.70             3788.42        1325.87
        3593.70             4301.41        1412.58
        4093.70             4875.16        1510.87
        4593.70             5240.33        1571.92
        5093.70             5730.16        1648.05
        5593.70             5903.15        1657.17
        6093.70             6466.36        1710.77
        6593.70             6950.85        1770.54
        7093.70             7431.94        1830.44
        7593.70             8009.56        1914.72
        8093.70             8509.27        1989.61
        8593.70             9008.25        2064.61
        9093.70             9506.85        2139.69
        9593.70            10021.97        2217.76
diffraction results from above shot:
diffractions from subsurface point at x=   6100.00
  emergence angle    x-surface coord.   arrival time
        0.00              6100.00         1687.83
       10.00              6266.01         1692.67
       20.00              6442.03         1707.91
       30.00              6640.75         1736.00
       40.00              6881.95         1782.28
       50.00              7202.62         1858.14
       60.00              7686.89         1990.85
       70.00              8585.41         2263.32
       80.00             11151.45         3092.66
all done. another shot!
raytrace entry 1                        file=    tcip4
layd*,shotx,shotd,xstart,xend,delx,<bend,tim0,beta1,dbeta,alpha1,dalpha>
*4 10000 0 10000 0 500;
raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),direc(2),.....
*1 6100 + ;
critical angle exceeded in subroutine angle
normal exit for raydown
the near-critical subsurface paramaters beneath the shot-point are
  horizon       x-coord.     elevation       time       refrac. angle
     4         9569.40      -850.00        335.61        88.14

do you want to continue? yes or no:yes
arrival time data for shot at x=  10000.00 depth=   0.00
  from off horizon   4
 x-refractor coord.    x-surface coord.   arrival time
         9569.39             9138.52         671.26
         9069.39             8636.28         749.77
         8569.39             8134.05         828.28
         8069.39             7631.82         906.79
         7569.39             7129.59         985.30
         7069.39             6513.29        1085.14
         6569.39             5978.38        1184.01
         6069.39             5087.10        1360.87
         5569.39             4428.82        1520.36
         5069.39             4309.01        1559.90
         4569.39             3796.85        1650.90
         4069.39             3433.12        1710.86
         3569.39             2996.91        1773.73
         3069.39             2507.37        1843.81
         2569.39             2017.83        1913.89
         2069.39             1528.29        1983.98
         1569.39             1038.75        2054.06
         1069.39              480.17        2139.99
          569.39              -20.01        2218.10
           69.39             -520.13        2296.20
diffraction results from above shot:
diffractions from subsurface point at x=   6100.00
  emergence angle    x-surface coord.   arrival time
        0.00              6100.00         1214.12
      -10.00              5933.99         1218.96
      -20.00              5757.97         1234.20
      -30.00              5559.25         1262.29
      -40.00              5318.05         1308.57
      -50.00              4997.38         1384.43
      -60.00              4513.11         1517.14
      -70.00              3614.59         1789.61
      -80.00              1048.55         2618.95
all done. another shot!
QUIT
```

Table 56.--(a) Printout of file tcip4.lay containing revised displacement model calculated in Table 55.
(b) Execution of subroutine RAYTRACE for horizon 4 in file tcip4.lay.....

Subroutine COMP (x,i,y,x1,y1,x2,y2,v,j,iflag)

Given a distance coordinate x and a horizon index i, subroutine COMP determines the following compartment parameters:

j-index of the compartment containing x,
y-the elevation coordinate on the ith horizon at the distance x,
(x1,y1)-the upper left compartment coordinates,
(x2,y2)-the upper right compartment coordinates,
v-the compartment velocity.

The variable iflag is normally zero. It becomes one if x lies outside the endpoints of the ith horizon, if x2 is less than or equal to x1 (e.g., the successive compartment boundaries are not of increasing value), or if the index of the compartment containing x exceeds 50.

Subroutine CSECT (xp,yp,ap,vp,xpp,ypp,app,vpp,tauc,jrefl,iflag,jinc,ibend)

Subroutine CSECT calculates several parameters related to an incident ray and a compartment boundary which it intersects.

Given an incident ray defined by a starting point with coordinates (xp,yp), an angle ap, with respect to the vertical, and a velocity vp; a vertical compartment boundary (interface) defined by a distance coordinate, xpp; and the velocity, vpp, on the opposite side of the compartment interface; then subroutine CSECT determines the following:

ypp - the elevation coordinate of the point of intersection between the incident ray and the compartment interface, resulting in an intersection point with coordinates (xpp, ypp);

tauc - the traveltime between (xp, yp) and (xpp, ypp) at velocity vp;

app - the angle with respect to the vertical which the ray assumes after impinging on the compartment interface.

The value which app assumes depends on whether or not ibend equals zero. If ibend equals "no", reflection or refraction is not permitted and the ray passes through the compartment interface with app=ap. If ibend is "yes", Snell's law is applied at the interface and the computed value for app is the angle of refraction with respect to the vertical. However, if incidence is beyond the critical angle, reflection is assumed and app=-ap.

The variable jinc may assume the values -1, 0 or +1. Jinc = -1 implies that the incident ray passed through the compartment boundary from right to left requiring that the new compartment index be decreased by one. Jinc = +1 implies propagation through the compartment boundary from left to right with the new compartment increased by one, and jinc = 0 indicates that reflection occurred and the compartment index remains unchanged.

Jrefl and iflag are normally zero. Jrefl becomes one if a reflection occurred at the compartment boundary. Iflag becomes one if the incident ray is vertical and hence an intersection with a vertical compartment boundary is impossible or if a compartment velocity is less than or equals zero.

Subroutine EQSPACE (x,t,nobs,dint,idim)

Subroutine EQSPACE takes t-x arrays (x(i),t(i),i=1,nobs) with unequally spaced x values and through a linear interpolation process, replaces them by new arrays in which the x values are equally spaced with interval approximately equal to dint. The initial arrays must be input with the values of x arranged in increasing order.

Idim normally equals zero but becomes one if either the initial arrays (x(i),t(i)) contain less than two points or if the program dimensions are exceeded.

Subroutine HSECT (xp,yp,ap,vp,x1,y1,x2,y2,xq,yq,tauh,iflag)

Given an interface defined as a straight line by two points (x1,y1) and (x2,y2); an incident ray defined by a starting point (xp,yp), an angle, ap, with respect to the vertical, and a velocity, vp; then subroutine HSECT computes the point of intersection (xq,yq) of the incident ray with the interface, and the traveltime, tauh, between the points (xp,yp) and (xq,yq).

Iflag normally equals zero. It becomes one if the incident ray is parallel to the interface and hence no intersection is possible.

Subroutine INTERP (x,t,nobs,xp,time,iterp)

Given the t-x arrays (x(i),t(i),i=1,nobs) and the distance coordinate xp which falls within the range of the x(i), subroutine INTERP interpolates linearly between the two successive values of x(i) which straddle xp and calculates a value for the variable time. The variable iterp is normally zero but becomes one if xp lies outside the range of the x(i).

Subroutine LINDT (nobs,x,t,xp,nval,time,deriv)

Given the t-x arrays (x(i),t(i),i=1,nobs), subroutine LINDT calculates the arrival-time time, and the derivative deriv, which satisfies the t-x plot at the distance coordinate xp in a least squares straight line sense. The variable nval represents the number of points comprising a subset of (x(i),t(i)) which are used for calculating the least squares straight line (see fig. 22). This subset of (x(i),t(i)) is selected such that xp lies approximately in the center of it. If xp lies at or near either end of the array, then nval of the end points are used.

Subroutine LSLIN (npts,x,t,a1,a2)

Subroutine LSLIN calculates the coefficients a1 and a2 for a least squares straight line through the points (x(i),t(i),i=1,npts). The equation of the line is t=a1 + (a2)x.

Subroutine MAFINA (filnam,filorg,filcom,fillay,ierror)

Subroutine MAFINA creates the file names filnam.org, filnam.com, and filnam.lay from the input root name filnam.

Subroutine RANGE (f,a,b,i)

Given two points with distance coordinates a and b, and a third point with distance coordinate f, subroutine RANGE assigns the value of zero to i if f lies within the range of a and b and one if it lies outside this range.

Subroutine RATHRU (x,id,a,tau,nrays,bend)
Common /block1/xp,yp,ap,vp,taup

Given a ray defined by a starting point with distance coordinate x, on horizon id; and initial angle, a, with respect to the vertical; and a starting time, tau; then subroutine RATHRU traces the ray down through the model beyond its boundaries to a distance coordinate of plus or minus infinity (ten million). It is assumed that no additional horizons occur below id, and hence only compartment boundaries can be intersected.

Because compartment boundaries will be encountered, the submerging ray consists of a series of ray segments. The variable nrays counts the number of these segments. The value of bend (either yes or no) determines whether Snell's law is applied at compartment boundaries (see subroutine CSECT).

The parameters for the computed ray segments are stored in the arrays of common block 1. The coordinates (xp(i), yp(i) i=1,nrays) mark the starting point of each ray segment. The variables ap(i) are their submergent angles with the vertical, vp(i) their respective ray propagation velocities and taup(i), their traveltimes.

Subroutine RAYD (xp,ip,id,beta,xd,yd,taut,ad,vd,jd,iflag,irefl,iq,bend)
and Subroutine RAYD1 (xp,ip,id,beta,xd,yd,taut,iflag,bend)

Given a ray defined by a starting point with distance coordinate, xp, on horizon ip; an initial angle, beta, with respect to the vertical; a starting time, taut; and an ending horizon, id, where id>ip; then subroutine RAYD calculates the following parameters:

(xd,yd) - the coordinates of the point at which the ray intersects horizon id.

ad - the angle with respect to the vertical at which the ray exits beneath horizon id.

vd - the velocity at which the ray exits horizon id.

jd - the compartment index at (xd,yd).

If for any reason, subroutine RAYD is halted before completion to horizon id, the variable iq shows the last horizon index encountered by the submergent ray.

Iflag and irefl are normally zero. Iflag becomes one if any of the called subroutines COMP, HSECT, CSECT or ANGLE failed during execution. Irefl becomes one if the ray failed to propagate through a horizon because the angle of incidence exceeded the critical angle.

The variable bend is either yes or no, depending on whether or not Snell's law is to be applied at compartment boundaries (see subroutine CSECT).

Subroutine RAYD1 is a modified version of RAYD for which the submergent ray is traced through only one layer (e.g. id=ip+1) and the ray parameters ad,vd,jd,irefl and iq are not determined.

Subroutine RAYUP (xp,ip,iu,xs,ys,xu,yu,taut,au,vu,ju,iflag,bend,istart)

Subroutine RAYUP traces an emerging ray upwards through a compartmented layered velocity model from horizon ip to horizon iu (ip>iu). The subroutine has two options, given by bend and istart. If bend equals no, the upgoing ray is not reflected or refracted upon encountering compartment boundaries; it simply crosses the boundary with no change in directions. If bend equals yes, the subroutine considers refraction and possible reflection at the vertical compartment boundaries. Option istart considers two possible initial conditions. If istart equals zero, the ray begins at the point (xs,ys) beneath the ipth horizon and intersects this horizon at a point with distance coordinate xp, thus providing an initial incident direction. If istart equals one, the variable xs represents an angle of refraction beginning at xp and extending upwards toward horizon (ip-1). In this case, the variable ys is ignored. The point represented by (xu,yu) is the computed point of emergence of the ray on the iuth horizon. The variable ju is the compartment index containing (xu,yu), au is the emergent angle with respect to the vertical and vu the velocity in this compartment. The initial value of taut is the travel-time at the starting point of the ray and its final value is that at the point (xu,yu).

The value for iflag is generally zero. However, it becomes one if certain error conditions described in the RAYUP condition statements (Appendix A) occur during execution of the subroutine.

Subroutine SELECT (xmin,xmax,ext,k,npts,t,x,jwatch,iflag)

Subroutine SELECT selects a subset of t-x data points in the data files (e.g. filnam.org or filnam.com) which fall within the range xmin to xmax and puts them into the new arrays (x(i),t(i),i=1,npts). The variable k is the shot index within the file with extension ext. Xmin and xmax are changed to agree with the precise limits of the new data set. Jwatch and iflag are normally zero. Jwatch becomes one if the new set contains less than two data points and iflag becomes one if the initial arrays from which the subset is taken are not in order of increasing x.

Subroutine SPLINDT (m,x,y,a,b,c,xx,yy,deriv)

Given cubic spline coefficients a,b,c, and m observation data arrays x,y, SPLINDT evaluates the piecewise cubic spline ordinate yy, and its derivative deriv, at the abscissa xx, where xx is in the closed interval (x(i),x(m)).

Subroutine WAVED (x,t,nobs,iu,id,delx,dint,nprint,xstar,nstar,curv,nval,
tstar,xprim,tprim,idim,isplin,bend)

A set of t-x data points (x(i),t(i),i=1,nobs) defined on horizon iu are reduced to a lower horizon id.

The initial data points (x(i),t(i),i=1,nobs) are first redefined through subroutine EQSPACE as a set of points equally spaced along the x axis at intervals dint. These redefined points are then approximated or fit by either a cubic spline function (if curv="spl") or a series of overlapping least square straight line segments (curv="lin") each approximating nval of the t-x data points. The approximating curves are sampled at constant intervals delx, and through the use of subroutine RAYD1, are reduced through one layer, defining a new set of t-x data points on horizon iu+1. They are placed in the arrays (x(i),t(i),i=1,nobs), where nobs may have been changed from its initial value. With the t-x data now defined on horizon iu+1, the process is repeated until the data are defined in the arrays (x(i),t(i)) on horizon id.

An array of given distance coordinates (xstar(i), i=1,nstar) which lie within the range of x(i) have their corresponding arrival-times tstar(i) calculated either by the spline function or the straight line approximations, on horizon iu. The arrays (xstar(i),tstar(i)) then define a subset of t-x points on the unreduced t-x graph. These "star" points are traced through the model separately from horizon iu to horizon id and their final reduced values put into the array (xprim(i),tprim(i)). The user thus has at his disposal, two sets of arrays (xstar(i),tstar(i)) and (xprim(i),tprim(i),i=1,nstar) which define corresponding t-x points on both the upper horizon iu, and the lower horizon id. These arrays are available for later optional printout in the calling subroutine because it is often desirable to know the starting and ending points of rays within the model and this option provides that choice.

If the variable nprint equals any value other than zero, each of the newly calculated arrays (x(i),t(i)) are printed out as the t-x graph is reduced within the model. For nprint=0, this step is omitted.

If the variable bend=yes, the submerging rays are refracted or reflected at compartment boundaries. For ibend=no, rays propagate across compartment boundaries without being refracted.

The variables idim and isplin are normally zero. Idim becomes one if the program dimensions for the arrays (x(i),t(i)) are exceeded and isplin becomes one if the subroutine for calculating cubic splines fails.

During execution of subroutine WAVED, certain conditions may arise which prevent a given ray from successfully propagating from horizon iu to id. Normally, these conditions will not stop the program; instead the program simply ignores that ray and proceeds to the next one. Some conditions involving dimensions or failure of subroutine SPLIN1 will cause program execution to fail.

Subroutine XISECT (xq,yq,aq,tq,vq,v2,xis,etas,xie,etae,taue,ib)

Subroutine XISECT calculates a point (xie,etae) (fig. 56) on the refracting horizon given the critical reflection subsurface point (xis,etas) on that horizon, a propagation velocity of that horizon vq, a point (xq,yq) on an emergent ray with emergence angle aq and velocity v2, and the travel-time tq between (xis,etas) and (xq,yq). The subroutine finds a ray (dashed in fig. 56) which intersects the emergent ray, resulting in the point (xie,etae) such that the combined travel-time along the two rays equals tq. Also calculated is the travel-time taue between (xis,etas) and (xie,etae). The variable ib is normally zero but changes value if an imaginary solution is found.

the equations employed are as follows:

$$etae = \frac{b - (b^2 - 4ac)}{2a}$$

$$xie = xq - (\tan aq)(yp-etae)$$

$$taue = tq - (yp-etae)/(vq)(\cos aq)$$

where

$$a = 1+(\tan aq)^2 - (v2)^2/(vq)^2(\cos aq)^2$$

$$b = 2(etas) + 2m(\tan aq) - 2(u)\ (vs)^2/(vq)(\cos aq)$$

$$c = (etas)^2 + m^2 - (vs)^2(u)^2$$

where

$$m = -(xq-(yq)(\tan aq) - xis)$$

$$u = yp/(vq)(\cos aq) - tq$$

Subroutine XTEND (x,t,nobs,nopt,dx,fn,ioptl,iopt3,idim)

Subroutine XTEND affixes one or more points, either to the beginning or the end of the t-x arrays (x(i),t(i),i=1,nobs). Hence the arrays are expanded. Four options exist for the way in which points are added, specified by the value of nopt.

If nopt equals zero (Case I), no points are added.

If nopt equals one (CaseII), between one and five points, specified by values of dx, are added. The values are in the array fn(i). If fn(i) is less than x(1), the points are affixed to the beginning of the arrays (x(i),t(i)). Otherwise, they are affixed to the end.

If nopt equals two (Case III), a single point is added to either end of the arrays (x(i),t(i)) using a velocity criteria. The value of dx specifies the x value of the new point by adding dx to either x(1) if dx is negative, or to x(nobs) if dx is positive. The t value of the new point is specified from fn(1) which in this case is a velocity value. It is obtained geometrically by constructing a line through either x(1) or x(nobs) with the velocity specified and terminating it at the x value of the new point.

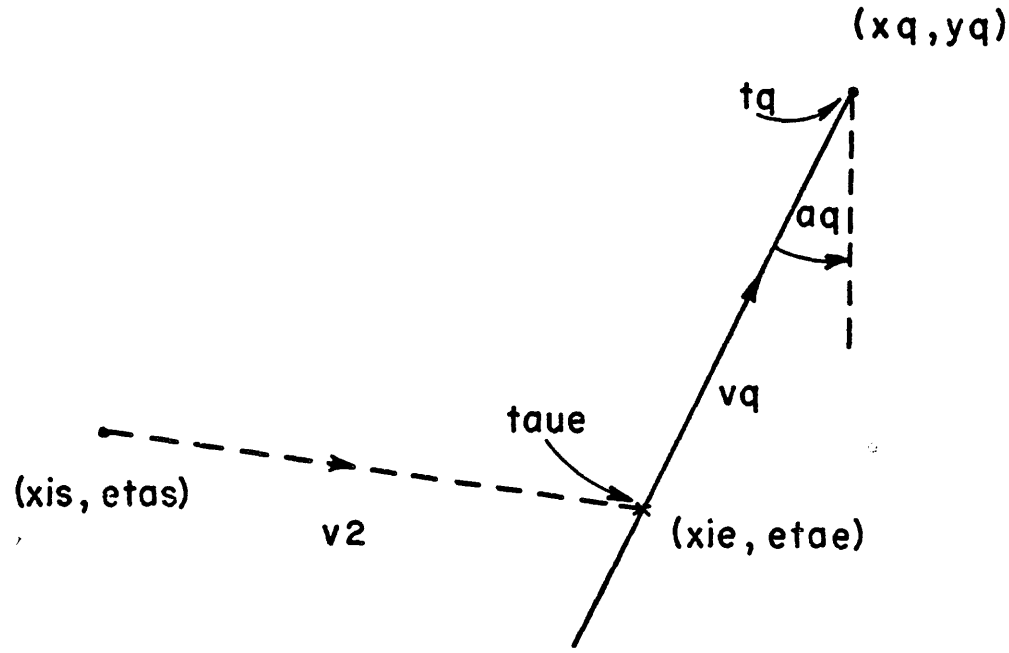

Figure 56.--Sketch showing parameters used in subroutine XISECT.....

If nopt equals three (Case IV), a single point is added to either end of the arrays (x(i),t(i)) according to a least squares criteria. The value of dx again specifies the x value of the new point as in Case III above. The t value of the new point is specified from fn(1) which is simply the number of end points in x(i) to be used for a least squares straight line approximation. Once the coefficients are obtained, the value of the new x coordinate is substituted to obtain the new t value.

ACKNOWLEDGMENTS

This project would not have been undertaken without the encouragement of M. F. Kane, under whose direction most of the work was accomplished. We gratefully acknowledge him for this support. In addition, we thank R. H. Godson, of the U.S. Geological Survey, for his continued interest in the project and valuable assistance in programming.

APPENDIX A

This section contains information which is frequently referred to during program execution. For subroutines requiring terminal input, it contains an Entry List, which defines each of the variables in their order of input. In addition, forms called Input Forms are included for some of the subroutines to facilitate execution and help avoid errors. Finally, condition statements are also included for the subroutines requiring them. During execution errors may occur. These may range from obvious input errors which the programs are designed to detect, to subtle errors in the velocity model and t-x data which do not permit rays to propagate through the model. Instead of printing out error messages, the program indexes a condition statement and refers it to a subroutine, which the user can then look up in this Appendix. Some conditions halt execution completely. Others, however, are related to a single ray. When this occurs, the program simply stops execution on that particular ray and skips on to the next one. The programs contain hundreds of condition statements, most of which seldom or never occur. Some, on the other hand, have become very familiar nemeses.

ACROSS Entry List

Entry 1: Type 4 to 43 values in list directed (v) format.

*1. nv2 The number of velocity values to be used for the (id+1)th horizon. Normally nv2=1. However, if the velocity is believed to vary laterally, more than one value may be used. nv2$\leq$20.

2. (xv2(i),i=1,nv2) The distance coordinates, in increasing value, corresponding to the locations where the velocity of the refracting horizon changes laterally (analogous to the distance coordinate of a compartment boundary). xv2(1) may have any value, such as zero, because the program automatically assigns it a value of minus infinity.

*An asterisk appearing next to a variable name in the input entry form means that (0; return), returns execution to the mainline.

3. (v2(i),i=1,nv2) The velocity values specified for each of the distance coordinates in item 2 above (analogous to a compartment velocity).

4. ±delx The distance interval at which successive depth calculations are attempted. A negative value means that the arrival-times to be used lie to the left of the shotpoint. Positive means to the right. The absolute value of delx must be small enough so that less than 101 calculations will be made.

5. nval (optional) The number of successive points selected from the t-x arrays during execution of subroutines LINDT for approximating these data by a least squares straight line (see fig. 22). The interval between successive points is dint (see subroutine CRIT entry form). Default=value of nval used in subroutine CRIT.

ACROSS Conditions

Condition 1: Input error. Either nv2>21, delx=0, nval<2, the xv2(i) are not in order of increasing value, or the absolute value of delx is too small resulting in over 100 depth calculations. (Statements 72-1, 70+3, or 73+3)

Condition 2: Subroutine COMP failed at the distance coordinate xa indicated. Statement 78+1)

Condition 3: The slope of the arrival-time curve at the distance coordinate xa indicated, is too steep to permit ray to propagate across the horizon indicated. The apparent velocity of the curve at that point is also indicated. (Statement 43+1)

Condition 4: The submerging ray at distance coordinate xa is propagating upward instead of downward. (Statement 79+4)

Condition 5: In projecting the ray originating at distance coordinate xa on horizon id to infinity, it passes above the critical subsurface point instead of below it. Therefore, a valid intersection can not be determined. (Statement 21-1 or 22+2)

Condition 6: No subsurface point for the ray emerging at the distance coordinate xa on horizon id could be found which satisfies the arrival-time curve at that point. Problem could be that horizon id is too deep at xa, the refractor velocity is too low, or the apparent velocity of the t-x curve is too high at xa. (Statement 41-1)

BUILD Entry List

Entry 1: Type one data string in list directed (v) format

*1. k The shot index desired for the combined data.

*Type 0; for exit via subroutine OUTPUT.

BUILD input form

File name_____

k	control set k1	ext1 org/com	xmin1	xmax1	subsidiary set k2	ext2 org/com	smin2	xmax2	opt(1) opt(2)	xsma1 add	xlarg

2. kl The shot index for the control data.

note: The shotpoint distance coordinate, shot depth and offset for the combined data are those of the control data.

3. ext1 The extension of the control data--either org or com.
4. xmin1 A value equal to or slightly less than the distance coordinate of the first point used.
5. xmax1 A value equal to or slightly greater than the distance coordinate of the last point used.

note: xmin1<xmax1. If xmin1=xmax1, no points of the control data are used and option opt=2 must be executed.

6. k2 The shot index for the subsidiary data. A constant is added to these data to line them up with the control data.

7. ext2 The extension of the subsidiary data--either org or com.

8. xmin2 A value equal to or slightly less than the distance coordinate of the first point used.

9. xmax2 A value equal to or slightly greater than the distance coordinate of the last point used.
note: xmin2<xmax2.

10. opt The option to be used, either=1 or =2. If opt=1, the program determines the constant added to the subsidiary data. If opt=2, the user selects the constant.
If opt=1:

11. xsmal xsmal and xlarg (xsmal<xlarg) define a distance interval within which the values of the control and subsidiary data are compared

12. xlarg for determining the constant (add) to be added to the subsidiary data
If opt=2:

13. add The constant to be added to the subsidiary data.

BUILD Conditions

Condition 1: Input data wrong. Either xmin1>xmax1, xmin2>xmax2, opt<1, opt>2, ext1 or ext2 misspelled, or k1 or k2 out of range. (Statement 30)

Condition 2: Subroutine SELECT failed. Either the subset selected for combining contained no points or data point distances are not of successively increasing value. (Statement 4 or 100)

Condition 3: The combined array has over 200 points which exceeds program dimensions. (Statement 7+2).

Condition 4: Subroutine SELECT failed. Either no points were found in the interval between xsmal and xlarg for the control or subsidiary data or data point distances are not of successively increasing value. (Statement 33+4 or 19)

Condition 5: The interval between xsmal and xlarg was such that only one point was selected from both the control and subsidiary data for determining the constant to be added. The value calculated for add is the last one of the five printed. This value is only valid if the x coordinates of both distances (the first and third value printed) are approximately equal. The second and fourth values are the corresponding arrival-times and add equals the difference between these two. If this value for add is satisfactory, it may be used by exercising the option opt=2.

Condition 6: Input data incorrect. xlarg$\leq$xsmal (Statement 3).

CRIT Entry List

Entry 1: Type 10 to 16 values in list directed (v) format

*1. k The shot number index for the t-x data set containing the arrivals to be calculated.

2. ext Either arg or com. The extension of the file name containing the t-x data.

3. curv Either lin or spl. The value of curv determines whether the t-x data are to be approximated by a series of straight line segments or by a cubic spline.

4. id The index of the deepest known horizon. id is one less than the index of the horizon being calculated. $1 \leq id \leq 10$.

5. xmin The t-x arrays may contain points from horizons other than the one being calculated. xmin is a distance coordinate equal to or less than the lower bound of the distance coordinates of the t-x data being used (but not to include unwanted points).

6. xmax A distance coordinate equal to or greater than the upper bound of the distance coordinates of the t-x data being used. Hence, xmin and xmax bracket the data. xmin$\leq$xmax. If xmin=xmax, the program will not search the data set for t-x values and artificial data must be input.

7. v2 The velocity of the layer being calculated at the critical reflection point.

8. ±delx The distance interval at which successive depth calculations are attempted. A negative value means that the arrival-times to be used lie to the left of the shotpoint. Positive means to the right. The absolute value of delx used should generally not exceed the interval between successive geophone groups. (If an answer is not obtained, for some unknown reason, reducing the absolute value of delx often solves the problem.)

9. delxl The increment in distance at which successive points are selected
on the t-x graph, during execution of subroutine WAVED, to reduce it to successively lower horizons. The smaller the value of delxl, the denser the arrays of arrival-times available for sampling by subroutine EQSPACE. delxl should generally be somewhat less than the absolute value of delx, large enough so that less than 400 points will be sampled between xmin and xmax and greater than zero.

10. dint The sampling interval used during execution of subroutine EQSPACE
for creating t-x data arrays at equally spaced distance intervals. Generally dint=delx and is subject to the same conditions.

11. nval (optional) The number of points used for fitting the t-x arrays in the least square straight line segments (see fig. 22) during execution of subroutine LINDT. LINDT is executed after EQSPACE and hence the interval between successive points is dint. The value of nval should be larger for scattered t-x data, and equal to 2 for "perfect" data. Nval$\geq$2, but less than the number of points in the t-x arrays after execution of EQSPACE. Default =3.

12. bend1 (optional) Either yes or no, depending on whether or not rays will be subject to refraction and reflection at compartment boundaries between horizons 1 and id. Default is no.

13. bend2 (optional) Either yes or no, depending on whether or not rays will be subject to refraction and reflection at compartment boundaries below horizon id. Default is no.

14. nprin (optional) Either 0 or 1. If the intermediate t-x data on horizons 2 to id are to be printed out during execution of WAVED, set nprin=1. Default=0.

15. nstar (optional) The number of "star" points to be calculated within the interval of the t-x data sets. The distance coordinates (xstar(i),i=1,nstar) of these points are equally spaced within the interval from xmin to xmax, which has been expanded from items 5 and 6 above, to include extension of the data (see entries 1a and 1b). Subroutine WAVED calculates their arrival-times (tstar(i)) on horizion 1 and traces the rays from these points, through the model to points (xprim(i),tprim(i),i=1,nstar) on horizon id. The "star" point values may be printed on command. 1 <nstar$\leq$30. Default=10.

16. dell1 (optional The distance which a submerging ray from horizon id is incremented each time a new trial calculation is made. Default=|delx|.

Entry 1a. Parameters for extending t-x data arrays to the left. Type 1 to 12 values in list directed (v) format. If the shotpoint lies to the left of the t-x data used, they must be extended at least sufficiently far to include the shotpoint.

*1. optl1 The index specifying the option to be used. If optl1<0 or optl1>3, control is returned to mainline
optl1=0 Data are not extended to the left.
optl1=1 Data are extended and to consist of specific points specified by the parameters in items 2 and 3 below.
optl1=2 Data are extended along a straight line which intersects the first point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.
optl1=3 Data are extended along a least squares straight line constructed through the t-x data points as specified in items 2 and 3 below.

2. ±dxl1
if optl1=1 The number of t-x data points to be added to the left of the original t-x arrays. $1 \leq dxl1 \leq 5$
if optl1=2 The distance which the data are to be extended to the left of the first distance coordinate of the t-x arrays dxl1<0.
if optl1=3 Same as for optl1=2.

3. ±fnl1(i)
if optl1=1 (fnl1(i),i=1,2dxl1) The distance coordinates, then followed by the arrival-times for the number of points specified by item 2 above. The distance coordinates must be of increasing value and all be less than the first t-x data point of the original array.
if optl1=2 ±fnl1(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to increase with increasing distance, then fnl1(1)>0. Otherwise fnl1(1)<0.
if optl1=3 fnl1(1) A value specifying the number of left-most points of the original t-x arrays to be used for a least squares straight line approximation. The extended data will fall on this line and the arrival-times of the first point of the original arrays will also be changed to fall on this line.

Entry 1b. Parameters for extending t-x data arrays to the right. Type 1 to 12 values in list directed (v) format. If the shotpoint lies to the right of the t-x data used, they must be extended at least sufficiently far to include the shotpoint.

*1. optlr The index specifying the option to be used. If optlr<0 or optlr>3, control is returned to the mainline.

optlr=0 Data are not extended to the right.

optlr=1 Data are extended to consist of specific points specified by parameters in items 2 and 3 below.

optlr=2 Data are extended along a straight line which intersects the last point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.

optlr=3 Data are extended along a least squares straight line constructed through the t-x data points as specified in items 2 and 3 below.

2. ±dxlr

if optlr=1 The number of t-x data points to be added to the right of the original t-x arrays. $1 \leq dxlr \leq 5$

if optlr=2 The distance which the data are to be extended to the right of the last distance coordinate of the t-x arrays. dxlr>0

if optlr=3 Same as for optlr=2.

3. ±fnlr(i)

if optlr=1 (fnlr(i),i=1,2dxlr) The distance coordinates then followed by the arrival-times for the number of points specified in item 2 above. The distance coordinates must be of increasing value and all be greater than the last t-x data point of the original array.

if optlr=2 ±fnlr(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to decrease with increasing distance, then fnlr(1)<0. Otherwise fnll(1)>0.

if optlr=3 fnlr(1) A value specifying the number of right-most points of the original t-x arrays to be used for least squares straight line approximation. The extended data will fall on this line and the arrival-time of the last point of the original arrays will also be changed to fall on this line.

Entry 2. To be used only if artificial arrival-time data will be used; e.g., xmin=xmax in entry 1. (note that artificial data may be extended under entries 1a and 1b).

Type (2nobs+1) values in list directed (v) format.

*1. nobs The number of artificial data points to be used.

2. (x(i),i=1,nobs) The distance coordinates of the artificial data points in increasing value.

3. (t(i),i=1,nobs) The corresponding arrival-times.

CRIT Conditions Statement

Condition 1: Error in input file. Either an alphanumeric word is misspelled, id<1 or id>10, delx=0, v2<0, nval<2, k out of range, gxtry is on the wrong side of the shotpoint, nstar<1 or nstar>30. (Statements 4-1 or 4+2)

CRIT AND ACROSS input form File Name_____

Entry 1 for CRIT

	k	ext org/com	curv lin/spl	id	xmin	xmax	v2	±delx	delxl	dint
1.										
2.										
3.										
4.										
5.										

Entry 1 continued (optional)

	(nval	bendl yes/no	bend2 yes/no	nprin 0/1	nstar	delll)
1.						
2.						
3.						
4.						
5.						

Entry 1a for CRIT extend data to left

	optll	±dxll	±fnll(i)
1.			
2.			
3.			
4.			
5.			

Entry 1b for CRIT extend data to right

	optlr	±dxlr	±fnlr(i)
1.			
2.			
3.			
4.			
5.			

Entry 2 for CRIT artificial data

	nobs	x(i), i = 1, nobs	t(i) i = 1, nobs
1.			
2.			
3.			
4.			
5.			

Entry 1 for ACROSS

	nv2	xv2(i) i = 1, nv2	v2(i) n = 1, nv2	±delx	nval(optional)
1.					
2.					
3.					
4.					
5.					

Condition 2: Input parameters for extending data are in error. For option 1, less than 1 or more than 5 points were used. For option 2, either the distance extended or the velocity are zero or negative. For option 3, either the distance extended is zero or negative or the number of points for curve fitting is less than 2.
(Statements 27+2, 52-1, 28, 29, 56+2, 70-1, 57 or 58)

Condition 3: Input parameters for artificial data are in error. The distance coordinates must be in order of increasing values.

Condition 4: Subroutine SELECT failed because iflag=1. (Statement 6+1)

Condition 5: In selecting a set of t-x points within the desired interval xmin to xmax, subroutine SELECT found fewer than two points (jwatch=1). The option is therefore presented to use artificial data.
(Statement 6+5)

Condition 6: Subroutine XTEND failed. In extending data, either one of the points added falls within the range of the t-x data already selected (ioptl=1), there are insufficient points in the t-x array to make a least squares fit (iopt3=1), the points added result in over 400 t-x pairs which exceeds program dimensions (idim=1), or the shotpoint lies outside the range of the data. (Statements 9+3, 10+1, or 12)

Condition 7: Subroutine COMP failed (iflag=1).
(Statements 13, 14+1, 18+1, 300+7)

Condition 8: In sampling the reduced arrival-time curve at equally spaced intervals, the program dimensions were exceeded (ipt=1). Increase dint in entry 1. (Statement 335+3)

Condition 9: After having been lowered to the idth horizon, the shotpoint no longer falls within the limits of the data as indicated. Data should be extended further. (Statement 16+2)

Condition 10: In searching for a solution, the data has been exhausted without finding one. If a prior solution has not yet been determined, data may have to be extended further. (Statement 402+1)

Condition 11: The slope of the arrival-time curve at the distance coordinate xd indicated, is too steep to permit ray to propagate into the layer or horizon indicated. The apparent velocity of the arrival-time curve at that point is also indicated. (Statement 18+3)

Condition 12: At the distance coordinate indicated, the submerging ray is propagating upward instead of downward. (Statement 19+4)

Condition 13: The velocity chosen in entry 1 for the horizon to be computed is less than occurs in the overlying layer as indicated. Either change the model, or v2 in entry 1. (Statement 82-4)

Condition 14: In reflecting or turning ray around to return to the surface from the distance coordinate indicated, the ray is propagating downward instead of upward. (Statement 82+3)

Condition 15: From the starting point gxtry on the idth horizon, the estimated critical reflection point at coordinates (xitry,etatry) is shallower than the point (xitry,ydum) with the same distance coordinate xitry, on the "overlying" horizon. Parameters are printed out. (Statement 300+8)

Condition 16: Subroutine HSECT failed. (Statement 30)

Condition 17: The ray starting at the distance coordinate gxtry indicated, on the idth horizon, is attempting to propagate outside the boundaries of the model after reflection. (Statement 84+1 or 87)

Condition 18: Subroutine CSECT failed. iflag=1. (Statement 86)

Condition 19: Subroutine RAYUP failed for a ray starting on the idth horizon from the distance coordinate gxtry indicated. iflag=1. (Statement 400+2)

Condition 20: For a ray starting on the idth horizon at the distance coordinate gxtry indicated, over 300 increments in delll (entry 1) have occurred without the reflected ray having successfully intersected the shotpoint. The value of delll in entry 1 is probably too small. (Statement 317+8)

Condition 21: This condition is often overcome by decreasing the absolute value of delx. The values for gxtry, taucr(i), treal, xicr(i), etacr(i) and tupcr(i) are printed out in order. Gxtry is the distance coordinate on the idth horizon nearest the shotpoint from which a reflected ray can possibly intersect the shotpoint. However, the reflection time taucr(i) is greater than the arrival-time treal at gxtry and for this nearest possible point, it should be less. This probably means that the deepest known horizon, the idth, is too deep; that the horizon being calculated is shallower than the idth. By comparing taucr(i) with treal, an estimate of the discrepancy can possibly be made. If they are nearly equal, some minor adjustments can be made in the arrival-time data or the velocity v2 decreased slightly and the program rerun successfully. The coordinates (xicr(i),etacr(i)) are the trial points on the horizon being calculated which correspond to reflection time taucr(i). Tupcr(i) is the travel-time from (xicr(i),etacr(i)) to the shotpoint. (Statement 330-4)

Condition 22: Subroutine WAVED failed, either because idim or isplin=1. (Statement 335+2)

DATAIN Entry List (entirely from terminal)

Entry 1. Type 3 sets of data strings each in list directed (v) format.

String 1

*1.	k	Shot number index. $\leq$50 for .org $\leq$10 for .com.
2.	ng(k)	Number of data points. $\leq$24 for .org $\leq$50 for .com.
3.	sx(k)	Shot location distance coordinate.

*Type 0; for exit via subroutine OUTPUT.

4. sd(k) Shot depth.
5. offset(k) Shot offset distance perpendicular to line. If actual distances are to be input for gx(k,j) below, set offset(k) = -1.

String 2

1. gx(k,1) Distance coordinate of data point furthest to left.
2. (gx(k,j), j=2,ng(k)) Intervals between successive data points. note: if intervals are constant, only 2 values are necessary i.e., 10, 20; implies the first data point is at x=10, the second at x=30, the third at x=50, etc.

String 3

1. (gt(k,j), j=1,ng(k)) Arrival times.

Entry 2: Type 4 sets of data strings in list directed (v) format.

*1. i Horizon number index.
2. nc(i) Number of compartments for horizon 1.

String 2

1. (cx(i,j), j=1,nc(i)) Compartment distance coordinates.

String 3

1. (cy(i,j), j=1,nc(i)) Compartment elevation coordinates.

String 4

1. (cv(i,j), j=1,nc(i)) Compartment velocities.

FUDGE Entry List

Entry 1: Type five values in list directed (v) format.

*1. const The constant value to be added to the variable in common.

2. Any one of the following words representing the variable to be changed:
   gt (arrival-time)
   gx (data point distance coordinate)
   cx (compartment distance coordinate)
   cy (compartment elevation coordinate)
   cv (compartment velocity)
   gtc (arrival-time for combined data)
   gxc (data point distance coordinate for combined data)

3. snum The index referring to the shot number (k) or layer number.

* Type 0; for exit via subroutine OUTPUT

DAITAIN input form

File Name_______________

k, ng, sx, sd, offset																							
x(i)																							
t(i)																							
k, ng, sx, sd, offset																							
x(i)																							
t(i)																							
k, ng, sx, sd, offset																							
x(i)																							
t(i)																							
k, ng, sx, sd, offset																							
x(i)																							
t(i)																							
k, ng, sx, sd, offset																							
x(i)																							
t(i)																							

4. is The index corresponding to the first value to be changed.
5. ie The index corresponding to the last value to be changed.
note: If only one value is to be changed, is=ie.

LINDT Condition

Condition 1: Either the value of xp indicated does not lie within the range of the array x(i) representing the distance coordinates of the t-x points, or the number of points nobs, available for least squares fitting is less than two or the number of points for least squares fitting nval, exceeds the number in the original array. (Statement 20-7)

PICK Entry List

Entry 1: Type one data string in list directed (v) format.

*1. shot1 The index number k, of the arrival-time data from which a subset is to be selected.

2. ext1 Either org or com. The file name extension for the data set being used.

3. xmin A distance value slightly less than or equal to that of the first point used.

4. xmax A distance value slightly greater than or equal to that of the last point used.

5. shot2 The index number assigned to the selected subset.

6. ext2 Either org or com. The extension assigned to the selected subset.

7. add (optional) A value added to the arrival-times of the selected subset. Default=0.

PICK Conditions

Condition 1: During execution of subroutine SELECT, no arrival-time points were found in the interval between xmin and xmax. (Statement 16+2)

Condition 2: During execution of subroutine SELECT, the initial arrival-time data were found to be incorrect. (Statement 25)

PLOTX Entry List
Mount a nine-track tape

Entry 1. Type eight values in list directed (v) format.

1. pl The plotter to be used. Type either var for Varian or cal for Calcomp.

* Type 0; for exit via subroutine OUTPUT

2. tmin The minimum time able to be plotted. Time lines and labels will begin with this value. Therefore, choose zero or some multiple of 10.

3. tmax The maximum time able to be plotted; tmax should also be a multiple of 10 and somewhat larger than the largest time value to be plotted.

4. xmin The smallest distance coordinate able to be plotted, preferably a multiple of 10 and less than the value for the furthest left-hand shotpoint to be plotted.

5. xmax A distance coordinate somewhat larger than the value for the furthest right-hand shotpoint to be plotted.

6. timehi The desired height of the plot, in inches.

7. xlong The desired length of a plot for a 24-seismometer spread on the paper in inches.

8. space The approximate distance between successive seismometers for this 24 trace spread.

The program now prints out suggested values of tscal, dscal, tlab and dlab. The user is free to use these values or any others. The suggested values of tscal and dscal are determined to make the plot the size specified in entry 1. Changing these values changes the plot size. The suggested values of tlab and dlab should be rounded off to some value suitable for printing in the margins of the plot, such as 5, 10, 50, 100, etc. If the exact values suggested are used, time lines and labels will be exactly 1/2-inch apart and distance lines and labels 3/4-inch apart. Rounding tlab and dlab off to a smaller value puts labels and lines closer together, which is frequently preferred.

Entry 2. Type four values in list directed (v) format.

1. tscal The number of time units per inch as plotted.

2. dscal The number of distance units per inch as plotted.

3. tlab The labeling interval and the interval between horizontal time lines across the plot. There will be 4 equally spaced tick marks between time lines along margins.

4. dlab The labeling interval and the interval between vertical distance lines across the plot. There will be 4 equally spaced tick marks between distance lines along the margins.

Entry 3. Type two or three values in list directed (v) format.

*1. k The index number of the data to be plotted.

Type 0; after completion of plot for return to mainline.

2. ext The extension of the data to be plotted--either org or com.

3. sym (optional) The plotting symbol. The default option is a predetermined symbol, corresponding to the last digit of k.

RAYD Conditions

Condition 1: The horizon indices used are incorrect. Note that ip and id must be between 1 and 10 and id greater than or equal to ip. (Statement 33-4 or 33-5)

Condition 2: Subroutine COMP failed at the distance coordinate xp indicated. (Statement 33 or 34, or 13, or 100+1)

Condition 3: A ray is propagating upward instead of downward at the distance coordinate xp indicated. The angle of propagation ap, with respect to the vertical is also indicated. (Statement 200+1)

Condition 4: Over 50 compartment boundaries were encountered in one of the layers at the distance coordinate xp indicated. (Statement 14+1)

Condition 5: Subroutine HSECT failed at the distance coordinate xp indicated. (Statement 34+2)

Condition 6: Subroutine CSECT failed. (Statement 16)

Condition 7: In propagating to the idth horizon, the ray beginning at the distance xp indicated, encountered 50 or more compartment boundaries. (Statement 38+6)

Condition 8: A ray, in propagating downward from the distance coordinate xp, has ventured outside the boundaries of the velocity model. (Statement 12+1 or 21)

Condition 9: Subroutine ANGLE failed at the distance coordinate xp indicated. (Statement 100+4)

RAYD1 Conditions

Condition 1: The horizon indices used are incorrect. Note that ip and id must be between 1 and 10 and id greater than or equal to ip. (Statement 33-4 or 33-5)

Condition 2: Subroutine COMP failed at the distance coordinate xp indicated. (Statement 33 or 34 or 13)

Condition 3. A ray is propagating upward instead of downward at the distance coordinate xp indicated. The angle of propagation ap, with respect to the vertical is also indicated. (Statement 32+3)

Condition 4: Over 50 compartment boundaries were encountered in one of the layers at the distance coordinate xp indicated. (Statement 14+1)

Condition 5: Subroutine HSECT failed at the distance coordinate xp indicated. (Statement 34+2)

Condition 6: Subroutine CSECT failed. (Statement 16)ndition 8: A ray, in propagating downward from the distance coordinate xp, has ventured outside the boundaries of the velocity model. (Statement 12+1 or 21)

Condition 8: A ray, in propagating downward from the distance coordinate xp, has ventured outside the boundaries of the velocity model. (Statement 12+1 or 21)

RAYTRACE Entry List

Entry 1: Type 6 to 12 values in list directed (v) format.

*1. layd The horizon index from which arrivals are to be calculated. $2 \leq layd \leq 10$.

2. shotx The distance coordinate of the shotpoint.

3. shotd Shot depth. Use a negative value for shotd if the shot is on or within the layer specified by horizon layd.

4. xstart The distance coordinate on horizon layd of the first up-going ray to be calculated. The value for xstart should lie beyond the critical reflection point. If it does not, the program will place it there. Therefore, if in doubt, let xstart=shotzx

5. xend The distance coordinate on layd of the last up-going ray to be calculated.

6. delx The calculation increment on horizon layd. delx>0.

7. bend (optional) Type "no" if rays are not to be refracted or reflected in subroutines RAYUP and RAYD at compartment boundaries. Type "yes" if they are. Default = no.

8. tim0 (optional) The initial starting time. Default = 0.

9. betal (optional) The initial submergence angle in degrees, to be tried. Default=0. In cases of steep dip, 0 may already result in an angle beyond critical (Condition 7), in which case some other angle, consistent with the sign convention, must be used.

10. dbeta (optional) The value in degrees by which betal is incremented. Default=0.5°. dbeta>0.

11. alphal (optional) Initial emergence angle for a diffraction. Default=0° (e.g., vertical). alpha $1 \geq 0$.

12. dalpha (optional) The value in degrees by which alphal is incremented. Default=5° dalpha>0.

Entry 2: Type 2 values for each diffraction point in list directed (v) format.

1. ndifr The number of diffraction points selected on horizon layd. $0 \leq ndifr \leq 30$. Zero assumes no diffraction points.

2,4,6 xdifr(i) The distance coordinate on horizon layd of the ith diffraction point.

3,5,7 direc(i) The value for direc(i) is an alphanumeric and is either + or - . Type + if the diffractions to be calculated are to bend in a direction away from the shotpoint and type - if they bend toward the shotpoint.

RAYTRACE Conditions

Condition 1: Subroutine COMP failed.
(Statement 151-1 or 2+1, or 3+1, or 100 or 13-1)

Condition 2: The shotpoint lies below the refracting horizon. This condition may be overcome by by-passing the down-going portion of the ray by setting the shot depth (shotd) in entry 1 to a negative value. This in effect places the shot on the refracting horizon, layd. (Statement 4-4)

Condition 3: In raising the shotpoint to the horizon immediately overlying it, the angle of the ray equals that of the overlying horizon. Hence no intersection is possible. (Statement 9+3)

Condition 4: May occur in tracing a ray from the horizon immediately above the shotpoint (lays) down to the refracting horizon (layd) during execution of subroutine RAYD2 (iflag = 1) because either subroutines COMP, CSECT, or HSECT failed. (Statement 6+1)

Condition 5: The submerging ray could not penetrate to the refracting horizon (layd) on the first trial because Condition 1 occurred (also see Condition 6). (Statement 6+5)

Condition 6: The submerging ray could not penetrate to the refracting horizon (layd) after the first trial because Condition 1 occurred. However, the results of the previous trial are printed out as though normal exit from subroutine RAYD2 had occurred and at the users discretion these values may be used for the critical reflection point on the refracting horizon. (Statement 6+5 or 11)

Condition 7: The submerging ray could not penetrate the refracting horizon (layd) during execution of subroutine RAYD2 (ibomb =1) on the first trial because the critical angle was exceeded at the horizon indicated. This condition may occur in cases of steep dip and can often be overcome by assigning a non-zero value to betal in entry 1.

Condition 8: The submerging ray could not penetrate to the refracting horizon (layd) during execution of subroutine RAYD2 after the first trial because the critical angle was exceeded at the horizon indicated which is above the refracting horizon. However, the results of the previous trial are printed

out as though the critical angle was exceeded on horizon layd and at the users discretion these values may be used for the critical reflection point on the refracting horizon. (Statement 74)

Condition 9: Subroutine COMP failed.
(Statement 32, or 31, or 52-3, or 52)

Condition 10. In attempting to trace ray from the refracting horizon (layd) at the location indicated (xup) to the surface, subroutine RAYUP failed (iflag = 1). Therefore, this ray is ignored, xup incremented and the process started again. (Statement 57+3)

Condition 11. Subroutine RAYD2 could not successfully trace a diffracted ray to the surface (iflag=1). The emergence angle with respect to the refracting horizon is printed out. This particular ray will be ignored and execution continued.

RAYUP Conditions

Condition 1: The horizon indices used are incorrect. Note that ip must be greater than iu and the maximum value allowed is 10.
(Statement 7+3 or 7+4)

Condition 2: Subroutine COMP failed at the distance coordinate xp, indicated. (Statement 2 or 18+1)

Condition 3: A ray inpinging on the refracting horizon (ip) approaches it from above instead of below at the distance coordinate xp indicated. Some adjustments are made and execution is permitted to continue.
(Statement 35+4 or 1+2)

Condition 4: Fifty compartment boundaries were encountered in one of the layers at the distance coordinate xp indicated. (Statement 14+1)

Condition 5: Subroutine ANGLE failed. The critical angle was exceeded at the horizon ip and distance coordinate xp indicated. Therefore, the ray can not refract into the next overlying layer. (Statement 27)

Condition 6: A refracted ray calculated by subroutine ANGLE is propagating downward instead of upward at the distance coordinate xp indicated. The angle of propagation ap, with the vertical is also indicated.
(Statement 22)

Condition 7: A ray, in propagating to the surface from the distance coordinate xp indicated, has ventured outside the boundaries of the velocity model. (Statement 12+1 or 21)

Condition 8: Subroutine CSECT failed. (Statement 31)

Condition 9: In propagating to the surface, a ray beginning at the distance coordinate xp indicated, encountered 50 or more compartment boundaries.
(Statement 100 +12)

Condition 10: Subroutine HSECT failed at the distance coordinate xp indicated. (Statement 32)

Condition 11: Subroutine ANGLE failed at the distance coordinate xp indicated. (Statement 19+1)

RECIP Entry Form

Entry 1: Parameters applicable to both shot points.

Type 3 to 8 values in list directed (v) format.

*1. id The index of the deepest known horizon. Id is one less than the index of the horizon being calculated. $1 \leq id \leq 10$.

2. curv Either lin or spl. The value of curv determines whether the t-x data are to be approximated by a series of straight line segments or by a cubic spline.

3. delx The distance interval at which successive depth calculations are attempted. Delx should generally not exceed the interval between successive geophone groups and be large enough so that less than 201 calculations will be attempted; e.g., (xmax1-xmin1)/delx$\leq$201. In cases when the value of the delx used approaches the thickness of the layer being calculated, improved coverage is achieved by decreasing the value of delx. delx>0.

4. bend1 (optional) Either yes or no, depending on whether or not rays will be subject to refraction and reflection at compartment boundaries between horizons iu and id. Default is "no".

5. bend 2 (optional) Either yes or no, depending on whether or not rays will be subject to refraction and reflection at compartment boundaries below horizon id. Default is "no".

6. iu (optional) The index of the horizon to which the initial t-x data are referred. Default=1. iu$\leq$id

7. mycip (optional) The value for mycip is either yes or no, depending

8. tss on whether or not the user inputs a value for the reciprocal

9. tuh1 time, tss, and values for the uphole time for the left shot

10. tuhr point, tuh1, and the right shotpoint, tuhr. Default for mycip is "no". If mycip = yes, the value for tss must be input and this value must equal the reciprocal time plus the uphole time (e.g., corrected as though the shotpoints were on the surface). If the default option is used (mycip = no), t-x data must extend or be extended at least as far as the reversed shotpoint so that the program may calculate a value

for the reciprocal time, in which case it also calculates the uphole times. A value of yes is usually used for mycip if the shotpoints are offset well beyond the ends of the t-x data sets being used.

Entry 2. Parameters applicable to the left shotpoint.
Type 6 to 9 values in list directed (v) format.

*1. kl Shot number index for the left shotpoint.

2. extl Either org or com. The extension of the file name containing the t-x data.

3. xminl The t-x arrays may contain points from horizons other than the one being calculated. Xminl is a distance coordinate equal to or less than the lower bound of the distance coordinates of the t-x data being used.

4. xmaxl A distance coordinate equal to or greater than the upper bound of the distance coordinates of the t-x data being used. Hence xminl and xmaxl bracket the data. xminl$\leq$xmaxl If xminl=xmaxl, the program will not search the data set for t-x values and artificial data must be used.

5. delxl The increment in distance at which successive points are selected on the t-x graph during execution of subroutine WAVED, to reduce it to successively lower horizons. The smaller the value of delxl, the denser the arrays of arrival-times available for sampling by subroutine EQSPACE. Delxl should generally be somewhat less than delx in entry 1, but large enough so that less than 401 points will be sampled between xminl and xmaxl.

6. dintl The sampling interval used during execution of subroutine EQSPACE for creating t-x data arrays at equally spaced distance intervals. Generally, dintl=delxl and is subject to the same conditions.

7. nvall (optional) The number of points used for fitting the t-x arrays with least squares straight line segments (see fig. 22) during execution of subroutine LINDT. LINDT is executed after EQSPACE and hence the interval between successive points is dint. The value of nvall should be larger for scattered t-x data, and equal to 2 for "perfect" data. Nvall$\geq$2, but less than the number of points in the t-x arrays after execution of EQSPACE. Default=3.

8. nprinl (optional) Either 0 or 1. If the intermediate t-x data on horizons 2 to id are to be printed out during execution of subroutine WAVED, set nprinl=1. Default=0.

9. nstar1 (optional) The number of "star" points to be calculated within the interval of the t-x data sets. The distance coordinates (xstar1(i),i=1,nstar1) of these points are equally spaced within the interval from xmin1 to xmax1 which may have been expanded from items 3 and 4 above to include extension of data (see below). Subroutine WAVED calculates their arrival-times, tstar1(i), on horizon iu, and traces the rays from these points through the model to points (xprim1(i), tprim1(i),i=1,nstar1) on horizon id. The "star" point values may be printed on command. 1<nstar=30. Default=10.

Entry 2a: Parameters for extending t-x data arrays for the left shotpoints to the left. Type 1 to 10 values in list directed (v) format.

*1. opt11 The index specifying the option to be used. If opt11<0 or opt11>3, control is returned to mainline.
- if opt11=0 Data are not extended to the left.
- if opt11=1 Data are extended and to consist of specific points specified by the parameters in items 2 and 3 below.
- if opt11=2 Data are extended along a straight line which intersects the first point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.
- if opt11=3 Data are extended along a least squares straight line constructed through the t-x data points as specified in items 2 and 3 below.

2. ±dx11
- if opt11=1 The number of t-x data points to be added to the left of the original t-x arrays. $1 \leq dx11 \leq 5$
- if opt11=2 The distance which the data are to be extended to the left of the first distance coordinate of the t-x arrays. dx11<0.
- if opt11=3 Same as for opt11=2.

3. ±fn11(i)
- if opt11=1 (fn11(i)i=1,2dx11) The distance coordinates, then followed by the arrival-times for the number of points specified by item 2 above. The distance coordinates must be of increasing value and all be less than the first t-x data point of the original array.
- if opt11=2 ±fn11(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to increase with increasing distance, then fn11(1)>0. Otherwise fn11(1)<0.
- if opt11=3 fn11(1) A value specifying the number of left-most points of the original t-x arrays to be used for a least squares straight line approximation. The extended data will fall on this line and the arrival-time of the first point of the original arrays will also be changed to fall on this line.

Entry 2b: Parameters for extending t-x data arrays for the left shotpoint to the right. Type 1 to 10 values in list directed (v) format. If mycip="no" in entry 1 above, data must be extended sufficiently far to include the reversed shotpoint.

*1. optlr The index specifying the option to be used. If optlr<0 or optlr>3, control is returned to the mainline.
if optlr=0 Data are not extended to the right.
if optlr=1 Data are extended to consist of specific points specified by parameters in items 2 and 3 below.
if optlr=2 Data are extended along a straight line which intersects the last point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.
if optlr=3 Data are extended along a least squares straight line constructed through the t-x data points as specified in items 2 and 3 below.

2. ±dxlr
if optlr=1 The number of t-x data points to be added to the right of the original t-x arrays. $1 \leq dxll \leq 5$
if optlr=2 The distance which the data are to be extended to the right of the last distance coordinate of the t-x arrays. dxlr>0
if optlr=3 Same as for optlr=2.

3. ±fnlr(i)
if optlr=1 (fnlr(i),i=1,2dxlr) The distance coordinates then followed by the arrival-times for the number of points specified in item 2 above. The distance coordinates must be of increasing value and all be greater than the last t-x data point of the original array.
if optlr=2 ±fnlr(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to decrease with increasing distance, then fnlr(1)<0, otherwise fnll(1)>0.
if optlr=3 fnlr(1) A value specifying the number of right-most points of the original t-x arrays to be used for least squares straight line approximation. The extended data will fall on this line and the arrival-time of the last point of the original arrays will also be changed to fall on this line.

Entry 3: Parameters applicable to the right shotpoint.
Type 6 to 9 values in list directed (v) format.

*1. kr Shot number index for the right shotpoint.

2. extr Either org or com. The extension of the file name containing the t-x data.

3. xminr The t-x arrays may contain points from horizons other than the one being calculated. Xminr is a distance coordinate equal to or less than the lower bound of the distance coordinates of the t-x data being used.

4. xmaxr A distance coordinate equal to or greater than the upper bound of the distance coordinates of the t-x data being used. Hence xminr and xmaxr bracket the data; $xminr \leq xmaxr$.

If xminr=xmaxr, the program will not search the data set for t-x values and artificial data must be input.

5. delxr The increment in distance at which successive points are selected on the t-x graph during execution of subroutine WAVED, to reduce it to successively lower horizons. The smaller the value of delxr, the denser the arrays of arrival-times available for sampling by subroutine EQSPACE. Delxr should generally be somewhat less than delx in entry 1, but large enough so that less than 401 points will be sampled between xminr and xmaxr.

6. dintr The sampling interval used during execution of subroutine EQSPACE for creating t-x data arrays at equally spaced distance intervals. Generally dintr=delxr and is subject to the same conditions.

7. nvalr (optional) The number of points used for fitting the t-x arrays with least squares straight line segments (see fig. 22) during execution of subroutine LINDT. LINDT is executed after EQSPACE and hence the interval between successive points is dint. The value of nvalr should be larger for scattered t-x data, and equal to 2 for "perfect" data. Nvalr $\geq$2, but less than the number of points in the t-x arrays after execution of EQSPACE. Default =3.

8. nprinr (optional) Either 0 or 1. If the intermediate t-x data on horizons 2 to id are to be printed out during execution of subroutine WAVED, set nprinr=1. Default=0.

9. nstarr (optional) The number of "star" points to be calculated within the interval of the t-x data sets. The distance coordinates (xstarr(i),i=1,nstarr) of these points are equally spaced within the interval from xminr to xmaxr which may have been expanded from items 3 and 4 above to include extension of data (see below). Subroutine WAVED calculates their arrival-times, tstarr(i), on horizon iu, and traces the rays from these points through the model to points (xprimr(i), tprimr(i),i=1,nstarr) on horizon id. The "star" point values may be printed on command; 1<nstarr=30. Default=10.

Entry 3a: Parameters for extending t-x data arrays for the right shotpoint to the left. Type 1 to 10 values in list directed (v) format. If mycip ="no" in entry 1 above, data must be extended sufficiently far to include the reversed shotpoint.

*1. optr1 The index specifying the option to be used. If optr1<0 or optr1>3, control is returned to mainline.
if optr1=0 Data are not extended to the left.
if optr1=1 Data are extended and to consist of specific points specified by the parameters in items 2 and 3 below.

if optrl=2 Data are extended along a straight line which intersects the first point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.
if optrl=3 Data are extended along a least squares straight line con structed through the t-x data points as specified in items 2 and 3 below.

2. ±dxrl
if optrl=1 The number of t-x data points to be added to the left of the original t-x arrays. $1 \leq dxrl \leq 5$
if optrl=2 The distance which the data are to be extended to the left of the first distance coordinate of the t-x arrays. $dxrl < 0$.
if optrl=3 Same as for optrl=2.

3. ±fnrl(i)
if optrl=1 (fnrl(i)i=1,2dxrl) The distance coordinates, then followed by the arrival-times for the number of points specified by item 2 above. The distance coordinates must be of increasing value and all be less than the first t-x data point of the original array.
if optrl=2 ±fnrl(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to increase with increasing distance, then $fnrl(i) > 0$. Otherwise, $fnrl(i) < 0$.
if optrl=3 fnrl(1) A value specifying the number of left-most points of the original t-x arrays to be used for a least squares straight line approximation. The extended data will fall on this line and the arrival-time of the first point of the original arrays will also be changed to fall on this line.

Entry 3b: Parameters for extending t-x data arrays for the right shotpoint to the right. Type 1 to 10 values in list directed (v) format.

*1. optrr The index specifying the option to be used. If $optrr < 0$ or $optrr > 3$, control is returned to the mainline.
if optrr=0 Data are not extended to the right.
if optrr=1 Data are extended to consist of specific points specified by parameters in items 2 and 3 below.
if optrr=2 Data are extended along a straight line which intersects the last point of the original t-x array. The slope of this line is the inverse of the velocity specified in item 3 below.
if optrr=3 Data are extended along a least squares straight line constructed through the t-x data points as specified in items 2 and 3 below.

2. ±dxrr
if optrr=1 The numbers of t-x data points to be added to the right of the original t-x arrays. $1 \leq dxrr \leq 5$
if optrr=2 The distance which the data are to be extended to the right of the last distance coordinate of the t-x arrays. $dxrr > 0$
if optrr=3 Same as for optrr=2.

3. ±fnrr(i)
   if optrr=1 (fnrr(i),i=1,2dxrr) The distance coordinates then followed by the arrival-times for the number of points specified in item 2 above. The distance coordinates must be of increasing value and all be greater than the last t-x data point of the original array.
   if optrr=2 ±fnrr(1) The velocity specifying the slope of the extended data points. If the arrival-time values are to decrease with increasing distance, then fnrr(1)<0. Otherwise fnrr(1)>0.
   if optrr=3 fnrr(1) A value specifying the number of right-most points of the original t-x arrays to be used for least squares straight line approximation. The extended data will fall on this line and the arrival-time of the last point of the original arrays will also be changed to fall on this line.

Entry 2c: To be entered only if artificial arrival-time data are used for the left shotpoint (e.g., if xminl=xmaxl in entry 2 above). Type (2nl+1) values in list directed (v) format.

*1. nl The number of artificial data points to be used.

2. (xl(i),i=1,nl) The distance coordinates of the artificial data points in increasing value.

3. (tl(i),1,nl) The corresponding arrival-times.

Entry 3c: To be entered only if artificial arrival-time data are used for the right shotpoint (e.g., if xminr=xmaxr in entry 3 above). Type (2nr+1) values in list directed (v) format.

*1. nr The number of artificial data points to be used.

2. (xr(i),i=1,nr) The distance coordinates of the artificial data points in increasing value.

3. (tr(i),i=1,nr) The corresponding arrival-times.

RECIP Conditions

Condition 1: Input parameters incorrect. Either id >10, id<iu (e.g., the index for the bottom layer is less than the upper), delx<0, or lin or spl are misspelled. (Statement 20-5)

Condition 2: Input parameters for left shot are incorrect. Either extension misspelled, shot index too large, xmin>xmax, incrementing or digitizing interval $\leq$0, number of star points >30 nval<2 or nval>30. (Statements 23-1, 22, or 121)

Condition 3: Input parameters for extending left shotpoint data are incorrect. (Statements 27+1, 52-1, 28, 29, 56+1, 70-1, 57, or 58)

RECIP Input Form

Condition 4: Input parameters for right shotpoint are incorrect. See Condition 2. (Statements 223-1, 322, or 221)

Condition 5: Input parameters for extending right shotpoint data are incorrect. (Statements 227+1, 252-1, 228, 229, 256+1, 270-1, 257, or 258)

Condition 6: Artificial data are not in increasing value of x. (Statement 1064-2)

Condition 7: Subroutine SELECT failed. (Statement 31 or 73)

Condition 8: Subroutine COMP failed. (Statements 76+2, or 79)

Condition 9: Subroutine XTEND failed. (Statements 304+1, 87+1, 89+1, or 90+1)

Condition 10: Data not extended far enough to project beyond reversed shotpoint. (Statement 91+1 or 92+1)

Condition 11: Program dimensions are exceeded during execution of subroutines WAVED or EQSPACE. (Statement 281+1 or 99+2)

Condition 12: Spline function parameters are out of range. (Statement 97 or 102)

Condition 13: Subroutine EQSPACE failed because program dimensions are exceeded. (Statement 122-4 or 106+1)

Condition 14: Spline function parameters are out of range. (Statement 156 or 157)

Condition 15: Subroutine COMP failed. (Statement 159+1 or 161+1)

Condition 16: The slope of the arrival-time curve at the distance coordinate indicated is too steep to permit ray to project into the layer indicated. The apparent velocity is the one indicated. (Statements 109+1 or 116+1)

Condition 17: At the distance coordinate indicated, the submerging ray is propagating upward instead of downward. (Statement 112+4 or 118+4)

Condition 18: Over 200 rays have been determined from the left shotpoint and hence, dimensions are exceeded. Increase value of delx. (Statement 113+2)

WAVED Conditions

Condition 1: Over 400 t-x points exist and dimensions permit only 400; idim=1. (Statement 32-5)

Condition 2: Over 400 t-x points resulted from subroutine EQSPACE. Increase the sampling interval dint; idim=1. (Statement 500+1)

Condition 3: Subroutine SPLIN1 failed. ispln=1 (Statement 55+1)

Condition 4: Subroutine COMP failed at the distance coordinate xp, and horizon index number indicated. (Statement 55+1)

Condition 5: The slope of the arrival-time curve at the distance coordinate xp, indicated is too steep to permit a ray to propagate into the layer iq, indicated. The apparent velocity is also indicated. (Statement 24+1)

Condition 6: Subroutine RAYD1 failed to projecct ray down to the next deepest horizon at the distance coordinate xp, and from the horizon indicated. (Statement 1+3)

Condition 7: In determining the t-x arrays from the horizon and distance coordinates indicated, over 400 points were calculated which are beyond the program dimensions. The variable delx is too large. idim=1 (Statement 25+5)

Condition 8: One of the "star" points at the distance coordinate xp, and horizon indicated, lies outside of the range of the data. It originated at the distance coordinate xstar, indicated. Its distance coordinate is arbitrarily set to 1,000,000. (Statement 6+2)

Condition 9: Subroutine COMP failed at the distance coordinate xp, and horizon indicated for a special "star" point originating at the coordinate indicated. Its distance coordinate is set to 1,000,000. (Statement 59+2)

Condition 10: The slope of the arrival-time curve for one of the "star" points at the distance coordinate xp and horizon indicated is too steep to permit ray to project into the underlying layer. The apparent velocity is also indicated as is the distance coordinate of its original location. Its distance coordinate is set to 1,000,000. (Statement 27+1)

Condition 11: Subroutine RAYD1 failed for one of the "star" points at the distance coordinate xp and horizon indicated. It originated at the point indicated. Its distance coordinate is set to 1,000,000. (Statement 28+3)

APPENDIX B

The following listings of the programs described in this report are written in Honeywell 68/88[1] FORTRAN IV, except for SPLIN1, which is coded in free format with tabs. These programs can be transported to other computer systems with moderate revision. Because attention was given to input/output timing over memory allocations, there are some very large storage arrays and common block data areas which may exceed the maximum allowed on some systems. In addition, some subroutines contain non-standard FORTRAN character strings, multiple entry points and calls to system commands.

These listings contain all code except system commands and calls to PL-1 file attaching and detaching routines which are done in subroutine DIGIT only.

[1] Use of trade names is for descriptive purposes only and does not constitute endorsement by the U.S. Geological Survey.

```
c     main driver program for interactive seismic refraction program
c
c     subr input and output are called automatically
c     other subr are by user request
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      dimension dell(200),delt(200),xi(200),eta(200),xplus(200),xminus(200),
     & v(200),vinst(200),tau(200)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/names/filnam,filorg,filcom,fillay,idsk
      character*6 filnam
      character*10 filorg,filcom,fillay
      character*10 route,sub,out1
      dimension route(25)
      logical ifirst
c
c      variable name route contains all the subroutine names which
c      can be called from the mainline.
c
      data route/"raytr","plotx","dum1","datain","dum1","dum1","crit",
     &"dum1", "output","input","fudge","perp","recip","pick","dum1",
     &"printr","term","dum1","build","dum1","dum1","dum1","dum1","dum1",
     &"dum1"/
      data out1/"out"/
      ifirst=.true.
      idsk=10
      itty=5
      jtty=6
c
c      input data from storage
c
      call input
   10 write(jtty,1000)
 1000 format(" type subroutine: ",$)
c
c      type subroutine to be called. type"out" for normal exit
c
   20 read(itty,1100) sub
      if(sub.eq.out1)go to 990
 1100 format(v)
      do 30 i=1,25
      if(sub.eq.route(i)) go to 40
   30 continue
      write(jtty,1200)sub
 1200 format(a10,1x," is not a valid name - try again: ",$)
      go to 20
c
c     branch to requested subroutine
c
   40 go to (50,60,70,80,90,100,110,120,130,140,150,160,170,180,190,200,
     &210,220,230,240,250,260,270,280,290),i
   50 call raytrace
      go to 10
   60 call plotx
      go to 10
   70 call dum1
      go to 10
   80 call datain(ifirst)
      go to 10
   90 call dum1
      go to 10
  100 call dum1
      go to 10
  110 call crit(xicrt,etacrt,taucrt,xcrt,tcrt)
      go to 10
  120   call dum1
      go to 10
  130   call output
      go to 10
  140 call input
      go to 10
  150 call fudge
      go to 10
  160 call perp
      go to 10
  170 call recip(nisect,xi,eta,vinst,v,tau,delt,dell,xminus,xplus)
      go to 10
  180 call pick
      go to 10
  190 call dum1
      go to 10
  200 call printr
      go to 10
  210 call term
      go to 10
  220 call dum1
      go to 10
  230 call build
      go to 10
  240 call dum1
      go to 10
  250 call dum1
      go to 10
  260 call dum1
      go to 10
  270 call dum1
      go to 10
  280 call dum1
      go to 10
  290 call dum1
      go to 10
c
c      output data to storage
c
  990 call output
```

```
  999 write(jtty,1500)
 1500 format("0end of program")
      call closet
      stop
      end


      real function acos(x)
c
c        This function returns the arc cos(x) in either the first or
c     fourth quadrant as a positive or negative angle.
c
      arg2=x
      if(abs(arg2).lt. .000001)go to 1
      arg1=sqrt(1.-x*x)
      acos=atan2(arg1,arg2)
      return
    1 acos=1.5707
      return
      end


      subroutine across(xicrt,etacrt,taucrt,xcrt,tcrt,id,xfar,x,t,nobs,
     & curv,al,bl,cl,nval,nstar,xstar,xprim,tstar,tprim,xir,etar,bend2)
c
c     subroutine across works in cunjunction with subroutine crit.
c     given critical subsurface parameters obtained from crit, xicrt,etacrt,
c     taucrt, xdrt, tcrt,  a set of t-x data points (x(i),t(i),i=1,nobs)
c     and the horizon with index id on which the t-x data are defined,
c     across calculates other subsurface points (xir(i),etar(i)),which
c     satisfy the t-x data.   the variable xfar is the most distant point
c     from the critical subsurface point for which depths are calculated.
c     curv ("lin" or "spl") determines whether t-x data are approximated
c     by a set of l.s. straight lines or a cubic spline.  if the former, nval
c     determines the number of points for each straight line segment.
c     al, bl, and cl  are the cubic spline coefficients.   nstar, xstar,
c     xprim, tstar, and tprim are star point values.   the value for
c     bend1 (either no or yes) determines whether refraction will occur
c     at compartment boundaries.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      dimension x(1),t(1),al(1),bl(1),cl(1),xstar(1),xprim(1),tstar(1),
     & tprim(1),xir(100),etar(100)
      dimension xv2(21),v2(21),xp(51),yp(51),ap(51),vp(51),taup(51)
      common/block1/xp,yp,ap,vp,taup
      character*3 curv,go,bend2
      external rathru(descriptors)
c
c     type input parameters. (see across entry form)
c
   71 write(jtty,1000)
 1000 format(" across entry 1; nv2*,xv2(i),v2(i),+or-delx,<nval>"
     & ,/," *",$)
      read(itty,1001)nv2,(xv2(i),i=1,nv2),(v2(i),i=1,nv2),delx,nval
 1001 format(v)
c
c     make checks on input data.
c
      if(nv2.lt.1)return
      if(nv2.le.20 .and. delx.ne.0. .and. nval.ge.2)go to 70
   72 write(jtty,1002)
 1002 format(" across condition 1")
      go to 71
c
c     assign xv2(1) a large negative value to include all of the
c     left hand portion of the model.
c
   70 xv2(1)=-1.e10
      if(nv2.eq.1)go to 73
      do 74 i=2,nv2
      if(xv2(i).gt.xv2(i-1))go to 74
      go to 72
   74 continue
c
c     assign f a value either plus or minus 1 depending on whether
c     delx is positive or negative.
c
   73 f=-1.
      if(delx.gt.0.)f=1.
c
c     a further check on input data.
c
      xdum=xcrt+99.*delx
      if(xdum*f.ge.xfar*f)go to 75
      go to 72
c
c     calculate arrival times relative to that of the critical subsurface point.
c
   75 do 76 i=1,nobs
      t(i)=t(i)-taucrt
   76 continue
c
c     initialize.  nr counts the number of calculated depth points,  and
c     xa is set to the first calculation point on the idth horizon.
c
      nr=0
      xis=xicrt
      etas=etacrt
      taus=taucrt
      xa=xcrt+delx
c
c     write a header for the answers.
c
      write(jtty,1003)
```

```
1003 format("       the answers are:",/,
     & 7x,"coordinate id horizon   coordinates of id+1 horizon",/,
     & 14x,"xr(i)",16x,"xir(i)     etar(i)")
  200 if(xa*f.gt.xfar*f)go to 100
c
c     calculate arrival times and derivatives using either spline or
c     straight line segment approximations.
c
      if(curv.eq."lin")go to 77
      call splindt(nobs,x,t,a1,b1,c1,xa,time,deriv)
      go to 78
   77 call lindt(nobs,x,t,xa,nval,time,deriv)
c
c     calculate submergent angle ad, with respect to vertical.
c
   78 call comp(xa,id,elev,x1,y1,x2,y2,v0,j,iflag)
      if(iflag.eq.0)go to 43
      write(jtty,1010)xa
 1010 format(" across condition 2 at xa=",f8.2)
      go to 103
   43 h=v0*deriv
      if(abs(h).lt.1.)go to 79
      vel=1./deriv
      write(jtty,1004)xa,id,vel
 1004 format(" across condition 3 at xa=",f10.1," horizon",i4," velocity",f10.1)
  103 xa=xa+delx
      go to 200
   79 beta=asin(h)
      slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      ad=beta-theta
c
c     make a check on the submergence angle.
c
      if(abs(ad).lt.1.57)go to 80
      write(jtty,1005)xa,id
 1005 format(" across condition 4 at xa=",f10.1," horizon",i4)
      go to 103
c
c     vertical rays are not permitted.
c
   80 if(abs(ad).lt..00001)ad=.001*f
c
c     initialize.
c
      taut=0.
c
c     project ray to infinity.
c
      call rathru(xa,id,ad,taut,nrays,bend2)
c
c     determine the index i, of ray at the same elevation as the
c     critical subsurface point.
c
      do 21 ii=1,nrays-1
      i=ii
      if(etas.gt.yp(i+1))go to 22
c
c     the unusual condition has occurred that the ray, extended to infinity,
c     does not extend below the depthe of the critical subsurface point.
c
   21 continue
      write(jtty,1006)xa
 1006 format(" across condition 5 from xa=",f8.1)
      go to 103
c
c     calculate a horizontal ray which intersects a ray segment obtained
c     from subroutine rathru.
c
   22 xiz=xp(i)-((yp(i)-etas)*(xp(i)-xp(i+1)))/(yp(i)-yp(i+1))
      if(xiz*f.gt.xa*f)xiz=xa
      if(xiz*f.gt.xis*f)go to 23
      write(jtty,1007)xa
 1007 format(" across condition 5 from xa=",f8.1)
      go to 103
c
c     calculate an average velocity vave, to be used. (this algorithm could
c     be improved)
c
   23 tsum=0.
      x0=amin1(xis,xiz)
      x2=amax1(xis,xiz)
      xv2(nv2+1)=x2+100.
      j=nv2
      x1=x0
      do 24 i=1,nv2
      if(x1.ge.xv2(j))go to 25
      j=j-1
   24 continue
      j=1
   25 if(j.eq.nv2)go to 26
   28 tsum=tsum+(xv2(j+1)-x1)/v2(j)
      if(x2.le.xv2(j+1))go to 27
      x1=xv2(j+1)
      j=j+1
      go to 28
   27 tsum=tsum-(xv2(j+1)-x2)/v2(j)
      go to 29
   26 tsum=tsum+(x2-x1)/v2(j)
   29 vave=(x2-x0)/tsum
c
c     calculate intersection of a ray originating at the critical subsurface
c     point with the ray from xa, projected to infinity which satisfies
c     the arrival time criteria.
c
      do 41 i=1,nrays-1
      tq=time-taup(i)
      if(tq.le.0.)go to 41
```

```
      xq=xp(i)
      yq=yp(i)
      vq=vp(i)
      aq=ap(i)
      call xisect(xq,yq,aq,tq,vq,vave,xis,etas,xie,etae,taue,izap)
      if(izap.ne.0)go to 41
      xdum=xp(i+1)
      x1=amin1(xq,xdum)
      x2=amax1(xq,xdum)
      if(xie.ge.x1 .and. xie.le.x2)go to 42
   41 continue
      write(jtty,1008)xa
 1008 format(" across condition 6 at xa=",f8.2)
      go to 103
c
c     print the results.
c
   42 write(jtty,1009)xa,xie,etae
 1009 format(10x,f9.2,10x,2f12.2)
c
c     increment and continue. (make and array not presently used
c     to store the new subsurface points)
c
      nr=nr+1
      xir(nr)=xie
      etar(nr)=etae
      xa=xa+delx
      go to 200
c
c     all done.  no star points are printed if the horizon calculated
c     was number 2.
c
  100 if(id.eq.1)go to 998
      write(jtty,1066)
 1066 format(" do you want the star points printed?",$)
 1016 format(a3)
      read(itty,1016)go
      if(go.ne."yes")return
      write(jtty,1011)(xstar(i),tstar(i),xprim(i),tprim(i),
     & i=1,nstar)
 1011 format(" the path of the special xstar points from horizons 1 to id:"
     & ,/,"      x(1)   time(1)     x(id)   time(id)",/,(4f10.1))
  998 return
      end


      subroutine angle(ap,vp,x1,y1,x2,y2,vq,aq,irefl,iflag)
c
c     given a ray propagating at an angle ap, with the vertical at velocity
c     vp, striking a horizon defined by coordinates (x1,y1),(x2,y2).
c     the velocity on the opposite side of the horizon is vq.    subroutine
c     angle calculates the angle of refraction aq, with respect to the
c     vertical.    irefl and iflag are normally zero.   irefl becomes 1 if
c     the impinging ray strikes the horizon at greater than the critical
c     angle.   iflag becomes one if the ray direction is incompatable
c     with the slope of the horizon.
c
      jtty=6
      irefl=0
      iflag=0
      slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      alpha=ap+theta
      if(abs(alpha).gt.(3.14158/2.))go to 900
      f=(vq/vp)*sin(alpha)
      if(abs(f).ge.1.)go to 910
      beta=asin(f)
      aq=beta-theta
      return
  900 write(jtty,1000)
 1000 format(" in subroutine angle ray direction incompatable with slope
     & of horizon")
      iflag=1
      return
  910 write(jtty,1001)
 1001 format(" critical angle exceeded in subroutine angle")
      irefl=1
      return
      end


       real function asin(x)
c
c        This function returns the arc sin(x) in either the first or
c     fourth quadrant as a positive or negative angle.
c
      arg1=x
      arg2=sqrt(1.-x*x)
      if(arg2.lt. .000001)go to 1
      asin=atan2(arg1,arg2)
      return
    1 asin=1.5707
      if(x.lt.0.)asin=-1.5707
      return
      end


      subroutine build
c
c      combines arrival-time data from 2 shotpoints as though they
c      were recorded from a single shotpoint and stores the resulting
c      data into file name extension .com
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
```

```
      dimension t1(200),x1(200),t2(200),x2(200)
      integer*4 opt
      character*3 ext1,ext2
      external select(descriptors)
c
c      input and check parameters. (see build entry form)
c
    1 write (jtty,1000)
 1000 format(" build entry 1; k*,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2,
     & opt",/," if opt=1; xsmal,xlarg or if opt=2; add",/," *",$)
      read(itty,1001)k,k1,ext1,xmin1,xmax1,k2,ext2,xmin2,xmax2,opt,dum1,dum2
 1001 format(v)
      if(k.lt.1)go to 300
      if(k.gt.10)return
      if(xmin1.le.xmax1 .and. xmin2.lt.xmax2 .and. (opt.eq.1 .or. opt.eq.2)
     & .and. (ext1.eq."org" .or. ext1.eq."com") .and.
     & (ext2.eq."org" .or. ext2.eq."com") .and. k1.le.50 .and. k2.le.50
     & .and. k1.gt.0 .and. k2.gt.0)go to 2
   30 write(jtty,1008)
 1008 format(" build condition 1")
      go to 1
    2 if(ext1.eq."org")go to 31
      if(k1.le.10)go to 31
      go to 30
   31 if(ext2.eq."org")go to 32
      if(k2.le.10)go to 32
      go to 30
c
c       direct execution according to option
c
   32 if(opt.eq.1)go to 3
c
c      user has selected own constant to be added.  opt=1
c
      add=dum1
c
c      select the appropriate data subsets and add the constant add
c      to the subsidiary data.
c
  100 call select(xmin1,xmax1,ext1,k1,npts1,t1,x1,jwatch,iflag)
      if(iflag.eq.0 .and. npts1.gt.0)go to 4
      if(xmin1.eq.xmax1)go to 4
    5 write(jtty,1003)
 1003 format(" build condition 2")
      go to 1
    4 call select(xmin2,xmax2,ext2,k2,npts2,t2,x2,jwatch,iflag)
      if(iflag.eq.0 .and. npts2.gt.0)go to 6
      go to 5
    6 do 7 i=1,npts2
    7 t2(i)=t2(i)+add
      nobs=npts1+npts2
      if(nobs.le.200)go to 8
      write(jtty,1004)
 1004 format(" build condition 3")
      go to 1
c
c      combine the points into one array
c
    8 if(xmin1.lt.xmin2)go to 9
      j=1
      do 10 i=npts2+1,nobs
      t2(i)=t1(j)
      x2(i)=x1(j)
      j=j+1
   10 continue
      do 11 i=1,nobs
      t1(i)=t2(i)
      x1(i)=x2(i)
   11 continue
      go to 15
    9 j=1
      do 12 i=npts1+1,nobs
      t1(i)=t2(j)
      x1(i)=x2(j)
      j=j+1
   12 continue
c
c      arrange points in increasing value of x
c
   15 continue
      do 13 i=1,nobs-1
      if(x1(i).eq.x1(i+1))go to 14
      x1(i+1)=x1(i+1)+abs(x1(i+1)*.000001)
   14 xsmall=amin1(x1(i),x1(i+1))
      if(xsmall.eq.x1(i))go to 13
      tsmall=t1(i+1)
      x1(i+1)=x1(i)
      t1(i+1)=t1(i)
      x1(i)=xsmall
      t1(i)=tsmall
      go to 15
c
c      put the arranged points into the new combined array
c
   13 continue
      ngc(k)=nobs
      if(ext1.eq."org")go to 16
      sxc(k)=sxc(k1)
      sdc(k)=sdc(k1)
      offsc(k)=offsc(k1)
      go to 17
   16 sxc(k)=sx(k1)
      sdc(k)=sd(k1)
      offsc(k)=offset(k1)
   17 do 18 i=1,nobs
      gxc(k,i)=x1(i)
      gtc(k,i)=t1(i)
   18 continue
```

```
      write(jtty,1005)k,nobs
 1005 format(" combined shot index",i5," has",i5," jugs")
      go to 1
c
c      return to mainline via subroutine output
c
  300 call output
      return
c
c      calculate the constant add to be added to subsidiary data
c
    3 if(dum1.lt.dum2)go to 33
      write(jtty,1009)
 1009 format(" build condition 2")
      go to 10
   33 xsmal1=dum1
      xlarg1=dum2
      xsmal2=dum1
      xlarg2=dum2
c
c      select the points with which the two data sets are compared
c
      call select(xsmal1,xlarg1,ext1,k1,npts1,t1,x1,jwatch,iflag)
      if(iflag.eq.0 .and. npts1.gt.0)go to 19
   25 write(jtty,1006)
 1006 format(" build condition 4")
      go to 1
   19 call select(xsmal2,xlarg2,ext2,k2,npts2,t2,x2,jwatch,iflag)
      if(iflag.eq.0 .and. npts2.gt.0)go to 20
      go to 25
   20 if(npts1.gt.1 .or. npts2.gt.1)go to 21
      add=t1(1)-t2(1)
      write(jtty,1007)x1(1),t1(1),x2(1),t2(1),add
 1007 format(" build condition 5",5f8.2)
      go to 1
c
c      calculate least square line through the comparison points and
c      calculate the constant add.
c
   21 if(npts1.eq.1)go to 22
      call lslin(npts1,x1,t1,a11,a12)
      if(npts2.eq.1)go to 23
   22 call lslin(npts2,x2,t2,a21,a22)
      if(npts1.eq.1)go to 24
      xmin=amax1(xsmal1,xsmal2)
      xmax=amin1(xlarg1,xlarg2)
      xave=(xmin+xmax)/2.
      tval1=a11+a12*xave
      tval2=a21+a22*xave
      add=tval1-tval2
      go to 101
   24 tval1=t1(1)
      tval2=a21+a22*x1(1)
      add=tval1-tval2
      go to 101
   23 tval2=t2(1)
      tval1=a11+a12*x2(1)
      add=tval1-tval2
  101 write(jtty,1010)add,k2,ext2,k1,ext1
 1010 format(" constant",f8.2," is added to shot",i4,1x,a3,/,
     &"  to combine it with shot",i4,1x,a3)
c
c      direct execution for addition of constant add to the control
c      and subsidiary data points
c
      go to 100
      end


      subroutine comp(x,i,y,x1,y1,x2,y2,v,j,iflag)
c
c     given a distance coordinate x and a horizon index i, subroutine comp
c     determines the y (elevation) coordinate corresponding to x on horizon
c     i, the upper left compartment coordinates (x1,y1) and the upper right
c     compartment coordinates (x2,y2) containing x, the compartment velocity
c     v, and the compartment index j.   iflag is normally 0.   it becomes 1
c     if x lies outside the horizon endpoints (the model boundaries for
c     the ith horizon), if the compartment boundary x2 is less than
c     or equal to x1 (the successive compartment boundaries are not of
c     increasing value), or if the index of the compartment containing
c     x exceeds 50.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      iflag=0
      jmax=nc(i)
      if(jmax.gt.50)go to 920
      j=1
      if(x.ge.cx(i,1))go to 11
      iflag=1
      write(jtty,1000)x,i
 1000 format(" subroutine comp found the distance x=",f9.2,1x,
     & "lies to left of model for horizon number",i5)
      return
   11 j=j+1
   12 if(x.lt.cx(i,j))go to 100
      j=j+1
      if(j.le.jmax)go to 12
      write(jtty,1010)x,i
 1010 format(" subroutine comp found the distance x=",f9.2,1x,
     & "lies to the right of model for horizon number",i5)
   13 iflag=1
      return
  100 x1=cx(i,j-1)
      y1=cy(i,j-1)
```

```
      x2=cx(i,j)
      y2=cy(i,j)
      v=cv(i,j-1)
      j=j-1
      if(x2.le.x1)go to 930
      slope=(y2-y1)/(x2-x1)
      y=y1+slope*(x-x1)
      return
  920 write(jtty,1020)i
 1020 format(1x,"subroutine comp found nc(i) exceeds dimension(50)
     & ",/," in layer",i5)
      go to 13
  930 write(jtty,1030)i
 1030 format(1x,"subroutine comp found cx(i,j+1).le.cx(i,j)",/,
     & 1x," for horizon",i5)
      go to 13
      end


      subroutine crit(xicrt,etacrt,taucrt,xcrt,tcrt)
c
c     subroutine crit calculates the critical reflection point
c     from one-way arrival-time data.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 ext,go,curv,bend1,bend2
      integer*4 optl1,optlr
      dimension dum(30),xp(51),yp(51),vp(51),ap(51),taup(51),t(400),
     &x(400),xi(2),eta(2),ttup(2),ttau(2),gxx(2),
     & xss(2),xicr(2),etacr(2),tupcr(2),taucr(2),gxcr(2),ttest(2)
      dimension fnll(10),fnlr(10),xstar(30),tstar(30),xprim(30),tprim(30),
     & al(400),bl(400),cl(400),d(2),p(400),s(400)
      dimension xir(100),etar(100)
      common/block1/xp,yp,ap,vp,taup
      common/names/filnam
      character*46 filnam
      external select(descriptors),waved(descriptors),rathru(descriptors),
     & csect(descriptors),rayup(descriptors),across(descriptors)
c
c     initialize
c
  100 iflag=0
      dell1=0.
      nstar=10
      nval=3
      nprin=0
      d(1)=0.
      d(2)=0.
      bend1="no"
      bend2="no"
c
c     input data and perform checks on it.
c
    1 write(jtty,1001)filnam
 1001 format(" crit entry 1",3x,"file="a10,"; k*,ext,curv,id,xmin,xmax,v2,
     &+or-delx,delx1,dint,",/,"  <nval,bend1,bend2,nprin,nstar,dell1>",/," *",$)
      read(itty,1002)k,ext,curv,id,xmin,xmax,v2,delx,delx1,dint,
     & nval,bend1,bend2,nprin,nstar,dell1
 1002 format(v)
      if(k.le.0)go to 995
      if((ext.eq."org" .or. ext.eq."com") .and. (curv.eq."lin"
     & .or. curv.eq."spl") .and. id.ge.1 .and. id.le.10 .and. delx.ne.0.
     & .and. v2.gt.0. .and. nval.ge.2 .and. xmin.le.xmax .and. nstar.le.30)
     & go to 2
    4 write(jtty,1003)
 1003 format(" crit condition 1")
      go to 1
c     determine shot location and shot depth depending on whether
c     extension is com or org.
c
    2 if(ext.eq."com")go to 3
      if(k.gt.50)go to 4
      xs=sx(k)
      ds=sd(k)
      go to 5
    3 if(k.gt.10)go to 4
      xs=sxc(k)
      ds=sdc(k)
c
c     if default option in entry 1 was used for gxtry and dell1, determine
c     these variables.
c
    5 gxtry=xs+.000001*sign(xs,delx)
      if(xs.eq.0.)gxtry=.000001*delx
      if(dell1.eq.0.)dell1=abs(delx)
      if(gxtry.eq.xs .or. (gxtry.gt.xs .and. delx.lt.0.) .or.
     & (gxtry.lt.xs .and. delx.gt.0.))go to 4
c
c     set variable f depending on whether data lies to left or
c     right of the shotpoint.
c
      f=1.
      if(delx.gt.0.)f=-1.
c
c     input parameters for extending data to left, and make checks.
c
   25 write(jtty,1005)
 1005 format(" crit entry 1a; optl1*,+or-dx11,+or-fn1r(i)",/," *",$)
      read(itty,1002)optl1,dx11,(fnl1(i),i=1,10)
      if(optl1.lt.0 .or. optl1.gt.3)go to 995
      i=optl1+1
c
c     make checks on extension parameters.
c
```

```
      go to(50,27,28,29),i
c
c     i=2,  for a specific set of data points.
c
   27 n=dxll
      if(n.lt.1 .or. n.gt.5)go to 51
      do 52 i=1,n
      if(i.eq.1)go to 52
      if(fnll(i).le.fnll(i-1))go to 51
   52 continue
      go to 50
   51 write(jtty,1006)
 1006 format(" crit condition 2")
      go to 25
c
c     i=3,  for velocity criteria.
c
   28 if(dxll.lt.0. .and. abs(fnll(1)).gt.0.)go to 50
      go to 51
c
c     i=4,  for least squares criteria.
c
   29 if(dxll.lt.0. .and. fnll(1).gt.0.)go to 50
      go to 51
   50 continue
c
c     input parameters for extending data to right and make checks.
c
   54 write(jtty,1007)
 1007 format(" crit entry 1b; optlr*,+or-dxlr,+or-fnlr(i)",/," *",$)
      read(itty,1002)optlr,dxlr,(fnlr(i),i=1,10)
      if(optlr.lt.0 .or. optlr.gt.3)go to 995
      i=optlr+1
c
c     make checks on extension parameters.
c
      go to(55,56,57,58),i
c
c     i=2,  for specific set of data points.
c
   56 n=dxlr
      if(n.lt.1 .or. n.gt.5)go to 59
      do 70 i=1,n
      if(i.eq.1)go to 70
      if(fnlr(i).le.fnlr(i-1))go to 59
   70 continue
      go to 55
   59 write(jtty,1006)
      go to 54
c
c     i=3,  for the velocity criteria.
c
   57 if(dxlr.gt.0. .and. abs(fnlr(1)).gt.0.)go to 55
      write(jtty,1006)
      go to 54
c
c     i=4,  for least squares criteria.
c
   58 if(dxlr.gt.0. .and. fnlr(1).gt.0.)go to 55
      write(jtty,1006)
      go to 54
   55 continue
c
c     if xmin=xmax from entry 1, make artificial data instead of using actual
c     t-x values.
c
      if(xmin.ne.xmax)go to 6
c
c     input parameters for artificial data and make checks.
c
    8 write(jtty,1008)
 1008 format(" crit entry 2; nobs*,x(i),t(i)",/," *",$)
      read(itty,1002)nobs,(x(i),i=1,nobs),(t(i),i=1,nobs)
      if(nobs.le.1 .or. nobs.ge.200)go to 995
      do 7 i=2,nobs
      if(x(i).gt.x(i-1))go to 7
      write(jtty,1009)
 1009 format(" crit condition 3")
      go to 8
    7 continue
      xmin=x(1)
      xmax=x(nobs)
      go to 9
c
c     select the points of the t-x array which will be used.
c
    6 call select(xmin,xmax,ext,k,nobs,t,x,jwatch,iflag)
      if(iflag.eq.0)go to 93
      write(jtty,1023)
 1023 format(" crit condition 4")
      go to 995
   93 if(jwatch.eq.0)go to 9
      write(jtty,1010)
 1010 format(" crit condition 5")
      go to 8
    9 continue
c
c     extend data to the left according to the parameters established above.
c
      call xtend(x,t,nobs,optll,dxll,fnll,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 10
      go to 11
c
c     extend data to the right according to the parameters established above.
c
   10 call xtend(x,t,nobs,optlr,dxlr,fnlr,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 12
   11 write(jtty,1011)
```

```
 1011 format(" crit condition 6")
      go to 995
c
c     check that shotpoint is within range of the data.
c
   12 if(xs.ge.x(1) .and. xs.le.x(nobs))go to 13
      go to 11
c
c     determine the layer index containing the shotpoint.
c
   13 call comp(xs,1,ye,x1,y1,x2,y2,vdum,j,iflag)
      if(iflag.ne.0)go to 992
      ys=ye-ds
      iu=1
   14 if(iu.eq.id)go to 15
      call comp(xs,iu+1,ye,x1,y1,x2,y2,vdum,j,iflag)
      if(iflag.ne.0)go to 992
      if(ys.gt.ye)go to 15
      iu=iu+1
      go to 14
   15 continue
c
c     determine the distance coordinates (xstar) of the special points
c     which will be traced separately for the users information.
c
      xstar(1)=x(1)
      xstar(nstar)=x(nobs)
      w=nstar-1
      dist=(x(nobs)-x(1))/w
      do 335 i=2,nstar-1
      xstar(i)=xstar(i-1)+dist
  335 continue
c
c     determine the t-x coordinates of a new travel-time curve reduced
c     to the idth. horizon along with the special "star" points.
c
      call waved(x,t,nobs,1,id,delxl,dint,nprin,xstar,nstar,curv,
     & nval,tstar,xprim,tprim,idim,isplin,bend1)
c
c     with the arrival-time data on the idth. horizon, sample it at
c     equally spaced intervals.
c
      if(idim.eq.0 .and. isplin.eq.0)go to 96
      write(jtty,1027)
 1027 format(" crit condition 22")
      go to 995
   96 call eqspace(x,t,nobs,dint,ipt)
      if(ipt.eq.0)go to 16
      write(jtty,1012)
 1012 format(" crit condition 8")
      go to 995
c
c     at this point a filtering routine could be called.
c16    call fltr(x,t,nobs,filt)
c
c     redefine xmin and xmax .   the shotpoint coordinate must still
c     fall within the limits of the reduced data.
c
   16 xmin=x(1)
      xmax=x(nobs)
      if(xs.ge.xmin .and. xs.le.xmax)go to 94
      write(jtty,1024)xmin,xmax
 1024 format(" crit condition 9. limits are:",2f8.1)
      go to 995
c
c     if the curv option is "spl", calculate the spline coefficients.   if
c     option is "lin", no coefficients are necessary.
c
   94 if(curv.eq."spl")call splin1(nobs,0.,x,t,a1,b1,c1,0,d,p,s)
c
c     determine the distance coordinate not to be exceeded
c     in doing calculations.
c
      xfar=xmax
      if(f.gt.0.)xfar=xmin
c
c     initialize.
c
      icr=1
      imess=0
  402 continue
      if(gxtry*f .gt. xfar*f)go to 326
      write(jtty,1013)gxtry,xfar
 1013 format(" crit condition 10.   gxtry=",f9.2," and xfar=",f9.2,/,
     &" you are searching for a solution beyond the limits of the data")
      go to 334
c
c     project a ray into the earth from horizon id at the distance
c     coordinate gxtry.
c
  326 xd=gxtry
c
c     calculate arrival-time and submergence angle at gxtry using either
c     the splin function or the least squares straight line approximation.
c
      if(curv.ne."spl")go to 17
      call splindt(nobs,x,t,a1,b1,c1,xd,treal,deriv)
      go to 18
   17 call lindt(nobs,x,t,xd,nval,treal,deriv)
   18 call comp(xd,id,elev,x1,y1,x2,y2,v0,j,iflag)
      if(iflag.ne.0)go to 992
c
c     calculate the arc sin of the submergence angle.
c
      h=v0*deriv
      if(abs(h).lt.1.)go to 19
      vel=1./deriv
      write(jtty,1014)xd,id,vel
```

```
 1014 format(" crit condition 11 at xd=",f10.1," horizon",i4,
     & " velocity",f10.1)
  103 gxtry=gxtry+delx
      go to 402
c
c     calculate the submergence angle with respect to the vertical.
c
   19 beta=asin(h)
      slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      ad=beta-theta
      if(abs(ad).lt.1.57)go to 81
      write(jtty,1015)xd,id
 1015 format(" crit condition 12 at xd=",f10.1," horizon",i4)
      go to 103
c
c     project ray downward to + or - infinity or through 50 projections,
c     whichever comes first.
c
c     dont permit a vertical ray.
c
   81 if(abs(ad).lt. .00001)ad=-.001*f
      taut=0.
      call rathru(xd,id,ad,taut,nrays,bend2)
c
c     initialize.
c
      idown=1
      ndown=1
      irayd=1
      x11=xp(1)
      y11=yp(1)
      dell=amax1(abs(x11),abs(y11))
      dell=.000001*dell
      ad=ap(1)
      taut=taup(1)
      v1=vp(1)
  315 x22=x11-dell*sin(ad)
      y22=y11-dell*cos(ad)
c
c     if newly established value (x22,y22) requires that compartment
c     boundaries be changed, do so and make other adjustments.
c
      adum=xp(irayd)
      adum1=xp(irayd+1)
      xdum1=amin1(adum,adum1)
      xdum2=amax1(adum,adum1)
      if((x22.ge.xdum1).and.(x22.le.xdum2))go to 313
c
c     change compartment index and associated parameters.
c
      irayd=irayd+1
      x22=xp(irayd)
      y22=yp(irayd)
      ad=ap(irayd)
      taut=taup(irayd)
      v1=vp(irayd)
      dist=(x11-x22)*(x11-x22)+(y11-y22)*(y11-y22)
      dist=sqrt(dist)
      dell=dell-dist
      x11=x22
      y11=y22
      go to 315
c
c     no change in compartment index.
c
  313 dist=(x11-x22)*(x11-x22)+(y11-y22)*(y11-y22)
      dist=sqrt(dist)
      taut=taut+dist/v1
      x11=x22
      y11=y22
c
c     the next two variable names are changed so an earlier version
c     could be copied directly.
c
      xitry=x22
      etatry=y22
c
c     take down-going ray at (xitry,etatry) with parameters taut,ad,v1,
c     and v2 and head it back to the surface through twice the critical angle.
c
      if(v1.lt.v2) go to 82
      write(jtty,2010)v1,v2
 2010 format(" crit condition 13 velocities=",2f8.2)
      go to 995
   82 phi=asin(v1/v2)*(-f)
      au=ad-2.*phi
      theta=-(au+ad)/2.
      if(abs(au).lt.1.5707)go to 300
      write(jtty,2020)xitry
 2020 format(" crit condition 14 at xitry=",f8.1)
      go to 103
  300 continue
      tup=0.
      xi1=xitry
      eta1=etatry
      vuu=v1
      auu=au
      call comp(xitry,id,ydum,x1,y1,x2,y2,v1,ju,iflag)
      if(iflag.ne.0)go to 992
      if(etatry.le.ydum)go to 83
      write(jtty,2030)gxtry,xitry,etatry,ydum
 2030 format(" crit condition 15 gxtry,xitry,etatry,ydum=",4f8.1)
      go to 103
   83 icount=0
c
c     calculate intersection of up-going ray with the idth. horizon,
c     keeping track of travel-times and other parameters.
```

```
c
   30 call hsect(xi1,eta1,auu,vuu,x1,y1,x2,y2,xu,yu,tauh,iflag)
      if(iflag.eq.0)go to 91
      write(jtty,1016)
 1016 format(" crit condition 16")
      go to 103
   91 if(xu.lt.x1.or.xu.ge.x2)go to 84
      if(icount.ne.0)go to 85
      au=auu
   85 taut=taut+tauh
      tup=tup+tauh
      go to 400
c
c     some compartment adjustments are necessary.
c
   84 if(au.gt.0.)go to 87
      if(ju.le.1)go to 88
      xi2=cx(id,ju)
      vl1=cv(id,ju-1)
      go to 86
   87 if(ju.ge.(nc(id)-1))go to 88
      xi2=cx(id,ju+1)
      vl1=cv(id,ju+1)
      go to 86
   88 write(jtty,2040)gxtry
 2040 format(" crit condition 17 from gxtry=",f8.1)
      go to 103
c
c     find intersection with new compartment.
c
   86 call csect(xi1,eta1,auu,vuu,xi2,eta2,au2,vl1,tauc,jrefl,
     &  iflag,jinc,bend2)
      if(iflag.eq.0)go to 95
      write(jtty,1031)
 1031 format(" crit condition 18")
      go to 995
   95 if(icount.eq.0)au=auu
      icount=1
      ju=ju+jinc
      x1=cx(id,ju)
      y1=cy(id,ju)
      x2=cx(id,ju+1)
      y2=cy(id,ju+1)
      taut=taut+tauc
      tup=tup+tauc
      xi1=xi2
      eta1=eta2
      auu=au2
      vuu=vl1
      go to 30
c
c     successful intersection of reflected ray with the idth. horizon
c     has occurred.
c
  400 continue
c
c     initialize.
c
      tdum=0.
c
c     move ray upwards from the idth. horizon to horizon number 1.
c
      call rayup(xu,id,iu,xi1,eta1,x0,y0,tdum,a0,v0,j0,iflag,bend1,0)
      if(iflag.eq.0)go to 92
      write(jtty,1017)gxtry
 1017 format(" crit condition 19 at gxtry=",f8.1)
      go to 103
   92 h=(y0-ys)/cos(a0)
      deltat=h/v0
      taut=taut-deltat+tdum
      tup=tup-deltat+tdum
      xstry=x0-h*sin(a0)
c
c     ray is back to surface at the distance coordinate xstry with
c     travel-times taut and tup.   put these newly found values into
c     an array.
c
      xi(idown)=xitry
      eta(idown)=etatry
      ttup(idown)=tup
      ttau(idown)=taut
      taut=taut-tup
      xss(idown)=xstry
      if(idown.eq.2)go to 317
c
c     does xstry fall on the correct side of the shotpoint coordinate xs?
c
      if((xstry*f).lt.(xs*f))go to 328
c
c     no.     increment and try again.
c
      gxtry=gxtry+delx
      go to 402
c
c     yes.    continue downward.
c
  328 idown=2
      dell=delll
      go to 315
  317 if((xstry*f).ge.(xs*f))go to 323
c
c     the shotpoint has not yet been successfully straddled by two points.
c     switch variables around and try again.
c
      xi(1)=xi(2)
      eta(1)=eta(2)
      ttup(1)=ttup(2)
      ttau(1)=ttau(2)
```

```
      xss(1)=xss(2)
      dell=dell1
      ndown=ndown+1
      if(ndown.lt.301)go to 315
      write(jtty,1018)gxtry
 1018 format(" crit condition 20 at gxtry=",f9.2)
      go to 995
c
c     a successful pair of points xss(1) and xss(2), straddling the shotpoint
c     xs have been found.   through interpolation determine the parameters
c     at the shotpoint.
c
  323 g=(xss(2)-xs)/(xss(2)-xss(1))
      xicr(icr)=xi(2)-g*(xi(2)-xi(1))
      etacr(icr)=eta(2)+g*(eta(1)-eta(2))
      tupcr(icr)=ttup(2)-g*(ttup(2)-ttup(1))
      taucr(icr)=ttau(2)-g*(ttau(2)-ttau(1))
      gxcr(icr)=gxtry
      ttest(icr)=treal
      if(icr.eq.2)go to 324
c
c     first time through.
c
      if(taucr(1).le.treal)go to 330
      if(imess.eq.1)go to 370
      write(jtty,1019)gxtry,taucr(1),treal,xicr(1),etacr(1),tupcr(1)
 1019 format(" crit condition 21. values are:",/,6f10.2)
      return
  330 if(imess.eq.1)imess=0
      icr=2
c
c     increment and try again.
c
  370 gxtry=gxtry+delx
      go to 402
  324 if(taucr(2).gt.ttest(2))go to 331
c
c     reflection time still too small.   switch variables and try again.
c
      gxtry=gxtry+delx
      xicr(1)=xicr(2)
      etacr(1)=etacr(2)
      taucr(1)=taucr(2)
      tupcr(1)=tupcr(2)
      gxcr(1)=gxcr(2)
      ttest(1)=ttest(2)
      go to 402
c
c     got a winner!!   do a little interpolation.
c
  331 a=(taucr(2)-taucr(1))/(gxcr(2)-gxcr(1))
      b=(ttest(2)-ttest(1))/(gxcr(2)-gxcr(1))
      if(a.ne.b)go to 325
      xcrt=gxcr(1)
      go to 333
  325 xcrt=gxcr(1)+(ttest(1)-taucr(1))/(a-b)
  333 tcrt=a*(xcrt-gxcr(1))+taucr(1)
      g=(gxcr(2)-xcrt)/(gxcr(2)-gxcr(1))
      xicrt=xicr(2)-g*(xicr(2)-xicr(1))
      etacrt=etacr(2)+g*(etacr(1)-etacr(2))
      taucrt=tupcr(2)-g*(tupcr(2)-tupcr(1))
c
c     write the answer.
c
      write(jtty,2070)xicrt,etacrt,taucrt,xcrt,tcrt
 2070 format(" good news: the critical points have been found.they are"
     &,/," critical subsurface coordinates xicrt=",f9.2," etacrt=",
     &f9.2,/," critical subsurface arrival time taucrt=",f9.2,/,
     &" critical distance on surface xcrt=",f9.2,/,
     &" critical surface arrival time tcrt=",f9.2)
      write(jtty,1020)
 1020 format(" do you want to try for another solution? yes or no:",$)
      read(itty,1021)go
 1021 format(a3)
      if(go.eq."yes")go to 332
      go to 334
c
c     if another critical reflection is to be attempted, reinitialize and
c     increment.
c
  332 imess=1
      icr=1
      gxtry=gxcr(2)+delx
      go to 402
  992 write(jtty,1022)
 1022 format(" crit condition 7")
      go to 995
  334 write(jtty,1026)
 1026 format(" try across for solutions along t-x graph? yes or no:",$)
      read(itty,1025)go
 1025 format(a3)
      if(go.ne."yes")go to 995
c
c     try subroutine across for other solutions on the horizon being calculated.
c
      call across(xicrt,etacrt,taucrt,xcrt,tcrt,id,xfar,x,t,nobs,
     & curv,al,bl,cl,nval,nstar,xstar,xprim,tstar,tprim,xir,etar,bend2)
  995 write(jtty,1030)
 1030 format(" do you wish to try anothor shot? yes or no:",$)
      read(itty,1021)go
      if(go.eq."yes")go to 100
      return
      end


      subroutine csect(xp,yp,ap,vp,xpp,ypp,app,vpp,tauc,jrefl,
     & iflag,jinc,bend)
c
```

```
c     given a point on a ra·· with coordinates (xp,yp),angle of propagation ap,
c     velocity vp, a vertical compartment boundary with distance coordinate
c     xpp, and the velocity on the opposite side of the compartment boundary
c     being vpp,calculate the elevation coordinate ypp, of the intersection
c     point of the ray with the compartment boundary, the angle of emergence
c     app, across the compartment boundary, and the travel-time tauc, of the
c     ray between (xp,yp) and (xpp,ypp).   bend is either no or yes.    if no,
c     ray passes through compartment boundary without undergoing either
c     reflection or refraction.    jrefl and iflag are normally 0.   jrefl
c     becomes 1 if reflection occurs at compartment boundary.   iflag becomes 1
c     either if a vertical ray occurs (ap=0) and hence an intersection with
c     a compartment is impossible, or a compartment velocity is zero or
c     negative.
c
      character*3 bend
      data itty/5/,jtty/6/
      iflag=0
      jrefl=0
      if(vp.le.0.)go to 900
      if(ap.eq.0.)go to 910
c
c     calculate tne elevation coordinate ypp, of intersection point of ray
c     with vertical boundary line at x=xpp, and also the travel-time
c     tauc, along the ray segment.
c
      ca=cos(ap)/sin(ap)
      ypp=ca*(xpp-xp)+yp
      w=(xp-xpp)*(xp-xpp)+(yp-ypp)*(yp-ypp)
      tauc=sqrt(w)/vp
      if(bend.ne."no")go to 12
c     do not bend ray at compartment boundary.
c
      app=ap
      go to 13
c
c     calculate the refraction or reflection angle.
c
   12 f=abs((vpp/vp)*cos(ap))
      if(f.le.1.)go to 10
c
c     reflection has occurred
c
      app=-ap
      vpp=vp
      jinc=0
      jrefl=1
      return
c
c     refraction has occurred.
c
   10 app=acos(f)*(ap/abs(ap))
c
c     determine whether ray enters compartment to left (jinc=-1) or
c     compartment to right (jinc=1).
c
   13 jinc=-1
      if(ypp.le.yp)go to 11
      if(ap.gt.0.)jinc=1
      return
   11 if(ap.lt.0.)jinc=1
      return
  900 write(jtty,1010)
 1010 format(1x,"subroutine csect found a compartment velocity is=0")
      go to 300
  910 write(jtty,1020)
 1020 format(1x,"a vertical ray found in csect.intersection impossible")
  300 iflag=1
      return
      end


      subroutine datain(ifirst)
c
c      input data into either files filenam.org
c      filenam.com or filenam.lay
c
      external io(descriptors)
      dimension ihdr(20)
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 route,ext
      character*5 idigt,indig
       dimension route(3)
      logical ifirst
      data route /"org","com","lay"/
      data idigt/"digit"/
 1000 format(1x,"type file extension either org,com,or lay: ",$)
 1010  format(a3)
 1020 format(1x,"your extension invalid: try again: ",$)
 1070 format(1x,"incorrect input try again")
 1080 format(1x,"datain complete")
c
c      direct program to desired file. i.e. org com or lay
c
      write(jtty,1000)
   10 read(itty,1010)ext
      do 11 i=1,3
      if (ext.eq.route(i))go to 20
   11 continue
      write(jtty,1020)
      go to 10
   20 go to (30,40,50)i
c
c      data for filenam.org being input.
c      determine whether data is in segment digit.out or will be
```

```
c      input from terminal.
c
   30 write(jtty,1001)
 1001 format(" enter  the word digit   if data digitized.",/,
     &" anything else if not: ",$)
      read(itty,1102)indig
 1102 format(a5)
      iity=5
      if(indig.ne.idigt)go to 34
      if(.not.ifirst)go to 35
      ifirst=.false.
      call io("attach","file15","vfile_ ","digit.out")
   35 iity=15
      read(iity,1028)ihdr
 1028 format(20a4)
      write(jtty,1029)ihdr
 1029 format(1x,20a4)
      go to 31
   34 write(jtty,2000)
 2000 format(" datain entry 1; k*,ng(k),sx(k),sd(k),offset(k)",/," *",$)
   31 read(iity,3000)i,l,b,c,d
 3000 format(v)
      if(i.lt.1)go to 990
      if(i.gt.50)i=50
      if(l.gt.24)l=24
      ng(i)=l
      sx(i)=b
      sd(i)=c
      offset(i)=d
      do 33 j=1,24
      gx(i,j)=0.
   33 gt(i,j)=0.
      if(indig.eq.idigt)go to 36
      write(jtty,1090)
 1090 format(" gx(k,j):",$)
   36 read(iity,3000)(gx(i,j),j=1,ng(i))
      if(indig.eq.idigt)go to 37
      write(jtty,1091)
 1091 format(" gt(k,j):",$)
   37 read(iity,3000)(gt(i,j),j=1,ng(i))
c
c      determine distance coordinate of last data point and print it.
c
      xlast=gx(1,1)
      dx=100000.
      do 32 j=2,ng(i)
      if(gx(i,j).ne.0.)dx=gx(i,j)
   32 xlast=xlast+dx
      write(jtty,2001)i,xlast
 2001 format(" for shot index=",i5," xlast=",f12.2)
      if(indig.eq.idigt)go to 31
      write(jtty,2002)
 2002 format(" next set of data please. 0; or -n; for return to mainline")
      go to 34
c
c      data from filenam.com being input from terminal
c
   40 write(jtty,2000)
   41 read(itty,3000)i,l,b,c,d
      if(i.lt.1)go to 990
      if(i.gt.10)i=10
      if(l.gt.200)l=200
      ngc(i)=l
      sxc(i)=b
      sdc(i)=c
      offsc(i)=d
      do 43 j=1,200
      gxc(i,j)=0.
   43 gtc(i,j)=0.
      write(jtty,1090)
      read(itty,3000)(gxc(i,j),j=1,ngc(i))
      write(jtty,1091)
      read(itty,3000)(gtc(i,j),j=1,ngc(i))
c
c      determine distance coordinate of last data point and print it.
c
      xlast=gxc(i,1)
      dx=100000.
      do 42 j=2,ngc(i)
      if(gxc(i,j).ne.0.)dx=gxc(i,j)
   42 xlast=xlast+dx
      write(jtty,2001)i,xlast
      write(jtty,2002)
      go to 41
c
c      data for filenam.lay being input from terminal
c
   50 write(jtty,1050)
 1050 format(" datain entry 2; i*,nc(i):",$)
   52 read(itty,3000)i,nc(i)
      if(i.le.0)go to 990
      if(i.gt.10.or.nc(i).gt.50.or.i.lt.1.or.nc(i).lt.1)go to 960
      do 51 j=1,50
      cx(i,j)=0.
      cy(i,j)=0.
      cv(i,j)=0.
   51 continue
      write(jtty,1092)
 1092 format(" cx(i,j):",$)
      read(itty,3000)(cx(i,j),j=1,nc(i))
      write(jtty,1093)
 1093 format(" cy(i,j):",$)
      read(itty,3000)(cy(i,j),j=1,nc(i))
      write(jtty,1094)
 1094 format(" cv(i,j):",$)
      read(itty,3000)(cv(i,j),j=1,nc(i))
      write(jtty,2003)
 2003 format(" next layer please.  type 0; or -n; for return to mainline")
```

```
      go to 50
  960 write(jtty,1070)
      go to 50
  800 write(jtty,2050)i
 2050 format(" e.o.f. encountered @ index",i4)
c
c      output data to disk
c
  990 call output
      write(jtty,1080)
      return
      end


      subroutine digit
      implicit double precision(x,y)
      double precision xydat(2,8)
      dimension ms(8),ihdr(80)
      external io(descriptors),close_file(descriptors)
      common/seiscm/x1,y1,x2,y2,x3,y3,xms1,xms2,xraw,yraw,xtim
      data isl/";"/,ibl/" "/,i0/"0"/
      data i0cs/"0;"/
      call io("attach","file09","vfile_ ","digit.in")
      call io("attach","file10","vfile_ ","digit.out")
   31 read(9,19,end=100)ihdr
      write(10,19)ihdr
      write(6,60)ihdr
   60 format(2x,"header:",2x,40a1,/,10x,40a1)
      write(6,1003)
 1003 format(" index readtime  digitime   error    readtime  digitime",
     &"  error")
   17 read(9,19,end=100) ihdr
   19 format(80a1)
      write(10,19) ihdr
      if(ihdr(1).eq.i0.and.ihdr(2).eq.isl) go to 31
      backspace 9
      read(9,1001)kk,dum1,dum2,dum3,dum4
 1001 format(v)
   21 read(9,19,end=110) ihdr
      write(10,19) ihdr
      do 27 i=1,80
      if(ihdr(i).eq.isl) go to 14
   27 continue
      go to 21
   14 read(9,15,end=50) njug,x1,y1,x2,y2,xms1,x3,y3,xms2
   15 format(i2,1x,2(2f4.2,1x),f4.0,1x,2f4.2,1x,f4.0)
      call arvscl
      ijug=0
   16 read(9,18,end=40) idat,xydat
   18 format(a3,8(2f4.2,1x))
      if(idat.ne.i0cs) go to 11
c
c   header card found when expecting data.
c
      write(6,12) njug,ijug
   12 format(2x,"****error - expected ",i2," jugs for data set.",
     &/,6x,"only found ",i2," before next 0; card.****")
      backspace 9
   10 write(10,13) isl
   13 format(a1)
      go to 17
c  ***************************************************************************
c
c   if this program is to output header cards for ackermann card deck,
c    they should be output here.....
c
c  ***************************************************************************
   11 do 20 i=1,8
      xraw=xydat(1,i)
      yraw=xydat(2,i)
      if(xraw.eq.0.0d0.and.yraw.eq.0.0d0) go to 23
      call arvtm
      ms(i)=xtim+0.5d0
      ijug=ijug+1
      if(ijug.eq.njug) go to 25
   20 continue
      write(10,22) ms
   22 format(8(i9,","))
      go to 16
   23 write(6,24) njug,ijug,ijug
   24 format(2x,"****error - expected ",i2," jugs for data set.",
     &/,6x,"only found ",i2," before blank field.  will output ",
     &i2," values.****")
      iout=i-1
      if(iout.eq.0) go to 10
      write(10,30) (ibl,ms(i),i=1,iout)
      go to 10
c
c   @ 25, output i good ms values, followed by ";".
c    note that i could be 8. (1 .le. i .le. 8)
c
   25 iout=i
   26 if(iout.eq.8) go to 35
      write(10,30) (ibl,ms(i),i=1,iout),isl
   30 format(8(a1,i8,","))
      go to 91
   35 write(10,36) ms
   36 format(7(i9,","),i8,";")
   91 read(9,62,end=110) x4,y4,x5,y5,xms3,xms4
   62 format(2(2f4.2),1x,2(f4.0,1x))
      xraw=x4
      yraw=y4
      call arvtm
      xtime1=xtim
      tc1=xms3-xtim
      xraw=x5
      yraw=y5
      call arvtm
```

```
      xtime2=xtim
      tc2=xms4-xtim
      write(6,1004)kk,xms3,xtime1,tc1,xms4,xtime2,tc2
 1004 format(2x,i2,2(4x,f6.0,4x,f6.0,2x,f8.1))
      go to 17
   40 write(6,41) njug,ijug
   41 format(2x,"****error - expected ",i2," jugs for data set.",
     &/,6x,"only found ",i2," before eof.****")
      if(ijug.ne.0) write(10,13) isl
   45 write(10,4) i0cs
    4 format(a3)
      go to 110
   50 write(6,51)
   51 format(2x,"****error - no digitized data present for current ",
     &/,14x,"shot profile before end of file.****")
      write(10,13) isl
      go to 45
  100 write(6,101)
  101 format(2x,"----normal end of file.----")
      call close_file("file09")
      call io("detach","file09")
      call close_file("file10")
      call io("detach","file10")
      stop
c   ***********************************************************************
c
c   if this program is to output final supplemental cards for ackermann
c    deck, they should be output here ( @ 110).
c
c   ***********************************************************************
  110 write(6,111)
  111 format(2x,"****abnormal termination.****")
      call close_file("file10")
      call io("detach","file10")
      stop
      end


      subroutine arvscl
      implicit double precision(a-h,o-z)
      common/seiscm/xy(2,3),xms1,xms2,xraw,yraw,xtim
      xbas=xy(1,1)-xy(1,2)
      ybas=xy(2,2)-xy(2,1)
      rbas=dsqrt(xbas*xbas+ybas*ybas)
      wsin=xbas/rbas
      wcos=ybas/rbas
      xscl=xy(1,3)-xy(1,1)
      yscl=xy(2,3)-xy(2,1)
      xdist=xscl*wcos+yscl*wsin
      xslop=(xms2-xms1)/xdist
      return
      entry arvtm
      x0=xraw-xy(1,1)
      y0=yraw-xy(2,1)
      xrot=x0*wcos+y0*wsin
      xtim=xms1+xslop*xrot
      return
      end


      subroutine dum1
      return
      end


      subroutine eqspace(x,t,nobs,dint,idim)
c
c     given t-x points defined by arrays (x(i),t(i)),i=1,nobs which
c     are in increasing value of x but not necessarily of constant interval
c     between successive points, determine a new array also defined by
c     (x(i),t(i)) which approximates the old but which have x values
c     equally spaced with interval approximately dint.
c
      dimension x(400),t(400),xx(400),tt(400)
      idim=0
c
c     if array contains less than two data points, get out.
c
      if(nobs.lt.2)go to 12
c
c     divide array into an integral number of n equally spaced intervals
c     and guarantee that it will include at least two points.
c
      dist=x(nobs)-x(1)
      n=dist/dint+1
      if(n.eq.1)n=2
      if(n.le.400)go to 11
   12 idim=1
      return
   11 w=n-1
      dint1=dist/w
c
c     initialize tha number of points in new array.
c
      npts=1
c
c     iobs keeps track of array indecies between which interpolation
c     takes place.
c
      iobs=2
c
c     determine first and last points of array.
c
      xx(1)=x(1)
      tt(1)=t(1)
      xx(n)=x(nobs)
      tt(n)=t(nobs)
c
```

```
c     (x1,t1) and (x2,t2) are the points between which interpolation
c     is being considered.
c
      x1=x(1)
      t1=t(1)
      x2=x(2)
      t2=t(2)
c
c     determine the next x to be considered in the xx(i) array
c     and increment npts by 1.
c
    1 x3=xx(npts)+dint1
      npts=npts+1
c
c     if npts=n, all done
c
      if(npts.eq.n)go to 6
    2 if(x3.gt.x2)go to 3
c
c     xx(npts) lies between x1 and x2.
c
      tt(npts)=((t2-t1)/(x2-x1))*(x3-x1)+t1
      xx(npts)=x3
      go to 1
c
c     x3 lies outside of interval x1, x2.  move one position.
c
    3 iobs=iobs+1
      x1=x2
      t1=t2
      x2=x(iobs)
      t2=t(iobs)
      go to 2
c
c     all done.  put new points into old array.
c
    6 nobs=n
      do 8 i=1,nobs
      x(i)=xx(i)
    8 t(i)=tt(i)
      return
      end


      subroutine fudge
c
c     adds a constant to some of the variables in common block datal
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10),var(7)
      common/datal/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 var,go
      integer snum
      data var/"gt","gx","cx","cy","cv","gtc","gxc"/
c
 1000 format(" fudge entry 1; const*,(either gt,gx,cx,cy,cv,gtc,gxc),
     &k,is,ie",/," *",$)
 1400 format(" variable ",a3,"  shot or layer no. ",i4,"  indeces ",
     & i3," to ",i3,/," changed with const = ",f10.1)
 1600 format(" input parameters are out of range. try again!")
c
      go to 10
   15 write(jtty,1400)var(nvar),snum,is,ie,const
   10 snum=1
      write(jtty,1000)
      read(itty,350)const,go,snum,is,ie
      if(const.eq.0.)go to 990
  350 format(v)
      do 351 nvar=1,7
      if(go.eq.var(nvar))go to 352
  351 continue
      go to 900
c
c     check input parms data
c
  352 if(snum.lt.1.or.is.lt.1.or.ie.lt.1.or.is.gt.ie
     & .or.nvar.lt.1.or.nvar.gt.7)go to 900
      if(nvar.lt.3.and.(snum.gt.50.or.ie.gt.24))go to 900
      if((nvar.gt.2.and.nvar.lt.6)
     & .and.(snum.gt.10.or.ie.gt.50))go to 900
      if((nvar.eq.6.or.nvar.eq.7).and.(snum.gt.10.or.ie.gt.200))
     & go to 900
c
c     branch to proper variable
c
      go to (50,70,170,190,210,230,250)nvar
   50 if(ie.gt.ng(snum))go to 900
      do 60 j=is,ie
      gt(snum,j)=gt(snum,j)+const
   60 continue
      go to 15
   70 if(ie.gt.ng(snum))go to 900
      do 80 j=is,ie
      gx(snum,j)=gx(snum,j)+const
   80 continue
      go to 15
  170 if(ie.gt.nc(snum))go to 900
      do 200 j=is,ie
      cx(snum,j)=cx(snum,j)+const
  200 continue
      go to 15
  190 if(ie.gt.nc(snum))go to 900
      do 220 j=is,ie
      cy(snum,j)=cy(snum,j)+const
  220 continue
      go to 15
```

```
  210 if(ie.gt.nc(snum))go to 900
      do 240 j=is,ie
      cv(snum,j)=cv(snum,j)+const
  240 continue
      go to 15
  230 if(ie.gt.ngc(snum))go to 900
      do 260 j=is,ie
      gtc(snum,j)=gtc(snum,j)+const
  260 continue
      go to 15
  250 if(ie.gt.ngc(snum))go to 900
      do 280 j=is,ie
      gxc(snum,j)=gxc(snum,j)+const
  280 continue
      go to 15
  900 write(jtty,1600)
      go to 10
  990 call output
      return
      end


      subroutine hsect(xp,yp,ap,vp,x1,y1,x2,y2,xq,yq,tauh,iflag)
c
c     given a ray with starting point (xp,yp) propagating at an angle ap
c     with the vertical at velocity vp, and a horizon defined by
c     coordinates (x1,y1), and (x2,y2).    hsect calculates the
c     intersection point (xq,yq) between the ray and the horizon,  and the
c     travel-time tauh, between (xp,yp) and (xq,yq).   iflag is normally zero
c     and becomes 1 if the velocity vp is less than or equals zero, or if
c     the ray and horizon are parallel, and hence no intersection can
c      be found.
c
      jtty=6
      iflag=0
      if(vp.le.0.)go to 900
      if(ap.ne.0.)go to 11
      xq=xp
      go to 12
   11 slope=(y2-y1)/(x2-x1)
      ca=cos(ap)/sin(ap)
      if(ca.eq.slope)go to 910
      xq=(xp*ca-yp-x1*slope+y1)/(ca-slope)
   12 yq=slope*xq-x1*slope+y1
      w=(xp-xq)*(xp-xq)+(yp-yq)*(yp-yq)
      tauh=sqrt(w)/vp
      return
  900 write(jtty,1000)
 1000 format(" subroutine hsect finds a compartment velocity equal to zero")
      go to 30
  910 write(jtty,1010)
 1010 format(" in subroutine hsect, a ray and a layer are parallel")
   30 iflag=1
      return
      end


      subroutine input
c
c     reads original time-distance data,combined time-distance data
c     and layer data from three disk files if they exist
c     the first six characters of the file name are typed on the tty
c     original data has org for the file extension
c     combined data has com for the file extension
c     layer data has lay for the file extension
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     & cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     & sdc(10),ngc(10),offset(50),offsc(10)
      external mafina(descriptors)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/names/filnam,filorg,filcom,fillay,idsk
      character*6 filnam
      character*10 filorg,filcom,fillay
      iffil=0
  100 format(" type input file name: ",$)
  200 format(v)
  300 format(2i3,3f11.2,/(2f11.2))
  400 format(2i3,(3f11.2))
  500 format(" input complete from disk file ",a10)
  600 format(" *ERROR* cannot have a blank filename!",/)
  700 format(" *ERROR* cannot have leading blanks in the filename!",/)
  710 format(" ******* you will be creating file ",a6)
c
c  get the filename and append the extension to it
c
  800 write(jtty,100)
      read(itty,200,end=2900) filnam
      call mafina(filnam,filorg,filcom,fillay,ierror)
      if (ierror.eq.1)go to 900
      if (ierror.eq.2)go to 1000
      go to 1100
  900 write(jtty,600)
      go to 800
 1000 write(jtty,700)
      go to 800
 1100 do 1200 k=1,50
 1200 ng(k)=0
      do 1300 k=1,10
 1300 ngc(k)=0
      do 1400 k=1,50
 1400 nc(k)=0
c
c     read time-distance data
c
      open(idsk,file=filorg,form="formatted",access="sequential",
     & mode="in",err=1700)
```

```
      write(jtty,500)filorg
      iffil=1
 1500 read(idsk,300,end=1600) i,ng(i),sx(i),sd(i),offset(i),(gx(i,j),
     &  gt(i,j),j=1,ng(i))
      go to 1500
 1600 close(idsk)
      go to 2000
c
c     branch to here if file does not exist and zero all varibles
c
 1700 continue
      do 1900 i= 1,50
      ng(i)=0
      sx(i)=0.0
      sd(i)=0.0
      offset(i)=0.0
      do 1800 j=1,24
      gt(i,j)=0.0
      gx(i,j)=0.0
 1800 continue
 1900 continue
c
c     read in combined time-distance data
c
 2000 open(idsk,file=filcom,form="formatted",access="sequential",
     & mode="in",err=2300)
      write(jtty,500)filcom
      iffil=1
 2100 read(idsk,300,end=2200) i,ngc(i),sxc(i),sdc(i),offsc(i),(gxc(i,j),
     &  gtc(i,j),j=1,ngc(i))
      go to 2100
 2200 close(idsk)
      goto 2600
c
c     branch to here if file does not exist and zero all varibles
c
 2300 continue
      do 2500 i=1,10
      ngc(i)=0
      sxc(i)=0.0
      sdc(i)=0.0
      offsc(i)=0.0
      do 2400 j=1,200
      gxc(i,j)=0.0
      gtc(i,j)=0.0
 2400 continue
 2500 continue
c
c     read in layer and compartment data
c
 2600 open(idsk,file=fillay,form="formatted",access="sequential",
     & mode="in",err=2900)
      write(jtty,500)fillay
      iffil=1
 2700 read(idsk,400,end=2800)i,nc(i),(cx(i,j),cy(i,j),cv(i,j),j=1,nc(i))
      go to 2700
 2800 close(idsk).
      goto 3300
c
c     branch to here if file does not exist and zero all varibles
c
 2900 do 3000 i=1,50
      nc(i)=0
 3000 continue
      do 3200 i=1,10
      do 3100 j=1,15
      cx(i,j)=0.0
      cy(i,j)=0.0
      cv(i,j)=0.0
 3100 continue
 3200 continue
 3300 if(iffil.eq.0)write(jtty,710)filnam
      return
      end


      subroutine interp(x,t,nobs,xp,time,iterp)
c
c     given t-x arrays (x(i),t(i),i=1,nobs)  and the distance coordinate
c     xp,  the variable time is calculated (provided that xp lies within
c     the :ange of the x(i))  by interpolating between the two x(i)
c     between which xp falls.  iterp is normally zero, and becomes one
c     if xp lies outside the range of the x(i).
c
      dimension x(400),t(400)
      iterp=0
      if(xp.ge.x(1) .and. xp.le.x(nobs))go to 1
      iterp=1
      return
    1 do 2 i=2,nobs
      if(xp.gt.x(i))go to 2
      go to 3
    2 continue
    3 time=t(i-1)+((t(i)-t(i-1))/(x(i)-x(i-1)))*(xp-x(i-1))
      return
      end


      subroutine lindt(nobs,x,t,xp,nval,time,deriv)
c
c     given t-x arrays (x(i),t(i)),i=1,nobs, and a distance coordinate xp,
c     within the range of x(i), calculate the arrival time time, and
c     derivative deriv, at xp which approximate the plot of the
c     t-x points.  the criteria used is a least squares straight line
c     through a subset of the x(i),t(i) containing nval of the nobs
c     points such that xp lies approximately in the center of the subset.
c
      dimension x(nobs),t(nobs),xx(30),tt(30)
```

```
c
c     initialize.
c
      n=nval
c
c     check number of points available for least squares computation,
c     and if incompatable, change them.
c
      if(nobs.ge.2 .and. nval.ge.2 .and. nval.le.nobs .and.
     & xp.ge.x(1) .and. xp.le.x(nobs))go to 20
      write(6,1001)xp
 1001 format(" lindt condition 1 at xp=",f9.2)
      if(nobs.lt.2)nobs=2
      if(nval.lt.2)n=2
      if(nval.gt.nobs)n=nobs
c
c     determine the interval within the range of points which will
c     be used for approximating straight lines.
c
   20 delx=x(2)-x(1)
      w=n-1
      xint=w*delx
      xint2=xint/2.
c
c     the first point to be used for l.s. fitting is half an interval
c     behind xp.
c
      xstart=xp-xint2
c
c     if xstart lies outside range of points, use the "tail end" points.
c
      if(xstart.gt.(x(1)-xint2))go to 1
      nstart=1
      go to 10
    1 if(xstart.lt.(x(nobs)-xint))go to 2
      nstart=nobs-n+1
      go to 10
c
c     determint the index nstart, of the point in the array x(i) which
c     lies immediately to the right of xp.
c
    2 do 3 i=2,nobs
      nstart=i
      if(xstart.le.x(i))go to 4
    3 continue
c
c     and if it is nearer the prior point, use it.
c
    4 if((x(nstart)-xstart).gt.(delx/2.))nstart=nstart-1
c
c     put the points for l.s. fitting into arrays.
c
   10 do 5 i=1,n
      xx(i)=x(nstart)
      tt(i)=t(nstart)
      nstart=nstart+1
    5 continue
c
c     do least squares calculations.
c
      c1=n
      c2=0.
      c3=0.
      c5=0.
      c6=0.
      do 6 i=1,n
      c2=c2+xx(i)
      c3=c3+tt(i)
      c5=c5+xx(i)*xx(i)
      c6=c6+xx(i)*tt(i)
    6 continue
      c4=c2
      a2=(c1*c6-c3*c4)/(c1*c5-c2*c4)
      a1=(c3-c2*a2)/c1
c
c     calculate arrival time and derivative.
c
      time=a1+a2*xp
      deriv=a2
      return
      end


      subroutine lslin(npts,x,t,a1,a2)
c
c      calculate the coefficients a1 and a2 for a least squares straight
c      line through points x(i),t(i) i=1,npts. t=a1+a2x
c
      dimension x(1),t(1)
      c1=npts
      c2=0.
      c3=0.
      c5=0.
      c6=0.
      do 1 i=1,npts
      c2=c2+x(i)
      c3=c3+t(i)
      c5=c5+x(i)*x(i)
      c6=c6+x(i)*t(i)
    1 continue
      c4=c2
      a2=(c1*c6-c3*c4)/(c1*c5-c2*c4)
      a1=(c3-c2*a2)/c1
      return
      end


      subroutine mafina(fn,fnorg,fncom,fnlay,ierror)
```

```
c
c  subroutine to append the file extensions on the root name,
c  eliminating the right justification problem on the honeywell
c
c  fn    = character*6
c  fnorg = character*10
c  fncom = character*10
c  fnlay = character*10
c  ierror = interger ( 1 = set if fn is blank )
c                    ( 2 = set if fn has leading blanks)
c
      character*10 fnorg,fnlay,fncom
      character*6 fn
      character*1 fnt(6)
      ierror=0
c
c  decode fn into a 6a1 so it can be checked for blanks return if all bl
c
      if(fn.ne."      ")goto 100
      fnorg="          "
      fncom="          "
      fnlay="          "
      ierror=1
      return
  100 decode(fn,200)fnt
  200 format(6a1)
c
c  check for blanks and append the extension after first one
c
      do 300 i = 1,6
      if(fnt(i).eq." ")goto 400
  300 continue
c
c  if it gets to here fn is 6 characters long
c
  400 goto (500,600,1000,1400,1800,2200,2600),i
c
c  this is a precautionary (stupid) check should never get to here
c
      stop  "*ERROR* i out of range"
  500 ierror=2
      fnorg="          "
      fnlay="          "
      fncom="          "
      return
  600 encode(fnorg,700)fn
  700 format(a1,".org     ")
      encode(fncom,800)fn
  800 format(a1,".com     ")
      encode(fnlay,900)fn
  900 format(a1,".lay     ")
      goto 3000
 1000 encode(fnorg,1100)fn
 1100 format(a2,".org    ")
      encode(fncom,1200)fn
 1200 format(a2,".com    ")
      encode(fnlay,1300)fn
 1300 format(a2,".lay    ")
      goto 3000
 1400 encode(fnorg,1500)fn
 1500 format(a3,".org    ")
      encode(fncom,1600)fn
 1600 format(a3,".com    ")
      encode(fnlay,1700)fn
 1700 format(a3,".lay    ")
      goto 3000
 1800 encode(fnorg,1900)fn
 1900 format(a4,".org    ")
      encode(fncom,2000)fn
 2000 format(a4,".com    ")
      encode(fnlay,2100)fn
 2100 format(a4,".lay    ")
      goto 3000
 2200 encode(fnorg,2300)fn
 2300 format(a5,".org    ")
      encode(fncom,2400)fn
 2400 format(a5,".com    ")
      encode(fnlay,2500)fn
 2500 format(a5,".lay    ")
      goto 3000
 2600 encode(fnorg,2700)fn
 2700 format(a6,".org")
      encode(fncom,2800)fn
 2800 format(a6,".com")
      encode(fnlay,2900)fn
 2900 format(a6,".lay")
 3000 return
      end


      subroutine output
c
c     outputs time-distance,combined time-distance and layer data
c     to three disk files with extensions of org,com,lay,respectively
c     if there is data in the arrays to output
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     & cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     & sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/names/filnam,filorg,filcom,fillay,idsk
      character*6 filnam
      character*10 filorg,filcom,fillay
  100 format(2i3,3f11.2,/(2f11.2))
  200 format(2i3,(3f11.2))
  300 format(" output of file ",a10" completed to disk")
  310 format(" ******* you have no data to output to disk")
```

```
      iffil=0
      open=0
      i=0
c
c     output time-distance data
c
  400 i=i+1
      if(i.gt.50)go to 600
      if(ng(i).eq.0)go to 400
      if(open.ne.0)goto 500
      open(idsk,file=filorg,form="formatted",access="sequential",
     & mode="out")
      open=1
      write(jtty,300)filorg
      iffil=1
  500 write(idsk,100)i,ng(i),sx(i),sd(i),offset(i),(gx(i,j),
     & gt(i,j),j=1,ng(i))
      go to 400
  600 if(open.eq.0)goto 700
      close(idsk)
      open=0
  700 continue
      i=0
c
c     output combined time-distance data
c
  800 i=i+1
      if(i.gt.10)go to 1000
      if(ngc(i).eq.0)go to 800
      if(open.ne.0)goto 900
      open(idsk,file=filcom,form="formatted",access="sequential",
     & mode="out")
      open=1
      write(jtty,300)filcom
      iffil=1
  900 write(idsk,100)i,ngc(i),sxc(i),sdc(i),offsc(i),(gxc(i,j),
     & gtc(i,j),j=1,ngc(i))
      go to 800
 1000 if(open.eq.1)goto 1100
      close(idsk)
      open=0
 1100 continue
      i=0
c
c     output layer data
c
 1200 i=i+1
      if(i.gt.10)go to 1400
      if(nc(i).eq.0)go to 1200
      if(open.ne.0) goto 1300
      open(idsk,file=fillay,form="formatted",access="sequential",
     & mode="out")
      open=1
      write(jtty,300)fillay
      iffil=1
 1300 write(idsk,200)i,nc(i),(cx(i,j),cy(i,j),cv(i,j),j=1,nc(i))
      go to 1200
 1400 if(open.eq.0)goto 1500
      close(idsk)
      open=0
 1500 if(iffil.eq.0)write(jtty,310)
      return
      end


      subroutine perp
c
c      corrects data point distance coordinates for shotpoint perpendicular
c      offset distance using pytagorean theorem and changes the offset
c      to a negative value -1.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 ext
   77 write(jtty,1000)
 1000 format(" type extension--either org or com: ",$)
c
c      direct execution to either org or com data
c
      read(itty,1001)ext
 1001 format(a3)
      if(ext.eq."org")go to 75
      if(ext.eq."com")go to 76
      write(jtty,1002)
 1002 format(" extension mispelled. try again")
      go to 77
c
c      make offset correction for extension .org and set offset distance to
c      -1. if distance already -1, make no correction.
c
   75 do 100 i=1,50
      if(offset(i).lt.0.)go to 100
      if(ng(i).eq.0)go to 100
      do 33 j=2,ng(i)
      if(gx(i,j).eq.0.)go to 34
      dx=gx(i,j)
   34 gx(i,j)=gx(i,j-1)+dx
   33 continue
      if(offset(i).ne.0.)go to 31
      offset(i)=-1.
      go to 100
   31 oset=offset(i)*offset(i)
      do 35 j=1,ng(i)
      dx=gx(i,j)-sx(i)
      if(dx.eq.0.)go to 36
```

```
      f=sign(1.,dx)
      go to 37
   36 if(j.le.4)go to 38
      f=-1.
      go to 37
   38 f=1.
   37 gx(i,j)=sx(i)+f*sqrt(oset+dx*dx)
      offset(i)=-1.
   35 continue
  100 continue
      go to 300
c
c      make offset correction for extension .com and set offset distance
c      to -1. if distance already -1, make no correction.
c
   76 do 200 i=1,10
      if(offsc(i).lt.0.)go to 200
      if(ngc(i).eq.0)go to 200
      do 43 j=2,ngc(i)
      if(gxc(i,j).eq.0.)go to 44
      dx=gxc(i,j)
   44 gxc(i,j)=gxc(i,j-1)+dx
   43 continue
      if(offsc(i).ne.0.)go to 41
      offsc(i)=-1.
      go to 200
   41 oset=offsc(i)*offsc(i)
      do 45 j=1,ngc(i)
      dx=gxc(i,j)-sxc(i)
      if(dx.eq.0.)go to 46
      f=sign(1.,dx)
      go to 47
   46 if(j.le.4)go to 48
      f=-1.
      go to 47
   48 f=1.
   47 gxc(i,j)=sxc(i)+f*sqrt(oset+dx*dx)
      offsc(i)=-1.
   45 continue
  200 continue
c
c      output corrected data to disk
c
  300 call output
      return
      end


      subroutine pick
c
c     sub pick permits user to select a subset of a t-x data set in an arrival
c     time data file (org or com) and create a new t-x data set of it
c     (org or com). a constant may be added to the arrival times of the
c     new set.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 ext1,ext2
      dimension x1(200),t1(200)
      external select(descriptors)
      integer shot1,shot2
c
c     initialize new data set to 0.
c
   10 do 201 i=1,200
      x1(i)=0.
      t1(i)=0.
  201 continue
      add=0.
c
c     input data
c
   12 write(jtty,1000)
 1000 format(" pick entry 1; shot1*,ext1,xmin,xmax,shot2,ext2,add",/," *",$)
      read(itty,1001)shot1,ext1,xmin,xmax,shot2,ext2,add
 1001 format(v)
      if(shot1.lt.1)go to 990
c
c     check for input errors.
c
      if((ext2.eq."org".or.ext2.eq."com").and.(ext1.eq."org".or.ext1.eq."com"))
     &go to 11
      write(jtty,1002)
 1002 format(" extension mispelled. try again")
      go to 12
   11 if(xmin.ge.xmax)go to 980
      if(ext1.eq."org")go to 13
      if(shot1.gt.10.or.shot1.lt.1)go to 980
      ds=sdc(shot1)
      xs=sxc(shot1)
      go to 14
   13 if(shot1.gt.50.or.shot1.lt.1)go to 980
      ds=sd(shot1)
      xs=sx(shot1)
   14 if(ext2.eq."org")go to 15
      if(shot2.gt.10.or.shot2.lt.1)go to 980
      go to 16
   15 if(shot2.gt.50.or.shot2.lt.1)go to 980
   16 continue
c
c     select the desired subset from the original data set
c
      call select(xmin,xmax,ext1,shot1,iel,t1,x1,jwatch,iflag)
      if(iel.gt.0)go to 25
```

```
      write(jtty,1005)
 1005 format(" pick condition 1")
      return
   25 if(lflag.eq.0)go to 17
      write(jtty,1004)
 1004 format(" pick condition 2")
      return
c
c     insert "picked" arrival-time data under the new shot indx
c
   17 if(ext2.eq."org")go to 18
      do 19 i=1,iel
      gxc(shot2,i)=xl(i)
      gtc(shot2,i)=tl(i)+add
   19 continue
      sxc(shot2)=xs
      sdc(shot2)=ds
      offsc(shot2)=-1.
      ngc(shot2)=iel
      go to 26
   18 do 20 i=1,iel
      gx(shot2,i)=xl(i)
      gt(shot2,i)=tl(i)+add
   20 continue
      sx(shot2)=xs
      sd(shot2)=ds
      offset(shot2)=-1.
      ng(shot2)=iel
   26 write(jtty,1007)ext2,shot2,iel
 1007 format(" new data set: extension=",a3,2x,
     &"index=",i2,3x,"njugs=",i3)
      go to 10
  980 write(jtty,1006)
 1006 format(" error in input data. try again")
      go to 12
c
c     outut file to disk
c
  990 call output
      return
      end


      subroutine plotx
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/names/filnam,filorg,filcom,fillay,idsk
      dimension dxp(2),dyp(2),xp(4),yp(4),lab(5),labl(6),xy(2),
     & dp1(2),dp2(2),dp3(2),jdim(1)
      character*13 fn
      character*6 filnam
      character*10 filorg,filcom,fillay
      character*3 ext,pl
      character*1 symbl,sym
      equivalence (dp1(1),xmin),(dp1(2),xmax),(dp2(1),tmin),
     & (dp2(2),tmax)
      external char(descriptors)
      dimension symbl(10)
      data symbl/"X","2","3","4","5","6","7","8","9","0"/
      data jdim(1)/0/
      plotn=-.5
      xpl=-2.9
c     read and check input parms
c
    1 write(jtty,6)
    6 format(" plotx entry 1; pl,tmin,tmax,xmin,xmax,timehi,xlong,space"
     & ,/," *",$)
      timehi=0.
      xlong=0.
      space=0.
      read(itty,11)pl,tmin,tmax,xmin,xmax,timehi,xlong,space
   11 format(v)
      if(pl.eq."cal" .or. pl.eq."var".or. pl.eq."tek")go to 21
      write(jtty,16)
   16 format(" incorrect plotter name. try again")
      go to 1
   21 if(tmin.lt.tmax .and. xmin.lt.xmax)go to 31
      write(jtty,26)
   26 format(" tmin gt tmax or xmin gt xmax. try again")
      go to 1
   31 if(timehi.le.0. .or. xlong.le.0. .or. space.le.0.)go to 46
      tscal=(tmax-tmin)/timehi
      dscal=(23.*space)/xlong
      tlab=tscal*.5
      dlab=dscal*.75
      write(jtty,36)tscal,dscal,tlab,dlab
   36 format(" Suggested input for entry 2 is:",/,
     & "     tscal     dscal     tlab      dlab",/,4f10.1)
      write(jtty,41)
   41 format(" plotx entry 2; tscal,dscal,tlab,dlab",/," *",$)
   46 read(itty,11)tscal,dscal,tlab,dlab
      if(tscal.gt.0. .and. dscal.gt.0.)go to 56
      write(jtty,51)
   51 format(" one or both scales not specified. Try again")
      go to 1
   56 if(tlab.gt.0. .and. dlab.gt.0.)go to 66
      write(jtty,61)
   61 format(" tlab and/or dlab not specified. Try again")
      go to 1
c
c     setup for plotting
c
   66 iplotr=1
      if(pl.eq."cal")iplotr=0
      if(pl.eq."var")iplotr=6
```

```
      if(pl.eq."tek")iplotr=1
      call pltset(iplotr,xbr,ybr,jdim)
      dxp(1)=xmin
      dxp(2)=xmax
      dyp(1)=tmin
      dyp(2)=tmax
      dp3(1)=tmin
      dp3(2)=tmax+tscal*.5
      xp(1)=(xmax-xmin)/dscal
      yp(1)=(tmax-tmin)/tscal
      xp(2)=0.
      yp(2)=0.
      xp(3)=7.
      yp(3)=1.
      xp(4)=xp(1)+xp(3)+1.
      yp(4)=yp(1)+yp(3)+1.
      if(xp(4).lt.xbr.and.yp(4).lt.ybr) go to 76
      write(jtty,71)
   71 format(" input scale too large for plotter plot rescaled")
      xp(3)=1.
      xp(4)=aminl(xbr,13.)
      yp(4)=aminl(ybr,10.)
      xp(1)=xp(4)-xp(3)
      yp(1)=yp(4)-yp(3)
   76 call scale(dxp,dyp,xp,yp,4,icode)
      if(icode.lt.0) go to 176
      call neatl
c
ccc   label axis
c
      encode(fn,81) filnam
   81 format("file = ",a6)
      call char(dxp(1),dyp(1),fn,53,2,.12,0.,0.,-.8)
      call char((dxp(1)+dxp(2))*.5,dyp(1),"distance",8,2,.12,
     & 0.,-.45,-.8)
      call char(dxp(1),(dyp(1)+dyp(2))*.5,"time",4,2,.12,
     & 1.5706,-.15,.8)
c
c     plot grid lines and label
c
      call laxis(dp1,dp2,dlab,4.712389)
      call laxisy(dp2,dp1,tlab,0.)
c      read and check input parameters
c
      write(jtty,86)
   86 format(" plotx entry 3; k*,ext,<sym>")
   91 sym="i"
      write(jtty,96)
   96 format(" *",$)
      read(itty,101)k,ext,sym
  101 format(v)
      if(k.le.0)go to 186
      if(ext.eq."org" .or. ext.eq."com")go to 111
      write(jtty,106)
  106 format(" extension mispelled. Try again.")
      go to 91
  111 if(k.gt.50)go to 116
      if(ext.eq."com" .and. k.gt.10)go to 116
      go to 126
  116 write(jtty,121)
  121 format(" shot index out of range. Try again.")
      go to 91
  126 kk=mod(k,10)
      if(kk.eq.0)kk=10
      if(sym.eq."i")sym=symbl(kk)
c
c     plot shot distance line and label in left margin
c
  131 plotn=plotn+.5
      if(plotn.lt.yp(1)) go to 136
      plotn=0.
      xpl=xpl-2.
  136 encode(lab,141) k,ext
  141 format("shot ",i2,"  ext = ",a3)
      if(ext.eq."com") go to 161
c
c     org data
c
      encode(labl,146) sym,sx(k)
  146 format("sym = ",a1,"  dist = ",f7.1)
      call char(xmin,tmax,lab,18,2,.08,0.,xpl,-plotn)
      call char(xmin,tmax,labl,23,2,.08,0.,xpl,-(plotn+.15))
      if(sx(k).le.xmin.or.sx(k).ge.xmax) go to 151
      xy(1)=sx(k)
      xy(2)=sx(k)
      call line(xy,dp3,2,0,0)
c
c     plot org t-x values
c
  151 np=ng(k)
      do 156 i=1,np
      call char(gx(k,i),gt(k,i),sym,1,2,.08,0.,0.,0.)
  156 continue
      go to 91
c
c     com data
c
  161 encode(labl,146) sym,sxc(k)
      call char(xmin,tmax,lab,18,2,.08,0.,xpl,-plotn)
      call char(xmin,tmax,labl,23,2,.08,0.,xpl,-(plotn+.15))
      if(sxc(k).le.xmin.or.sxc(k).ge.xmax) go to 166
      xy(1)=sxc(k)
      xy(2)=sxc(k)
      call line(xy,dp3,2,0,0)
c
c     plot com t-x values
c
  166 np=ngc(k)
```

```
      do 171 i=1,np
      call char(gxc(k,i),gtc(k,i),sym,1,2,.08,0.,0.,0.)
  171 continue
      go to 91
  176 write(jtty,181)
  181 format(" plot cannot be scaled")
      go to 91
  186 call endpt(jdim)
      return
c
c     axis labelling routine
c
      end


      subroutine laxis(dxp,dyp,dlab,ang)
      external char(descriptors)
      external line(descriptors)
      dimension dxp(2),dyp(2),lab(2),xy(2)
      xoff=.15
      inc=(dxp(2)-dxp(1))/dlab
      inc=inc*5
      xd=dxp(1)
      yd=dlab*.2
      call char(xd,dyp(1),"-",1,2,.08,ang,0.,0.)
      encode(lab,1) xd
      call char(xd,dyp(1),lab,5,2,.1,ang,xoff,0.)
      do 6 i=1,inc
      xd=xd+yd
      call char(xd,dyp(1),"-",1,2,.08,ang,0.,0.)
      if(mod(i,5).ne.0) go to 6
      if(abs(xd-dxp(2)).le..00001) go to 6
      encode(lab,1) xd
    1 format(f6.0)
      call char(xd,dyp(1),lab,5,2,.1,ang,xoff,0.)
      xy(1)=xd
      xy(2)=xd
      call line(xy,dyp,2,0,0)
    6 continue
      call char(dxp(2),dyp(1),"-",1,2,.08,ang,0.,0.)
      encode(lab,1) dxp(2)
      call char(dxp(2),dyp(1),lab,5,2,.1,ang,xoff,0.)
c
c     put ticks at top
c
      xd=dxp(1)
      do 11 i=1,inc
      xd=xd+yd
      call char(xd,dyp(2),"-",1,2,.08,ang,0.,0.)
   11 continue
      return
c
c     label left side
c
      entry laxisy(dxp,dyp,dlab,ang)
      xoff=-.6
      inc=(dxp(2)-dxp(1))/dlab
      inc=inc*5
      xd=dxp(1)
      yd=dlab*.2
      call char(dyp(1),xd,"-",1,2,.08,ang,0.,0.)
      encode(lab,1) xd
      call char(dyp(1),xd,lab,5,2,.1,ang,xoff,0.)
      do 16 i=1,inc
      xd=xd+yd
      call char(dyp(1),xd,"-",1,2,.08,ang,0.,0.)
      if(mod(i,5).ne.0) go to 16
      if(abs(xd-dxp(2)).le..00001) go to 16
      encode(lab,1) xd
      call char(dyp(1),xd,lab,5,2,.1,ang,xoff,0.)
      xy(1)=xd
      xy(2)=xd
      call line(dyp,xy,2,0,0)
   16 continue
      call char(dyp(1),dxp(2),"-",1,2,.08,ang,0.,0.)
      encode(lab,1) dxp(2)
      call char(dyp(1),dxp(2),lab,5,2,.1,ang,xoff,0.)
c
c     label right side
c
      xoff=.1
      xd=dxp(1)
      call char(dyp(2),xd,"-",1,2,.08,ang,0.,0.)
      encode(lab,1) xd
      call char(dyp(2),xd,lab,5,2,.1,ang,xoff,0.)
      do 21 i=1,inc
      xd=xd+yd
      call char(dyp(2),xd,"-",1,2,.08,ang,0.,0.)
      if(mod(i,5).ne.0) go to 21
      if(abs(xd-dxp(2)).le..00001) go to 21
      encode(lab,1) xd
      call char(dyp(2),xd,lab,5,2,.1,ang,xoff,0.)
   21 continue
      call char(dyp(2),dxp(2),"-",1,2,.08,ang,0.,0.)
      encode(lab,1) dxp(2)
      call char(dyp(2),dxp(2),lab,5,2,.1,ang,xoff,0.)
      return
      end


      subroutine range(f,a,b,i)
c
c     given two points with distance coordinates a and b,  and a third
c     point with distance coordinate f,  then the variable i  assumes
c     the value zero if f lies within the range of a  and b,  and one
c     if it lies outside the range.
c
      i=0
```

```
      xl=aminl(a,b)
      x2=amaxl(a,b)
      if(f.ge.xl.and.f.le.x2)go to 10
      i=1
   10 return
      end


      subroutine rathru(x,id,a,tau,nrays,bend)
c
c     given a starting distance coordinate x, on horizon id, a submerging
c     angle with respect to the vertical a, and a travel time tau,
c     subroutine rathru constructs a series of ray segments starting
c     from x which propagate through the model until the last ray segment
c     approaches infinity (plus or minus ten million).  nrays counts the
c     the number of segments.  the parameters for the ray segments are
c     stored in the variables of common block 1, where xp,yp are
c     the coordinates of a ray segment, ap is the submergent angle,
c     vp its velocity of propagation and taup its travel-time.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/datal/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      dimension xp(51),yp(51),ap(51),vp(51),taup(51)
      common/block1/xp,yp,ap,vp,taup
      character*3 bend
      external csect(descriptors)
c
c     determine compartment parameters for point x on horizon id.
c
      call comp(x,id,yd,x1,y1,x2,y2,vd,jd,iflag)
      if(iflag.ne.0)go to 995
c
c     set the first point for the ray propagating through the model.
c
      xp(1)=x
      yp(1)=yd
      vp(1)=vd
      ap(1)=a
      taup(1)=tau
c
c     initialize
c
      ad=a
      xd=x
      nrays=1
c
c     exit if ray is at minus infinity or over 50 ray segments
c     have been determined.
c
   10 if(yd.le.(-1000000.).or.nrays.ge.50)go to 995
c
c     is ray vertical?
c
      if(ad.eq.0.)go to 1
      if(abs(ad).lt..01)ad=.01*(ad/abs(ad))
      if(ad.gt.0.)go to 6
c
c     next compartment boundary to be intersected lies oto the right.
c
      if(jd.eq.nc(id))go to 7
c
c     determine distance coordinate of the next compartment boundary to
c     be intersected and the velocity on the other side.
c
      vnext=cv(id,jd+1)
      xnext=cx(id,jd+1)
      go to 20
    7 vnext=cv(id,jd)
      xnext=1000000.
      go to 20
c
c     nest compartment boundary to be intersected lies to the left.
c
    6 if(jd.eq.1)go to 8
c
c     determine distance coordinate of nest compartment boundary to be
c     intersected and the velocity on the other side.
c
      vnext=cv(id,jd-1)
      xnext=cx(id,jd)
      go to 20
    8 vnext=cv(id,1)
      xnext=-1000000.
   20 continue
c
c     find intersection with next adjacent compartment boundary
c     and emerging ray parameters.
c
      call csect(xd,yd,ad,vd,xnext,ynext,anext,vnext,tauc,
     & jrefl,iflag,jinc,bend)
      go to 2
c
c     ray is vertical.  prepare to exit.
c
    1 jinc=0
      vnext=vd
      xnext=xd
      ynext=-1010000.
      anext=ad
      tauc=(yd-ynext)/vd
c
c     change variables and insert calculated values into ray
c     segment output arrays.
c
    2 jd=jd+jinc
```

```
      nrays=nrays+1
      tau=tau+tauc
      xd=xnext
      yd=ynext
      ad=anext
      vd=vnext
      xp(nrays)=xd
      yp(nrays)=yd
      ap(nrays)=ad
      vp(nrays)=vd
      taup(nrays)=tau
c
c     has the end of the model been encountered?
c
      if(jd.eq.0.or.jd.eq.(nc(id)+1))go to 995
      go to 10
  995 return
      end


      subroutine rayd(xp,ip,id,beta,xd,yd,taut,ad,vd,jd,iflag,irefl,iq,bend)
c
c      given a starting distance coordinate xp on horizon ip, an initial
c      submergent angle beta, and travel time taut, a submergent ray is
c      traced down to horizon id.   (xd,yd) are the coordinates at the
c      point of emergence of the ray on horizon ip, ad the submergent
c      angle, vd the velocity, jd the compartment index and taut the travel
c      time at that point.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 bend
      external csect(descriptors)
c
c     initialize.    nntry counts the total number of compartment
c     boundaries encountered.
c
      xpq=xp
      ipq=ip
      nntry=0
       iflag=0
c
c     check horizon indecies.
c
      if(ip.le.0.or.id.le.0.or.id.lt.ip)go to 42
      if(ip.le.10.or.id.le.10)go to 33
   42 write(jtty,1001)ip,id
 1001 format(" rayd condition 1.  ip and id=",2f7.1)
      go to 300
   33 call comp(xp,ip,yp,x1,y1,x2,y2,vp,jp,iflag)
      if(iflag.eq.0)go to 32
   35 write(jtty,1002)xpq
 1002 format(" rayd condition 2 at xp=",f5.1)
      go to 300
c
c     calculate submergence angle ap, for the initial horizon ip.
c
   32 slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      ap=beta-theta
  200 if(ip.ge.id)go to 31
      if(abs(ap).le.1.5707)go to 11
      api=ap*180./3.14159
      write(jtty,1010)xpq,api
 1010 format(" rayd condition 3.  xp=",f7.1,"   ap=",f7.1)
      go to 300
c
c     all done except for a little book-keeping.
c
   31 xd=xp
      yd=yp
      ad=ap
      vd=vp
      jd=jp
      xp=xpq
      ip=ipq
      return
c
c      initialize parameters for the next horizon underlying the ipth.
c      ntry counts the number of compartment boundaries which will
c      be encountered.
c
   11 xtry=xp
      iq=ip+1
      ntry=0
   14 ntry=ntry+1
      if(ntry.le.50)go to 34
      write(jtty,1030)xpq
 1030 format(" rayd condition 4 at xp=",f7.1)
      go to 300
   34 call comp(xtry,iq,ytry,x1,y1,x2,y2,vq,jq,iflag)
      if(iflag.ne.0)go to 35
c
c      calculate the intersection point of the ray on the horizon
c      specified by points (x1,y1),(x2,y2)
c
      call hsect(xp,yp,ap,vp,x1,y1,x2,y2,xq,yq,tauh,iflag)
      if(iflag.eq.0)go to 36
      write(jtty,1031)xpq
 1031 format(" rayd condition 5 at xp=",f7.1)
      go to 300
   36 if(xq.lt.x1)go to 12
      if(xq.lt.x2)go to 13
c
c     intersection outside of compartment boundaries.
```

```
c
   12 if(ap.lt.0.)go to 21
c
c     ray slants downward to left.
c
      if((jq-1).ge.1)go to 39
   41 write(jtty,1062)xpq
 1062 format(" rayd condition 8 at xp=",f7.1)
      go to 300
   39 xtry=(x1+cx(iq,jq-1))/2.
      go to 14
c
c     ray slants downward to right.
c
   21 if((jq+2).gt.nc(iq))go to 41
      xtry=(x2+cx(iq,jq+2))/2.
      go to 14
c
c     the intersection is within compartment boundaries specified. now
c     determine whether any compartment boundaries were intersected on
c     the way up.
c
   13 call comp(xq,ip,ydum,x1,y1,x2,y2,vpp,jpp,iflag)
      if(iflag.ne.0)go to 35
      if(jpp.eq.jp)go to 100
c
c     compartment boundaries have been crossed.
c
      if(ap.gt.0.)go to 15
c
c     move right.
c
      xpp=cx(ip,jp+1)
      vpp=cv(ip,jp+1)
      go to 16
c
c     move left.
c
   15 xpp=cx(ip,jp)
      vpp=cv(ip,jp-1)
c
c     calculate point of intersection (xpp,ypp) with compartment boundary.
c
   16 call csect(xp,yp,ap,vp,xpp,ypp,app,vpp,tauc,jrefl,iflag,jinc,bend)
      if(iflag.eq.0)go to 37
      write(jtty,1032)
 1032 format(" rayd condition 6")
      go to 300
c
c     now that compartment intersection is found, a new set of coordinates
c     must be established to determine intersection with next deeper horizon.
c
   37 xp=xpp
      yp=ypp
      ap=app
      vp=vpp
      taut=taut+tauc
c
c     and a new xtry to find the most likely underlying compartment.
c
      xtry=xp
c
c     if reflection occurred, compartment index stays the same.
c
      if(jrefl.eq.1)go to 14
      if(ap.lt.0.)go to 17
      jp=jp-1
      go to 14
   17 jp=jp+1
      go to 14
c
c     the ray has successfully impinged on the iqth. horizon without
c     encountering any more overlying compartments. calculate travel-time
c     to the iqth. horizon and the angle of refraction.
c
  100 taut=taut+tauh
      call comp(xq,iq,yq,x1,y1,x2,y2,vq,jq,iflag)
      if(iflag.ne.0)go to 35
      call angle(ap,vp,x1,y1,x2,y2,vq,aq,irefl,iflag)
      if(iflag.eq.0)go to 43
      write(jtty,1033)xpq
 1033 format(" rayd condition 9 at xp=",f7.1)
      go to 300
c
c     if irefl=1, the critical angle has been exceeded.
c
   43 if(irefl.eq.1)go to 995
c
c     make changes necessary to move into next lower layer.
c
   38 jp=jq
      ip=iq
      xp=xq
      yp=yq
      vp=vq
      ap=aq
      nntry=nntry+ntry
      if(nntry.ne.50)go to 200
      write(jtty,1061)xpq
 1061 format(" rayd condition 7 at xp=",f7.1)
      go to 200
  300 iflag=1
  995 xp=xpq
      ip=ipq
      return
      end
```

```
      subroutine raydl(xp,ip,id,beta,xd,yd,taut,iflag,bend)
c
c     given a starting distance coordinate xp on horizon ip,  an initial
c     submergent angle beta, and travel time taut, a submergent ray is
c     traced through one layer to horizon id=ip+1.    (xd,yd) are the coordinates
c     of the ray emerging on horizon id and taut the travel time at that point.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 bend
      external csect(descriptors)
c
c     initialize.
c
      xpq=xp
      ipq=ip
       iflag=0
c
c     check horizon indecies.
c
      if(ip.le.0.or.id.le.0.or.id.lt.ip)go to 42
      if(ip.le.10.or.id.le.10)go to 33
   42 write(jtty,1001)ip,id
 1001 format(" raydl condition 1. ip and id=",2f7.1)
      go to 300
   33 call comp(xp,ip,yp,x1,y1,x2,y2,vp,jp,iflag)
      if(iflag.eq.0)go to 32
   35 write(jtty,1002)xpq
 1002 format(" raydl condition 2 at xp=",f7.1)
      go to 300
c
c     calculate submergence angle ap for the initial horizon ip.
c
   32 slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      ap=beta-theta
      if(abs(ap).le.1.5707)go to 31
      api=ap*180./3.14159
      write(jtty,1010)xpq,api
 1010 format(" raydl condition 3. xp=",f7.1,"   ap=",f7.1)
      go to 300
   31 if(ip.lt.id)go to 11
c
c     all done except for a little book-keeping.
c
      xd=xp
      yd=yp
      xp=xpq
      ip=ipq
      return
c
c     initialize parameters for the next horizon underlying the ipth.
c     ntry counts the number of compartment boundaries which will
c     be encountered.
c
   11 xtry=xp
      iq=ip+1
      ntry=0
   14 ntry=ntry+1
      if(ntry.le.50)go to 34
      write(jtty,1030)xpq
 1030 format(" raydl condition 4 at xp=",f7.1)
      go to 300
   34 call comp(xtry,iq,ytry,x1,y1,x2,y2,vq,jq,iflag)
      if(iflag.ne.0)go to 35
c
c     calculate the intersection point of the ray on the horizon
c     specified by points(x1,y1),(x2,y2).
c
      call hsect(xp,yp,ap,vp,x1,y1,x2,y2,xq,yq,tauh,iflag)
      if(iflag.eq.0)go to 36
      write(jtty,1031)xpq
 1031 format(" raydl condition 5 at xpq=",f7.1)
      go to 300
   36 if(xq.lt.x1)go to 12
      if(xq.lt.x2)go to 13
   12 if(ap.lt.0.)go to 21
c
c     ray slants downward to left.
c
      if((jq-1).ge.1)go to 39
   41 write(jtty,1062)xpq
 1062 format(" raydl condition 8 at xp=",f7.1)
      go to 300
   39 xtry=(x1+cx(iq,jq-1))/2.
      go to 14
c
c     ray slants downward to right.
c
   21 if((jq+2).gt.nc(iq))go to 41
      xtry=(x2+cx(iq,jq+2))/2.
      go to 14
c
c     the intersection is within compartment boundaries specefied.  now
c     determine whether any compartment boundaries were intersected on
c     the way down.
c
   13 call comp(xq,ip,ydum,x1,y1,x2,y2,vpp,jpp,iflag)
      if(iflag.ne.0)go to 35
      if(jpp.eq.jp)go to 100
c
c     compartment boundaries have been crossed.
c
      if(ap.gt.0.)go to 15
c
```

```
c     move right.
c
      xpp=cx(ip,jp+1)
      vpp=cv(ip,jp+1)
      go to 16
c
c     move left.
c
   15 xpp=cx(ip,jp)
      vpp=cv(ip,jp-1)
c
c     calculate point of intersection (xpp,ypp) with compartment boundary.
c
   16 call csect(xp,yp,ap,vp,xpp,ypp,app,vpp,tauc,jrefl,iflag,jinc,bend)
      if(iflag.eq.0)go to 37
      write(jtty,1032)
 1032 format(" raydl condition 6")
      go to 300
c
c     now that compartment intersection is found, a new set of coordinates
c     must be established to determine intersection with next deeper horizon.
c
   37 xp=xpp
      yp=ypp
      ap=app
      vp=vpp
      taut=taut+tauc
c
c     and a new xtry to find the most likely underlying compartment.
c
      xtry=xp
c
c     if reflection occurred, compartmint index stays the same.
c
      if(jrefl.eq.1)go to 14
      if(ap.lt.0.)go to 17
      jp=jp-1
      go to 14
   17 jp=jp+1
      go to 14
c
c     the ray has succsessfully inpinged on the idth  horizon without
c     encountering any more intervening compartments.  calculate travel
c     time and exit.
c
  100 taut=taut+tauh
      xd=xq
      yd=yq
      go to 990
  300 iflag=1
  990 continue
  995 xp=xpq
      ip=ipq
      return
      end


      subroutine raytrace
c
c     given a shotpoint location, subroutine raytrace calculates
c      arrival-times for critically refracted rays at pre-set
c      intervals, and diffraction arrival-times at specified points
c      from a selected horizon of a layered model.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/names/filnam,filorg,filcom,fillay,idsk
      character*6 filnam
      character*10 filorg,filcom,fillay
      dimension xdifr(30),x(50),y(50)
      character*3 go,bend
      character*1 direc(30)
      external rayd(descriptors),rayup(descriptors)
      logical test
      rad=3.14159/180.
      deg=1./rad
c
c     initialize default options
c
    5 test=.true.
      betal=0.
      time0=0.
      dbeta=0.1
      alphal=0.
      dalpha=5.
      bend="no"
c
c     read input parameters and make checks.
c
      write(jtty,2000)filnam
 2000 format(" raytrace entry 1",20x,"file=",a6,/," layd*,shotx,shotd,xstart,
     &xend,delx,<bend,tim0,betal,dbeta,alphal,dalpha>",/," *",$)

      read(itty,2001)layd,shotx,shotd,xstart,xend,delx,bend,time0,
     & betal,dbeta,alphal,dalpha
 2001 format(v)
      if(layd.le.1)return
      if(layd.lt.11 .or. delx.gt.0.)go to 72
      write(jtty,2004)
 2004 format(" layd or delx out of range. try again.")
      go to 5
   72 if(bend.eq."no" .or. bend.eq."yes")go to 71
      write(jtty,2006)
 2006 format(" bend should be either yes or no.  try again.")
      go to 5
```

```
   71 write(jtty,2002)
 2002 format(" raytrace entry 2; ndifr,xdifr(1),direc(1),xdifr(2),
     &direc(2),.....",/," *",$)
      read(itty,2001)ndifr,(xdifr(i),direc(i),i=1,ndifr)
      if(ndifr.ge.0 .and. ndifr.le.30)go to 91
      write(jtty,1053)
 1053 format(" only 30 diffraction points permitted")
      go to 71
c
c     if f=1, computed values are to right of shotpoint.  otherwise
c     they are to the left.
c
   91 f=-1.
      if(xend.gt.xstart)f=1.
c
c      fix the sign of various angles and change degrees to radians
c
      betal=betal*rad
      dbeta=dbeta*rad*(-f)
      alphal=alphal*rad*f
      dalpha=dalpha*rad*f
      delx=delx*f
c
c     the option is available to place the shot directly on the refracting
c     horizon (layd) and bypass subroutine rayd2 which traces the
c     ray to this horizon.
c
      if(shotd.lt.0.)go to 1
      beta=betal
      ndown=1
c
c     raise shot to the top of the layer containing it at the angle beta.
c     calculate new shot time and layer index.
c
      call comp(shotx,1,yzero,x1,y1,x2,y2,v1,j,iflag)
      if(iflag.eq.0)go to 101
  151 write(jtty,1051)
 1051 format(" subroutine raytrace condition 1")
      return
  101 shoty=yzero-shotd
      lay=2
    2 call comp(shotx,lay,ylay,x1,y1,x2,y2,v2,j,iflag)
      if(iflag.ne.0)go to 151
      if(shoty.gt.ylay)go to 3
      if(lay.eq.layd)go to 4
      lay=lay+1
      v1=v2
      go to 2
    4 write(jtty,1000)
 1000 format(" subroutine raytrace condition 2")
      go to 5
c
c     shot lies above horizon number lay.  place shot on ovrlying horizon
c     and calculate corrected x and y coordinates (xs,ys) and starting time (ts)
c
    3 lays=lay-1
      call comp(shotx,lays,ylay,x1,y1,x2,y2,v,j,iflag)
      if(iflag.ne.0)go to 151
    9 if(abs(beta).lt..00001)go to 76
      tanb=cos(beta)/sin(beta)
      slope=(y1-y2)/(x1-x2)
      if(tanb.ne.slope)go to 73
      write(jtty,2003)
 2003 format(" raytrace condition 3")
      return
   73 xs=(tanb*shotx-shoty-slope*x1+y1)/(tanb-slope)
      ys=slope*(xs-x1)+y1
      go to 6
   76 xs=shotx
      ys=ylay
    6 ts=time0-sqrt((shotx-xs)**2+(shoty-ys)**2)/v
c
c     trace ray down to desired horizon (from lays to layd)
c
      call rayd(xs,lays,layd,beta,xd,yd,ts,ad,vd,jd,iflag,irefl,lrefl,bend)
      if(iflag.eq.0)go to 7
      write(jtty,1001)
 1001 format(" subroutine raytrace condition 4")
      if(ndown.gt.1)go to 11
      write(jtty,1002)
 1002 format(" subroutine raytrace condition 5")
      return
c
c     if irefl=0, the ray has passed through all the horizons at less
c     than the critical angle.  hence increment and try a ray with slightly
c     greater submergence angle.  if irefl=1, the critical angle has been
c     exceeded somewhere in the model.  therefore revert to the last
c     previous ray for the critical parameters.
c
    7 if(irefl.ne.0)go to 8
c
c     change variable names to permit recall later on.
c
      xdown=xd
      ydown=yd
      tdown=ts
      adown=ad
c
c     increment and try again.
c
      beta=beta+dbeta
      ndown=ndown+1
      go to 9
   11 write(jtty,2005)
 2005 format(" subroutine raytrace condition 6")
      go to 100
c
c     normal exit for down-going segment of ray provided ndown>1.
```

```
c
    8 if(ndown.gt.1)go to 74
      write(jtty,1003)lrefl
 1003 format(" raytrace condition 7 in layer",i5)
      return
   74 if(lrefl.eq.layd)go to 12
      write(jtty,1017)lrefl
 1017 format(" raytrace condition 8 in layer",i5)
      go to 100
   12 write(jtty,1004)
 1004 format(" normal exit for raydown")
c
c     calculate the angle of refraction for the "near"-critical reflection
c     point on horizon layd,change degrees to radians and print
c     out the "near"-critical reflection parameters on this refracting
c     horizon.
c
  100 call comp(xdown,layd,ydum,x1,y1,x2,y2,v,j,iflag)
      if(iflag.ne.0)go to 151
      slope=(y2-y1)/(x2-x1)
      theta=atan(slope)
      betad=(adown+theta)*deg
      go to 13
c
c     subroutine rayd has been bypassed.  make variable change and determine
c     y coordinate (ydown) of shot point.
c
    1 xdown=shotx
      tdown=time0
      betad=90.*(-f)
      call comp(shotx,layd,ydown,x1,y1,x2,y2,v,j,iflag)
      if(iflag.ne.0)go to 151
c
c     print "near"-critical parameters on refracting horizon.
c
   13 write(jtty,1005)layd,xdown,ydown,tdown,betad
 1005 format(" the near-critical subsurface paramaters beneath the shot-point
     & are",/,
     & "   horizon       x-coord.    elevation      time     refrac. angle",/,
     & 5x,i3,4(f13.2))
      write(jtty,1006)
 1006 format(/," do you want to continue? yes or no:",$)
      read(itty,1007)go
 1007 format(a3)
      if(go.eq."yes")go to 21
      go to 5
c
c     compare xstart with xdown.  if not compatable, chang xstart.
c
   21 if((f.gt.0.).and.((xstart-xdown).lt..001))go to 22
      go to 25
   22 xstart=xdown+.000001*abs(xdown)
      if(xend.le.xstart)xend=xstart+2.*delx
      go to 23
   25 if((f.lt.0.).and.((xdown-xstart).lt..001))go to 24
      go to 23
   24 xstart=xdown-.000001*abs(xdown)
      if(xend.ge.xstart)xend=xstart+2.*delx
c
c     check that all the diffraction points lie on the correct side of xdown.
c
   23 ii=ndifr
      do 26 i=1,ndifr
      if((f.gt.0. .and. xdifr(i).gt.xdown).or.(f.lt.0. .and.
     & xdifr(i).lt.xdown))go to 26
      do 27 j=i,ndifr
   27 xdifr(j)=xdifr(j+1)
      ii=ii-1
   26 continue
      ndifr=ii
c
c     calculate the lateral travel-times from xdown to the various points
c     on the refractting horizon using a rather complex algorithm.
c
c     write a header.
c
      write(jtty,1010)shotx,shotd,layd
 1010 format(" arrival time data for shot at x=",f10.2," depth=",f7.2,/,
     & "    from off horizon",i4,/,
     & "  x-refractor coord.   x-surface coord.  arrival time")
c
c     initialize.
c
      xup=xstart
      idifr=1
c
c     arrange for possible interchange of xup and xdown.
c
  300 xmin=amin1(xdown,xup)
      xmax=amax1(xdown,xup)
c
c     do not permit either xmin or xmax to fall on a compartment boundary.
c
   32 call comp(xmin,layd,ymin,x1,y1,x2,y2,v,jmin,iflag)
      if(iflag.ne.0)go to 152
      if(xmin.ne.x1)go to 31
      xmin=xmin+.0000001*abs(xmin)
      go to 32
   31 call comp(xmax,layd,ymax,x1,y1,x2,y2,v,jmax,iflag)
      if(iflag.ne.0)go to 152
      if(xmax.ne.x1)go to 33
      xmax=xmax+.0000001*abs(xmax)
      go to 31
  152 write(jtty,1052)
 1052 format(" subroutine raytrace condition 9")
      return
c
c     put all relevant compartment boundaries and end points into an array.
```

```
c
   33 nbndry=jmax-jmin
      npts=nbndry+2
      x(1)=xmin
      y(1)=ymin
      if(npts.eq.2)go to 41
c
c     compartment boundaries must be crossed.
c
      jbndry=jmin+1
      do 42 i=2,npts
      x(i)=cx(layd,jbndry)
      y(i)=cy(layd,jbndry)
      jbndry=jbndry+1
   42 continue
   41 x(npts)=xmax
      y(npts)=ymax
c
c     eliminate the various points not used for constructing ray.
c
   48 if(npts.eq.2)go to 43
      i=1
   46 x1=x(i)
      y1=y(i)
      x2=x(i+1)
      y2=y(i+1)
      x3=x(i+2)
      y3=y(i+2)
      ytest=y1+(y3-y1)*(x2-x1)/(x3-x1)
      if(y2.ge.ytest)go to 44
      if((i+2).eq.npts)go to 43
      i=i+1
      go to 46
   44 do 47 j=(i+2),npts
      x(j-1)=x(j)
      y(j-1)=y(j)
   47 continue
      npts=npts-1
      nbndry=nbndry-1
      go to 48
c
c     a set of points have now been determined which define a ray between x(1)
c     and x(npts).   x(1) and x(npts) do not lie on compartment boundaries
c     but all the rest do.   calculate travel-time between x(1) and x(npts)
c     taking into account all compartment velocities.
c
   43 ipt=1
      tacros=0.
   55 xleft=x(ipt)
      yleft=y(ipt)
      xright=x(ipt+1)
      yright=y(ipt+1)
      costh=(xright-xleft)/sqrt((yright-yleft)**2+(xright-xleft)**2)
      call comp(xright,layd,yright,x3,y3,x4,y4,vright,jright,iflag)
      if(iflag.ne.0)go to 152
      if(xright.eq.x3)jright=jright-1
   52 call comp(xleft,layd,yleft,x1,y1,x2,y2,vleft,jleft,iflag)
      if(iflag.ne.0)go to 152
      if(jright.eq.jleft)go to 53
      xplus=cx(layd,jleft+1)
      tacros=tacros+(xplus-xleft)/(cv(layd,jleft)*costh)
      xleft=xplus
      go to 52
   53 tacros=tacros+(xright-xleft)/(cv(layd,jleft)*costh)
      ipt=ipt+1
      if(ipt.eq.npts)go to 54
      go to 55
c
c     the travel-time from xdown to xup is completed.
c
c     if test equals false, calculate diffractions.
c
   54 if(test.eq..false.)go to 200
c
c     subroutine rayup requires two points to determine the refraction
c     angle of the incident ray into the bottom layer (layd).   one of these is
c     xup.   determine the other (xs,ys).
c
      if(f.lt.0.)go to 56
      xs=x(npts-1)
      ys=y(npts-1)
      go to 57
   56 xs=x(2)
      ys=y(2)
   57 tup=0.
      ys=ys-.01*abs(xs-xup)
c
c     trace ray to surface.
c
      call rayup(xup,layd,1,xs,ys,xtop,ytop,tup,au,vu,ju,iflag,bend,0)
      if(iflag.eq.0)go to 61
      write(jtty,1008)xup
 1008 format(" subroutine raytrace condition 10 at xup=",f10.2)
      go to 75
c
c     ray successfully traced to the surface.  calculate its arrival-time.
c
   61 time=tdown+tacros+tup
c
c      print end points of emerging ray, namely the point on the refracting
c      horizon and the point on the surface.   also print the arrival-time.
c
      write(jtty,1009)xup,xtop,time
 1009 format(3x,f13.2,6x,2(f13.2))
   75 xup=xup+delx
      if(xup*f .gt. xend*f)go to 62
      go to 300
c
```

```
c     turn on diffraction switch.
c
   62 test=.false.
c
c     calculate diffraction arrival-times.
c
      if(ndifr.lt.1)go to 90
c
c     write a header.
c
      write(jtty,1018)
 1018 format(" diffraction results from above shot:")
      xup=xdifr(1)
      ff=-1.
      if(direc(1).eq."+")ff=1.
      go to 300
  200 write(jtty,1011)xdifr(idifr)
 1011 format(" diffractions from subsurface point at x=",f10.2,/,
     & "   emergence angle   x-surface coord.  arrival time")
c
c      initialize the emergence angle for diffractions.
c
      alpha=alpha1
   82 tup=0.
c
c     trace ray to surface.
c
      call rayup(xup,layd,1,alpha,0.,xtop,ytop,tup,au,vu,ju,iflag,bend,1)
c
c     if iflag=0, an error condition has occurred in tracing ray to surface.
c     ignore it, increment and continue to next ray.
c
      if(iflag.eq.0)go to 84
      alph=alpha*deg
      write(jtty,1019)alph
 1019 format(" raytrace condition 11 for alpha=",f5.1)
      go to 83
c
c     calculate arrival-time.
c
   84 time=tdown+tacros+tup
      alph=alpha*deg
c
c     print angle of emergence at diffraction point on refracting horizon,
c     emergent point on the surface, and the travel-time.
c
      write(jtty,1013)alph,xtop,time
 1013 format(f14.2,6x,2(f14.2))
   83 alpha=alpha+dalpha*ff
      if(abs(alpha*deg).gt.80)go to 81
      go to 82
   81 idifr=idifr+1
      if(idifr.gt.ndifr)go to 90
      xup=xdifr(idifr)
      ff=-1.
      if(direc(idifr).eq."+")ff=1.
      go to 300
   90 write(jtty,1016)
 1016 format(" all done. another shot!")
      go to 5
      end


      subroutine rayup(xp,ip,iu,xs,ys,xu,yu,taut,au,vu,ju,iflag,bend,istart)
c
c     traces a ray upwards in a layered and compartmented velocity
c     model between horizons ip and iu (ip>iu).
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 bend
      external csect(descriptors)
c
c     initialize.  nntry counts the total number of compartment boundaries
c     encountered.
c
      xpq=xp
      ipq=ip
      iflag=0
      nntry=0
      if(ip.lt.0.or.iu.lt.0.or.ip.lt.iu)go to 7
      if(ip.le.10.and.iu.le.10)go to 2
    7 write(jtty,1000)ip,iu
 1000 format(" rayup condition 1. ip and iu=",2i5)
      go to 300
c
c     calculate ap, the incident angle on horizon ip,    yp, the elevation
c     coordinate corresponding to xp,  vp, the velocity  and jp, the
c     compartment index beneath xp.
c
    2 call comp(xp,ip,yp,x1,y1,x2,y2,vp,jp,iflag)
      if(iflag.eq.0)go to 8
    9 write(jtty,1011)xpq
 1011 format(" rayup condition 2 at xp=",f7.1)
      go to 300
    8 if(istart.eq.0) go to 36
c
c     for option istart=1, the angle of refraction is pre-set and ap need
c     not be calculated.
c
      aq=xs
      go to 4
   36 if(xp.ne.xs)go to 34
      rslope=1.0e6
      go to 35
```

```
    34 rslope=(yp-ys)/(xp-xs)
    35 slope=(y2-y1)/(x2-x1)
       if(xp.lt.xs)go to 1
       g=1.
       if(rslope.gt.slope)go to 3
       if(rslope.eq.slope)go to 5
       write(jtty,1012)xpq
  1012 format(" raytrace condition 3 at xp=",f7.1)
     5 rslope=slope+.0001
       go to 3
     1 g=-1.
       if(rslope.lt.slope)go to 3
       if(rslope.eq.slope)go to 6
       write(jtty,1012)xpq
     6 rslope=slope-.0001
     3 theta=atan(slope)
       phi=atan(rslope)
       beta=g*(3.14158/2.-abs(theta-phi))
       ap=beta-theta
   200 continue
c
c      if ip>iu the ray tracing process is not yet completed.
c
     4 if(ip.gt.iu)go to 11
c
c      all done except for a little book-keeping.
c
       xu=xp
       yu=yp
       au=ap
       vu=vp
       ju=jp
       xp=xpq
       ip=ipq
       return
c
c      initialize parameters for intersecting the horizon above the ipth.   ntry
c      counts the compartment boundaries encountered in this layer.
c
    11 xtry=xp
       iq=ip-1
       ntry=0
c
c      make calculations for the (ip-1)th layer.
c
    14 ntry=ntry+1
       if(ntry.le.50)go to 18
       write(jtty,1030)xpq
  1030 format(" rayup condition 4 at xp=",f7.1)
       go to 300
    18 if(ntry.gt.1)go to 32
c
c      determine angle of emergence aq, in the iqth layer.
c
       call comp(xtry,iq,ytry,x1q,y1q,x2q,y2q,vq,jq,iflag)
       if(iflag.ne.0)go to 9
c
c      if istart=1 and ip=ipq(1st. time through)the emergence angle
c      is pre-fixed.
c
       if(istart.eq.0)go to 19
       if(ip.eq.ipq) go to 32
    19 call angle(ap,vp,x1,y1,x2,y2,vq,aq,iref1,iflag)
       if(iflag.eq.0)go to 27
       write(jtty,1031)xpq
  1031 format(" rayup condition 11 at xp=",f7.1)
       go to 300
    27 if(iref1.eq.0)go to 22
       write(jtty,1050)ip,xpq
  1050 format(" rayup condition 5 at ip and xp=",i5,f7.1)
       go to 300
    22 if(abs(aq).le.(3.14159/2.))go to 32
       api=ap*180./3.14159
       write(jtty,1010)xpq,api
  1010 format(" rayup condition 6 at xp and ap=",2f7.1)
       go to 300
c
c      calculate point of intrsection (xq,yq) of ray with higher horizon
c      specified with compartment boundaries (x1q,y1q),(x2q,y2q)
c
    32 call hsect(xp,yp,aq,vq,x1q,y1q,x2q,y2q,xq,yq,tauh,iflag)
       if(iflag.eq.0)go to 26
       write(jtty,1013)xpq
  1013 format(" rayup condition 10 at xp=",f7.1)
       go to 300
c
c      test whether xq falls within compartment limits.
c
    26 if(xq.lt.x1q)go to 12
       if(xq.lt.x2q)go to 100
c
c      intersection from hsect lies outside (left or right) of
c      boundaries specified by x1q and x2q.
c
    12 if(aq.gt.0.)go to 21
c
c      ray slants upwards to left.
c
       if((jq-1).ge.1)go to 25
       write(jtty,1040)xpq
  1040 format(" rayup condition 7 at xp=",f7.1)
       go to 300
c
c      get compartment boundary and velocity.
c
    25 xpp=cx(iq,jq)
       vqq=cv(iq,jq-1)
       go to 31
```

```
c
c     ray slants upward to right.
c
   21 if((jq+2).le.nc(iq))go to 23
      write(jtty,1040)xpq
      go to 300
c
c     get compartment boundary and velocity.
c
   23  xpp=cx(iq,jq+1)
       vqq=cv(iq,jq+1)
c
c     move ray across compartment boundary.
c
   31 call csect(xp,yp,aq,vq,xpp,ypp,aqq,vqq,tauc,jrefl,iflag,jinc,bend)
      if(iflag.eq.0)go to 24
      write(jtty,1060)
 1060 format(" rayup condition 8")
      go to 300
c
c     change variables around and add in compartment travel time.
c
   24 xp=xpp
      yp=ypp
      aq=aqq
      vq=vqq
      taut=taut+tauc
c
c     if reflection occurred, leave compartment index alone.
c
      if(jrefl.eq.1)go to 14
c
c     adjust compartment index.
c
      if(aq.gt.0.)go to 17
c
c     ray up to left.
c
      jq=jq-1
      go to 33
c
c     ray up to right.
c
   17 jq=jq+1
c
c     estabiish new compartment limits.
c
   33 x1q=cx(iq,jq)
      y1q=cy(iq,jq)
      x2q=cx(iq,jq+1)
      y2q=cy(iq,jq+1)
      go to 14
c
c     the ray has successfully impinged on the iqth. horizon without
c     encountering any more underlying compartment bounderies.   calculate
c     travel-time to this point.
c
  100 taut=taut+tauh
c
c     make changes necessary to move into next higher horizon.
c
      x1=x1q
      y1=y1q
      x2=x2q
      y2=y2q
      jp=jq
      ip=iq
      xp=xq
      yp=yq
      vp=vq
      ap=aq
c
c     count the total number of compartment bounderies the ray has
c     encountered so far.
c
      nntry=nntry+ntry
      if(nntry.ne.50)go to 200
      write(jtty,1061)xpq
 1061 format(" raytrace condition 9 at xp=",f7.1)
      go to 200
  300 iflag=1
      xp=xpq
      ip=ipq
      return
      end


      subroutine recip(nisect,xi,eta,vinst,v,tau,delt,dell,xminus,xplus)
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      common/block1/xp,yp,ap,vp,taup
      common/block2/xpl,ypl,apl,vpl,tpl,xpr,ypr,apr,vpr,tpr,xpll,
     &ypll,apll,vpll,tpll
      dimension dum(20),fnll(10),fnlr(10),fnrl(10),fnrr(10),tl(400),xl(400),
     &tr(400),xr(400),xstarl(30),tstarl(30),xpriml(30),tpriml(30),
     &xstarr(30),tstarr(30),xprimr(30),tprimr(30)
      dimension xp(51),yp(51),ap(51),vp(51),taup(51),xpl(51),ypl(51),
     &apl(51),vpl(51),tpl(51),xpr(51),ypr(51),apr(51),vpr(51),tpr(51),
     &xpll(200,51),ypll(200,51),apll(200,51),vpll(200,51),tpll(200,51)
      dimension ncompl(200),x22(200),dell(200),delt(200),xi(200),eta(200),
     &xplus(200),xminus(200),v(200),vinst(200),tau(200)
      dimension al(400),bl(400),cl(400),ar(400),br(400),cr(400),
     &p(400),s(400),d(2)
      common/names/filnam,filorg,filcom,fillay,idsk
      character*6 filnam
```

```
      character*10 filorg,filcom,fillay
      character*3 ext1,extr,go,curv,bend1,bend2,mycip
      integer*4 optll,optlr,optrl,optrr
      external select(descriptors),waved(descriptors),rathru(descriptors)
c
c     initialize.
c
      d(1)=0.
      d(2)=0.
      iu=1
      bend1="no"
      bend2="no"
      mycip="no"
      nvall=3
      nvalr=3
      tuhl=0.
      tuhr=0.
   21 write(jtty,1000)filnam
 1000 format(" recip entry 1",20x,"file=",a6,/," id*,curv,delx,
     &<bend1,bend2,iu,mycip,tss,tuhl,tuhr>",/," *",$)

c
c     read input parameters applicable to both shotpoints.
c
      read(itty,1001)id,curv,delx,bend1,bend2,iu,mycip,tss,tuhl,tuhr
 1001 format(v)
c
c     make checks on input parameters.
c
      if(id.le.0)return
      if(id.le.10 .and. id.ge.iu .and. delx.gt.0.
     & .and. (curv.eq."lin".or.curv.eq."spl"))go to 20
      write(jtty,1002)
 1002 format(" recip condition 1")
      go to 21
c
c     initialize
c
   20 nprinl=0
      nstarl=11
      write(jtty,1003)
 1003 format(" recip entry 2; kl*,ext1,xminl,xmaxl,delxl,dintl,
     &<nvall,nprinl,nstarl>",/," *",$)
c
c     read input parameters relevant to left shotpoint.
c
      read(itty,1001)kl,ext1,xminl,xmaxl,delxl,dintl,nvall,nprinl,nstarl
c
c     do checks on the input data.
c
      if(kl.le.0)return
      if(ext1.eq."com")go to 22
      if(ext1.eq."org")go to 23
      go to 24
   23 if(kl.gt.50)go to 24
      xsl=sx(kl)
      dsl=sd(kl)
      go to 121
   22 if(kl.gt.10)go to 24
      xsl=sxc(kl)
      dsl=sdc(kl)
  121 if(xminl.le.xmaxl .and. delxl.gt.0. .and. dintl.gt.0. .and.
     & nstarl.le.30 .and.nvall.ge.2 .and.
     & nvall.le.30)go to 25
   24 write(jtty,1004)
 1004 format(" recip condition 2")
      go to 20
   25 write(jtty,1005)
 1005 format(" recip entry 2a; optll*,+or-dxll,+or-fnll(i)",/," *",$)
c
c     read parameters for extending left shotpoint data to the left and check.
c
      read(itty,1001)optll,dxll,(fnll(i),i=1,10)
      if(optll.lt.0 .or. optll.gt.3)return
      i=optll+1
c
c     make checks on extension parameters.
c
      go to(50,27,28,29),i
c
c     i=2,   for a specific set of data points.
c
   27 n=dxll
      if(n.lt.1 .or. n.gt.5)go to 51
      do 52 i=1,n
      if(i.eq.1)go to 52
      if(fnll(i).le.fnll(i-1))go to 51
   52 continue
      go to 50
   51 write(jtty,1006)
 1006 format(" recip condition 3")
      go to 25
c
c     i=3,   for velocity criteria.
c
   28 if(dxll.lt.0.  .and.  abs(fnll(1)).gt.0.)go to 50
      go to 51
c
c     i=4,  for least squares criteria.
c
   29 if(dxll.lt.0. .and. fnll(1).gt.0.)go to 50
      go to 51
   50 continue
   54 write(jtty,1007)
 1007 format(" recip entry 2b; optlr,+or-dxlr,+or-fnlr(i)",/," *",$)
c
c     read parameters for extending left shotpoint data to the right and check.
c
```

```
      read(itty,1001)optlr,dxlr,(fnlr(i),i=1,10)
      if(optlr.lt.0 .or. optlr.gt.3)return
      i=optlr+1
c
c     make checks on extension parameters.
c
      go to(55,56,57,58),i
c
c     i=2,  for a specific set of data points.
c
   56 n=dxlr
      if(n.lt.1 .or. n.gt.5)go to 59
      do 70 i=1,n
      if(i.eq.1)go to 70
      if(fnlr(i).le.fnlr(i-1))go to 59
   70 continue
      go to 55
   59 write(jtty,1006)
      go to 54
c
c     i=3,  for velocity criteria.
c
   57 if(dxlr.gt.0.  .and.  abs(fnlr(1)).gt.0.)go to 55
      write(jtty,1006)
      go to 54
c
c     i=4,  for least squares criteria.
c
   58 if(dxlr.gt.0. .and. fnlr(1).gt.0.)go to 55
      write(jtty,1006)
      go to 54
   55 continue
c
c     initialize.
c
  220 nprinr=0
      nstarr=11
      write(jtty,2003)
 2003 format(" recip entry 3; kr*,extr,xminr,xmaxr,delxr,dintr,
     &<nvalr,nprinr,nstarr>",/," *",$)
c
c     read parameters relavent to the right shotpoint.
c
      read(itty,1001)kr,extr,xminr,xmaxr,delxr,dintr,nvalr,nprinr,nstarr
c
c     make checks on input parameters.
c
      if(kr.le.0)return
      if(extr.eq."com")go to 322
      if(extr.eq."org")go to 223
      go to 224
  223 if(kr.gt.50)go to 224
      xsr=sx(kr)
      dsr=sd(kr)
      go to 221
  322 if(kr.gt.10)go to 224
      xsr=sxc(kr)
      dsr=sdc(kr)
  221 if(xminr.le.xmaxr .and. delxr.gt.0. .and. dintr.gt.0. .and.
     & nstarr.le.30 .and. nvalr.ge.2 .and.
     & nvalr.le.30)go to 225
  224 write(jtty,2004)
 2004 format(" recip condition 4")
      go to 220
  225 write(jtty,2005)
 2005 format(" recip entry 3a; optrl*,+or-dxrl,+or-fnrl(i)",/," *",$)
c
c     read parameters for extending right shotpoint data to left and check.
c
      read(itty,1001)optrl,dxrl,(fnrl(i),i=1,10)
      if(optrl.lt.0 .or. optrl.gt.3)return
      i=optrl+1
c
c     make checks on input parameters.
c
      go to(250,227,228,229),i
c
c     i=2,  for a specific set of data points.
c
  227 n=dxrl
      if(n.lt.1 .or. n.gt.5)go to 251
      do 252 i=1,n
      if(i.eq.1)go to 252
      if(fnrl(i).le.fnrl(i-1))go to 251
  252 continue
      go to 250
  251 write(jtty,2006)
 2006 format(" recip condition 5")
      go to 225
c
c     i=3,  for velocity criteria.
c
  228 if(dxrl.lt.0. .and. abs(fnrl(1)).gt.0.)go to 250
      go to 251
c
c     i=4,  for least squares criteria.
c
  229 if(dxrl.lt.0.  .and.  fnrl(1).gt.0.)go to 250
      go to 251
  250 continue
  254 write(jtty,2007)
 2007 format(" recip entry 3b; optrr*,+or-dxrr,+or-fnrr(i)",/," *",$)
c
c     read parameters for extending right shotpoint data to the right and check.
c
      read(itty,1001)optrr,dxrr,(fnrr(i),i=1,10)
      if(optrr.lt.0 .or. optrr.gt.3)return
      i=optrr+1
```

```
c
c     make checks on extension parameters.
c
      go to(255,256,257,258),i
c
c     i=2,  for a specific set of data points.
c
  256 n=dxrr
      if(n.lt.1 .or. n.gt.5)go to 259
      do 270 i=1,n
      if(i.eq.1)go to 270
      if(fnrr(i).le.fnrr(i-1))go to 259
  270 continue
      go to 255
  259 write(jtty,2006)
      go to 254
c
c     i=3  for the velocity criteria.
c
  257 if(dxrr.gt.0. .and. abs(fnrr(i)).gt.0.)go to 255
      write(jtty,2006)
      go to 254
c
c     i=4,  for least squares criteria.
c
  258 if(dxrr.gt.0. .and. fnrr(1).gt.0.)go to 255
      write(jtty,2006)
      go to 254
  255 continue
c
c     select data points to be used for calculating depths and put into arrays
c     xl(i),tl(i)   and xr(i),tr(i).
c
  155 continue
      if(xminl.ne.xmaxl)go to 31
   34 write(jtty,1063)
 1063 format(" recip entry 2c; nl*,xl(i),tl(i)",/," *",$)
c
c     read artificial data for the left shotpoint.
c
      read(itty,1001)nl,(xl(i),i=1,nl),(tl(i),i=1,nl)
      if(nl.le.1 .or. nl.gt.200)return
      do 32 i=2,nl
      if(xl(i).gt.xl(i-1))go to 32
      write(jtty,1064)
 1064 format(" recip condition 6")
      go to 34
   32 continue
      xminl=xl(1)
      xmaxl=xl(nl)
      go to 33
   31 call select(xminl,xmaxl,extl,kl,nl,tl,xl,jwatch,iflag)
      if(iflag.eq.0 .and. jwatch.eq.0)go to 33
   74 write(jtty,1008)
 1008 format(" recip condition 7")
      return
   33 if(xminr.ne.xmaxr)go to 73
   36 write(jtty,1065)
 1065 format(" recip entry 3c; nr*,xr(i),tr(i)",/," *",$)
c
c     read artificial data for right shotpoint.
c
      read(itty,1001)nr,(xr(i),i=1,nr),(tr(i),i=1,nr)
      if(nr.le.1 .or. nr.gt.200)return
      do 35 i=2,nr
      if(xr(i).gt.xr(i-1))go to 35
      write(jtty,1064)
      go to 36
   35 continue
      xminr=xr(1)
      xmaxr=xr(nr)
      go to 304
   73 call select(xminr,xmaxr,extr,kr,nr,tr,xr,jwatch,iflag)
      if(iflag.eq.0 .and. jwatch.eq.0)go to 304
      go to 74
c
c     extend the data from both shot points as far as desired to the left and/or
c     right. note!! Data must extend beyond opposite shot point, unless
c     mycip=yes.
c
  304 call xtend(xl,tl,nl,optll,dxll,fnll,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 87
      write(jtty,1010)
 1010 format(" left sp extended to left")
   88 write(jtty,1011)
 1011 format(" recip condition 9")
      return
   87 call xtend(xl,tl,nl,optlr,dxlr,fnlr,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 89
      write(jtty,1038)
 1038 format(" left sp extended to right")
      go to 88
   89 call xtend(xr,tr,nr,optrl,dxrl,fnrl,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 90
      write(jtty,1012)
 1012 format(" rght sp extended to left")
      go to 88
   90 call xtend(xr,tr,nr,optrr,dxrr,fnrr,ioptl,iopt3,idim)
      if(ioptl.eq.0 .and. iopt3.eq.0 .and. idim.eq.0)go to 75
      write(jtty,1013)
 1013 format(" right sp extended to right")
      go to 88
c
c     move shots to top of lu horizon by calculating uphole times and
c     adding them to the arrival times.
c
   75 if(mycip.eq."no")go to 301
      do 302 i=1,nl
```

```
  302 tl(i)=tl(i)+tuhl
      do 303 i=1,nr
  303 tr(i)=tr(i)+tuhr
      go to 91
  301 icount=1
      xs=xsl
      ds=dsl
   76 tuh=0.
      idown=iu
      call comp(xs,iu,y0,x1,y1,x2,y2,v0,j,iflag)
      if(iflag.eq.0)go to 77
   78 write(jtty,1009)
 1009 format(" recip condition 8")
      return
   77 yshot=y0-ds
   80 idown=idown+1
      if(idown.lt.id)go to 79
      tuh=tuh+(y0-yshot)/v0
      go to 82
   79 call comp(xs,idown,ydown,x1,y1,x2,y2,vdown,j,iflag)
      if(iflag.eq.0)go to 81
      go to 78
   81 tuh=tuh+(y0-ydown)/v0
      if(yshot.lt.ydown)go to 83
      tuh=tuh-(yshot-ydown)/v0
      go to 82
   83 y0=ydown
      v0=vdown
      go to 80
   82 if(icount.gt.1)go to 84
      do 85 i=1,nl
   85 tl(i)=tl(i)+tuh
      icount=2
      xs=xsr
      ds=dsr
      go to 76
   84 do 86 i=1,nr
   86 tr(i)=tr(i)+tuh
c
c     calculate reciprocal times by interpolation unless mycip=yes
c     in which case user supplies reciprocal time.
c
   91 if(mycip.eq."yes")go to 94
      call interp(x1,t1,nl,xsr,tslr,iterp)
      if(iterp.eq.0)go to 92
      write(jtty,1038)
      go to 93
   92 call interp(xr,tr,nr,xsl,tsrl,iterp)
      if(iterp.eq.0)go to 94
      write(jtty,1012)
   93 write(jtty,1044)
 1044 format(" recip condition 10")
      return
c
c     determine new maximum and minimum values of array
c     in order to determine the star points.
c
   94 xminl=x1(1)
      xmaxl=x1(nl)
      xminr=xr(1)
      xmaxr=xr(nr)
      if(mycip.eq."yes")go to 124
c
c     print out interpolated reciprocal times from both shotpoints and the
c     computed reciprocal time, and give user choice to continue.
c
      tss=(tslr+tsrl)/2.
      write(jtty,1014)tslr,tsrl,tss
 1014 format(" recip times adding uphole;  left to right, right to left,
     & and average" ,/,3f11.1)
      write(jtty,1015)
 1015 format(" do you want to continue? yes or no:",$)
      read(itty,1016)go
 1016 format(a3)
      if(go.ne."yes")return
c
c     determine the values of (xstar(i)) which will be traced thru
c     subroutine waved.
c
  124 xstarl(1)=xsr
      if(mycip.eq."yes")xstarl(1)=xminl
      dumm=nstarl-2.
      xstarl(2)=xminl
      xstarl(nstarl)=xmaxl
      xint=(xmaxl-xminl)/dumm
      do 95 i=3,nstarl-1
   95 xstarl(i)=xstarl(i-1)+xint
      xstarr(1)=xsl
      if(mycip.eq."yes")xstarr(1)=xminr
      dumm=nstarr-2
      xstarr(2)=xminr
      xstarr(nstarr)=xmaxr
      xint=(xmaxr-xminr)/dumm
      do 96 i=3,nstarr-1
   96 xstarr(i)=xstarr(i-1)+xint
c
c     if iu=id, bypass waved.
c
      if(iu.ne.id)go to 281
      go to 282
c
c     move both arrival time curves from the top layer iu down to the
c     bottom layer id.
c
  281 call waved(x1,t1,nl,iu,id,delxl,dintl,nprinl,xstarl,nstarl,
     & curv,nvall,tstarl,xpriml,tpriml,idim,isplin,bendl)
      if(idim.eq.0)go to 97
      write(jtty,1017)
```

```
 1017 format(" left shotpoint")
   98 write(jtty,1018)
 1018 format(" recip condition 11")
      return
   97 if(isplin.eq.0)go to 99
      write(jtty,1017)
  101 write(jtty,1019)
      return
 1019 format(" recip condition 12")
   99 continue
      call waved(xr,tr,nr,iu,id,delxr,dintr,nprinr,xstarr,nstarr,
     & curv,nvalr,tstarr,xprimr,tprimr,idim,isplin,bend1)
      if(idim.eq.0)go to 102
      write(jtty,1020)
 1020 format(" right shotpoint")
      go to 98
  102 if(isplin.eq.0)go to 103
      write(jtty,1020)
      go to 101
  103 continue
  282 write(jtty,1024)id
 1024 format(" print t-x data on horizon",i5," ? yes or no:",$)
      read(itty,1023)go
 1023 format(a3)
      if(go.ne."yes")go to 123
      write(jtty,1025)(xl(i),tl(i),i=1,nl)
 1025 format(" left hand shot data",/,4(" distance    time "),/,
     & 4(f9.1,f9.1))
      write(jtty,1026)(xr(i),tr(i),i=1,nr)
 1026 format(" right hand shot data",/,(8f9.1))
      write(jtty,1027)
 1027 format(" continue??????")
      read(itty,1023)go
      if(go.ne."yes")return
c
c     print a header for final answers
c
  123 write(jtty,1050)
 1050 format(" The reciprocal time solutions are as follows",/,
     &14x,"point values",9x,"incremental parameters    surface coordinates",/,
     &8x,"xl",7x,"eta velocity velocity    tau    delta-t delta-x",1x,
     &"x-minus  xplus",/,65x,"right sp left sp")
c
c     equally space the arrival time data for left-hand shot-point
c     on horizon id.
c
      call eqspace(xl,tl,nl,dintl,idim)
      if(idim.eq.0)go to 122
      write(jtty,1028)
 1028 format(" recip condition 13,  left hand data")
      return
c
c     if a filter routine is to be used, insert it here.
c
c     calculate the spline coefficients if necessary.
c
  122 if(curv.ne."spl")go to 156
      call splinl(nl,0.,xl,tl,al,bl,cl,0,d,p,s)
  156 if(nl.gt.0)go to 106
      write(jtty,1029)
 1029 format(" recip condition 14,  left hand data")
      return
c
c     equally space the arrival time data for the right-hand shot point
c     on horizon id.
c
  106 call eqspace(xr,tr,nr,dintr,idim)
      if(idim.eq.0)go to 107
      write(jtty,1030)
 1030 format(" recip condition 13,  right hand data")
      return
c
c     insert filter function here if desired, and calculate the
c     spline coefficients if necessary.
c
  107 if(curv.ne."spl")go to 157
      call splinl(nr,0.,xr,tr,ar,br,cr,0,d,p,s)
  157 if(nr.gt.0)go to 108
      write(jtty,1031)
 1031 format(" recip condition 14,  right hand data")
      return
c
c     determine the limits xminl,xmaxl for the left shot point and
c     the limits xminr, xmaxr for the right shot-point between which
c     computations will be made.
c
  108 xminl=xl(1)
      xmaxl=xl(nl)
      xminr=xr(1)
      xmaxr=xr(nr)
c
c     for the left hand shot-point:
c
c     pick the starting point for the left shotpoint.
c
      x2=xmaxl
      nratru=0
c
c     compute the rathru parameters for successive points at intervals
c     delx for the left shotpoint and enter them into doubly subscripted
c     arrays.
c
c     calculate arrival time tdum, and the slope deriv, of the t-x
c     graph at x2.
c
  110 continue
      if(curv.ne."spl")go to 158
      call splindt(nl,xl,tl,al,bl,cl,x2,tdum,deriv)
```

```
      go to 159
  158 call lindt(n1,x1,t1,x2,nvall,tdum,deriv)
  159 call comp(x2,id,elev,x1,y1,x3,y3,v0,j,iflag)
      if(iflag.eq.0)go to 109
      write(jtty,1032)x2,id
 1032 format(" recip condition 15, left sp at x2=";f10.1," horizon",i5)
      go to 111
  109 f=v0*deriv
      if(abs(f).lt.1.)go to 112
      vel=1./deriv
      write(jtty,1033)x2,id,vel
 1033 format(" recip condition 16 left hand shot at x2=",f7.1,
     &" horizon",i4," velocity",f7.1)
      go to 111
c
c     determine emergence angle (adum) at x2 with respect to the vertical.
c
  112 beta=asin(f)
      slope=(y3-y1)/(x3-x1)
      theta=atan(slope)
      adum=beta-theta
      if(abs(adum).lt.1.57)go to 113
      write(jtty,1034)x2,id
 1034 format(" recip condition 17, left sp at x2=",f10.1," horizon",i5)
      go to 111
c
c     project the ray from x2 thru the entire model
c
  113 call rathru(x2,id,adum,tdum,icompl,bend2)
      nratru=nratru+1
      if(nratru.lt.201)go to 115
      write(jtty,1035)xp,id
 1035 format(" recip condition 18 at x2=",f10.1," horizon",i5)
      return
c
c     put these rathru parameters into an array to be used later for
c     intersection calculations.
c
  115 do 114 i=1,icompl
      xpll(nratru,i)=xp(i)
      ypll(nratru,i)=yp(i)
      apll(nratru,i)=ap(i)
      vpll(nratru,i)=vp(i)
      tpll(nratru,i)=taup(i)
  114 continue
      ncompl(nratru)=icompl
      x22(nratru)=x2
c
c     increment x2 and do the next ray.
c
  111 x2=x2-delx
      if(x2.lt.xminl)go to 120
      go to 110
c
c     now start with the right hand shotpoint at point xright
c     on the reduced t-x graph.
c
c
c     initialize.  nisect counts solutions and xright is the distance
c     coordinate on the idth  horizon for the first rathru ray for
c     the right shotpoint.
c
  120 nisect=0
      xright=xminr
c
c     icheck=0 means conditions are not yet ready for determining solutions
c     by interpolation.       nwin counts the number of solutions for
c     the xright being considered.
c
  500 icheck=0
      nwin=0
c
c     calculate arrival time tdum, and slope deriv, of t-x graph at xright
c     on horizon id.
c
      if(curv.ne."spl")go to 160
      call splindt(nr,xr,tr,ar,br,cr,xright,tdum,deriv)
      go to 161
  160 call lindt(nr,xr,tr,xright,nvalr,tdum,deriv)
  161 call comp(xright,id,elev,x1,y1,x3,y3,v0,j,iflag)
      if(iflag.eq.0)go to 116
      write(jtty,1045)xright,id
 1045 format(" recip condition 15, right sp at xright=",f10.1," horizon",i5)
      go to 358
  116 f=v0*deriv
      if(abs(f).lt.1.)go to 118
      vel=1./deriv
      write(jtty,1036)xright,id,vel
 1036 format(" recip condition 16, right sp at xright=",f8.1," horizon",i3,
     & " velocity",f6.1)
      go to 358
c
c     calculate emergence angle adum from the value v0 and deriv.
c
  113 beta=asin(f)
      slope=(y3-y1)/(x3-x1)
      theta=atan(slope)
      adum=beta-theta
      if(abs(adum).lt.1.57)go to 119
      write(jtty,1037)xright,id
 1037 format(" recip condition 17, right sp at xright =",f10.1," horizon",i5)
      go to 358
c
c     calculate the rathru parameters from xright thereby projecting
c     emerging ray down to infinity.
c
  119 call rathru(xright,id,adum,tdum,nraysr,bend2)
c
```

```
c     insert the rathru parameters from xright into special arrays.
c     (this step really not necessary because old array could be used.)
c
      do 62 i=1,nraysr
      xpr(i)=xp(i)
      ypr(i)=yp(i)
      apr(i)=ap(i)
      vpr(i)=vp(i)
      tpr(i)=taup(i)
   62 continue
c
c     do the process of calculating the intersections between the rathru
c     ray from xright with each of the rathru rays from the left shotpoint
c     from 1 to nratru, thereby permitting the compution of all possible
c     solutions along the ray emerging at xright.
c
      do 360 iratru=1,nratru
      nraysl=ncompl(iratru)
c
c     x2 is the point of emergence of the ray being used (the iratru-th)
c     from the left shotpoint on the idth  horizon.
c
      x2=x22(iratru)
c
c     datermine the rathru parameters for the iratru-th ray.
c
      do 361 i=1,nraysl
      xpl(i)=xpll(iratru,i)
      ypl(i)=ypll(iratru,i)
      apl(i)=apll(iratru,i)
      vpl(i)=vpll(iratru,i)
      tpl(i)=tpll(iratru,i)
  361 continue
c
c     calculate the intersection point (xi2,eta2) between the rathru
c     ray emerging at xright and the one emerging at x2.
c
      do 41 ii=1,nraysr-1
      i=ii
      xdumr1=xpr(i)
      xdumr2=xpr(i+1)
      if(abs(apr(i)).lt.1.57089.and.abs(apr(i)).gt.1.57069)
     &   apr(i)=apr(i)+.001
      tanar=sin(apr(i))/cos(apr(i))
      do 41 jj=1,nraysl-1
      j=jj
      xduml1=xpl(j)
      xduml2=xpl(j+1)
      if(abs(apl(j)).lt.1.57089.and.abs(apl(j)).gt.1.57069)
     &  apl(j)=apl(j)+.001
      if(apl(j).eq.apr(i))go to 41
      tanal=sin(apl(j))/cos(apl(j))
      eta2=(xpl(j)-xpr(i)-ypl(j)*tanal+ypr(i)*tanar)/(tanar-tanal)
      xi2=xpl(j)-(ypl(j)-eta2)*tanal
      call range(xi2,xdumr1,xdumr2,ll)
      if(ll.eq.1)go to 41
      call range(xi2,xduml1,xduml2,ll)
      if(ll.eq.1)go to 41
      if(eta2.le.ypl(j).and.eta2.le.ypr(i))go to 42
   41 continue
c
c     no intersection could be found.  move over one ray in the do loop
c     and try again.
c
      go to 360
c
c     calculate the travel time t2r, from the right shotpoint to (xi2,eta2),
c     from the left shotpoint t2, to (xi2,eta2), and the shot to shot
c     travel time tlr2, through (xi2,eta2).
c
   42 tupl=sqrt((xi2-xpl(j))*(xi2-xpl(j))+(eta2-ypl(j))*
     &    (eta2-ypl(j)))
      tupl=tupl/vpl(j)+tpl(j)-tpl(1)
      tupr=sqrt((xi2-xpr(i))*(xi2-xpr(i))
     &   +(eta2-ypr(i))*(eta2-ypr(i)))
      tupr=tupr/vpr(i)+tpr(i)-tpr(1)
      t2=tpl(1)-tupl
      t2r=tpr(1)-tupr
      tlr2=t2+t2r
      tdiff=tlr2-tss
c
c     if tlr2=tss, a solution has been found without interpolation.
c
      if(tlr2.eq.tss)go to 10
c
c     for a solution to occur, (tlr2-tss) must change sign from negative
c     to positive for two successive iratru rays.  if tlr2>tss, the
c     difference is positive and if the previous difference was negetive,
c     a solution is at hand.
c
      if(tlr2.gt.tss)go to 11
c
c     the difference is negative.  set icheck=1.
c
      icheck=1
c
c      change variables at statement labeled 2 and try next iratru ray.
c
      go to 2
c
c     test whether previous value of (tlr2-tss)<0.  if not change variables
c     at statement labeled 2 and try the next iratru ray.
c
   11 if(icheck.eq.0)go to 2
c
c     good!  a change in sign has been encountered.  find the solutions,
c     (xi3,eta3)  x3  and t3 by interpolation.
c
```

```
      f=(tlr1-tss)/(tlr1-tlr2)
      xi3=xi1+(xi2-xi1)*f
      eta3=eta1+(eta2-eta1)*f
      x3=x1+(x2-x1)*f
      t3=t1+(t2-t1)*f
      go to 1
c
c     a solution found without interpolation.  change variable names
c     however to fit scheme used in above interpolation procedure.
c
   10 xi3=xi2
      eta3=eta2
      x3=x2
      t3=t2
c
c     increment nwin to indicate another solution along the emerging ray
c     through xright.  another solution will be sought but first set
c     icheck=0 to show that tlr2 was positive and that it will have to
c     go negative again before another solution is possible.
c
    1 nwin=nwin+1
      icheck=0
      if(nwin.eq.1)go to 7
c
c     if nwin>1  a previous solution along xright was already found.
      write(jtty,1039)
 1039 format(" a multiple solution found with values:")
c
c     put the answers into arrays.
c
    7 nisect=nisect+1
      xi(nisect)=xi3
      eta(nisect)=eta3
      xplus(nisect)=x3
      xminus(nisect)=xright
      tau(nisect)=t3
c
c     calculate the velocity at (xi,eta) using critical angle approach
c
      phi=abs(apr(i)-apl(j))/2.
      vinst(nisect)=vpr(i)/sin(phi)
c
c     if incremental parameters can not be calculated go to 5
c
      if((nisect-nwin).eq.0)go to 5
      dell(nisect)=sqrt((xi3-xi(nisect-nwin))*(xi3-xi(nisect-nwin))
     &    +(eta3-eta(nisect-nwin))*(eta3-eta(nisect-nwin)))
      delt(nisect)=t3-tau(nisect-nwin)
      v(nisect)=0.
      if(delt(nisect).ne.0.)v(nisect)=dell(nisect)/delt(nisect)
      if(xi3.lt.xi(nisect-nwin))dell(nisect)=-dell(nisect)
c
c     determine the limits of the idth  horizon within which
c     solutions will be found.
c
      if(x3.gt.x3max)x3max=x3
      if(x3.lt.x3min)x3min=x3
      if(xright.gt.xrmax)xrmax=xright
      if(xright.lt.xrmin)xrmin=xright
      go to 222
c
c     the incremental parameters are set to 0 for the first solution.
c     also initialize x3max x3min xrmax and xrmin.
c
    5 x3max=x3
      x3min=x3
      xrmax=xright
      xrmin=xright
      dell(1)=0.
      delt(1)=0.
      v(1)=0.
  222 write(jtty,1052)xi(nisect),eta(nisect),vinst(nisect),v(nisect),
     & tau(nisect),delt(nisect),dell(nisect),xminus(nisect),xplus(nisect)
 1052 format(1x,f9.1,f10.1,2f8.1,2f8.2,1x,f8.2,2f9.1)
c
c     change variables and search for another solution.
c
    2 x1=x2
      xi1=xi2
      eta1=eta2
      t1=t2
      tlr1=tlr2
  360 continue
c
c     increment xright and try another ray unless the end has been reached.
c
  358 xright=xright+delx
      if(xright.ge.xmaxl .or. xright.gt.xmaxr)go to 600
      go to 500
  600 continue
c
c     print the range in which no solutions were found.
c
      write(jtty,1040)xminr,xrmin,xrmax,xmaxr
 1040 format(" for the right shotpoint, no solutions occur along the",/,
     & " reduced t-x graph in the intervals:",/,
     &  f10.1," to",f10.1,"  and",f10.1," to",f10.1)
      write(jtty,1041)xminl,x3min,x3max,xmaxl
 1041 format(" and for the left shotpoint, in the intervals:",/,
     & f10.1," to",f10.1,"  and",f10.1," to",f10.1)
      continue
      write(jtty,1066)
 1066 format(" do you wish the xstar points? yes or no:",$)
      read(itty,1016)go
      if(go.ne."yes")return
      write(jtty,1060)
 1060 format(" the path of special points (xstar) from horizon iu to id",/,
     &" left-hand shot-point",/,
```

```
     &"     x(iu)  time(iu)     x(id)  time(id)")
      write(jtty,1061)(xstarl(i),tstarl(i),xpriml(i),tpriml(i),i=1,nstarl)
 1061 format(4f10.1)
      write(jtty,1062)
 1062 format(" the right-hand shot-point")
      write(jtty,1061)(xstarr(i),tstarr(i),xprimr(i),tprimr(i),i=1,nstarr)
      return
      end


      subroutine select(xmin,xmax,ext,k,npts,t,x,jwatch,iflag)
c
c      selects a subset t( ),x( ) from arrival-time points corresponting to
c      a given shot indx k, begining at xmin and ending at xmax. xmin and
c      xmax are changed to agree with the precise limits of the new data
c      set.   xmin=lower bound of new data set   xmax=upper bound
c      ext=extension of data file, either org or com   npts=number of points
c      in new array   x=distance coordinates of new array   t=arrival-times
c      of new array   jwatch=0 and becomes 1 if npts .le. 1
c      iflag=0 and becomes 1 if the arrays gx(i) or gxc(i) are not in order.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
      common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      character*3 ext
      dimension t(400),x(400)
      iflag=0
      jwatch=0
c
c      determine extension of points to be used and dump t-x  arrays into
c      arrays t( ) and x( ).
c
      if(ext.eq."org")go  to 11
      ngk=ngc(k)
      if(ngk.le.1.or.ngk.gt.200)go to 990
      x(1)=gxc(k,1)
      t(1)=gtc(k,1)
      do 12 i=2,ngk
      x(i)=gxc(k,i)
      t(i)=gtc(k,i)
      if(x(i).le.x(i-1))go to 995
   12 continue
      go to 13
   11 ngk=ng(k)
      if(ngk.le.1.or.ngk.gt.24)go to 990
      x(1)=gx(k,1)
      t(1)=gt(k,1)
      do 14 i=2,ngk
      x(i)=gx(k,i)
      t(i)=gt(k,i)
      if(x(i).le.x(i-1))go to 995
   14 continue
c
c      select appropriate points from the input files
c
   13 npts=0
      icount=1
   16 if(icount.gt.ngk)go to 100
      if(x(icount).ge.xmin.and.x(icount).le.xmax)go to 15
      icount=icount+1
      go to 16
   15 npts=npts+1
      x(npts)=x(icount)
      t(npts)=t(icount)
      icount=icount+1
      go to 16
c
c      direct execution for return
c
  100 if(npts.gt.1)go to 200
      write(jtty,1000)npts,xmin,xmax
 1000 format("0sub select found only",i5," points within the prescribed"
     &,/,"  interval. xmin=",f9.2," and xmax=",f9.2,". jwatch=1")
      jwatch=1
      return
  995 write(jtty,1020)i,x(i),x(i+1)
 1020 format("0sub select found that at jug number",i5,/,
     &" the distance coordinates are decreasing.",/,
     &" distances are",2f9.2)
      iflag=1
      return
  990 write(jtty,1010)ngk
 1010 format("0sub select found",i5," geophones.your t-x files are bad")
      iflag=1
      return
c
c     redefine limits xmin and xmax
c
  200 xmin=x(1)
      xmax=x(npts)
      return
      end


         subroutine splin1(m,h,x,y,a,b,c,t,d,p,s)
c--one dimensional cubic spline interpolation
c
c        by  w.l.anderson, u.s. geological survey,
c             denver, colorado
c
c  parms--- m= number of data points .gt. 2
c           h= equal interval option when h.gt.0.
c              (use dummy x here),
c              unequal intervals if h=0. (x required storage)
c           x= indep.var when h=0. (dim .ge. m).
c           y= dependent variable  (dim .ge. m).
```

```
c           a,b,c=coeff.arrays (each dim .ge. m)
c                 results are returned in 1st(m-1) elements of
c                 a,b,&c.
c                 also used as work arrays during execution.
c           t= type of boundary condition supplied in d array.
c              use t=1 if 1st derivatives given at end points, or
c              t=0 if 2nd derivatives given at end points.
c           d= boundary array (dim 2) at point 1 and m
c              respectively.
c           p,s= work arrays (each dim=m).
c--error return with m=-(abs(m)) if any parm out of range.
c  the resulting cubic spline is of the form:
c     y=y(i)+a(i)*(x-x(i))+b(i)*(x-x(i))**2+c(i)*(x-x(i))**3
c       for i=1,2,...,m-1
c
          real   x(1),y(2),a(2),b(100),c(100),d(2),
          & p(100),s(100),mul
          integer  t
          if(t.lt.0.or.t.gt.1.or.m.lt.3) go to 999
          n=m-1
          if(t.eq.0) go to 20
c--1st derivative boundaries given
          ne=n-1
          if(h) 1,11,1
c--equal spacing h .gt. 0. and t=1
1         hh=3.0/h
          do 2 i=1,ne
          b(i)=4.0
          c(i)=1.0
          a(i)=1.0
2         p(i)=hh*(y(i+2)-y(i))
          p(1)=p(1)-d(1)
          p(ne)=p(ne)-d(2)
c--solution of tridiagonal matrix eq. of order ne
3         fa=1./b(1)
          c(1)=c(1)*fa
          p(1)=p(1)*fa
          do 4 i=2,ne
          mul=1.0/(b(i)-a(i)*c(i-1))
          c(i)=mul*c(i)
4         p(i)=mul*(p(i)-a(i)*p(i-1))
c--obtain spline coefficients
          a(ne+t)=p(ne)
          i=ne-1
5         a(i+t)=p(i)-c(i)*a(i+t+1)
          i=i-1
           if(i.ge.1) go to 5
          if(t.eq.0) go to 6
          a(1)=d(1)
          a(m)=d(2)
6         if(h.eq.0.) go to 14
          hh=1.0/h
          do 7 i=1,n
          mul=hh*(y(i+1)-y(i))
          b(i)=hh*(3.0*mul-(a(i+1)+2.0*a(i)))
7         c(i)=hh*hh*(-2.0*mul+a(i+1)+a(i))
          return
c--unequal spacing h=0.. and t=1
11        do 12 i=1,n
12        s(i+1)=x(i+1)-x(i)
          do 13 i=1,ne
          b(i)=2.0*(s(i+1)+s(i+2))
          c(i)=s(i+1)
          a(i)=s(i+2)
13        p(i)=3.0*(s(i+1)**2*(y(i+2)-y(i+1))+s(i+2)**2*(y(i+1)-y(i)))/
          &      (s(i+1)*s(i+2))
          p(1)=p(1)-s(3)*d(1)
          p(ne)=p(ne)-s(n)*d(2)
          go to 3
14        do 15 i=1,n
          hh=1.0/s(i+1)
          mul=(y(i+1)-y(i))*hh**2
          b(i)=3.0*mul-(a(i+1)+2.0*a(i))*hh
15        c(i)=-2.0*mul*hh+(a(i+1)+a(i))*hh**2
          return
c--2nd derivative boundaries given
20        ne=n+1
          if(h) 21,31,21
c--equal spacing h .gt. 0 and t=0
21        hh=3.0/h
          do 22 i=2,n
          b(i)=4.0
          c(i)=1.0
          a(i)=1.0
22        p(i)=hh*(y(i+1)-y(i-1))
          b(1)=2.0
          b(ne)=2.0
          c(1)=1.0
          c(ne)=1.0
          a(ne)=1.0
          p(1)=hh*(y(2)-y(1))-0.5*h*d(1)
          p(ne)=hh*(y(m)-y(n))+0.5*h*d(2)
          go to 3
c--unequal spacing h=0 and t=0
31        do 32 i=1,n
32        s(i+1)=x(i+1)-x(i)
          n1=n-1
          do 33 i=1,n1
          b(i+1)=2.0*(s(i+1)+s(i+2))
          c(i+1)=s(i+1)
          a(i+1)=s(i+2)
33        p(i+1)=3.0*(s(i+1)**2*(y(i+2)-y(i+1))+s(i+2)**2*(y(i+1)-y(i)))/
          &      (s(i+1)*s(i+2))
          b(1)=2.0
          b(ne)=2.0
          c(1)=1.0
          c(ne)=1.0
          a(ne)=1.0
```

```
          p(1)=3.0*(y(2)-y(1))/s(2)-0.5*s(2)*d(1)
          p(ne)=3.0*(y(m)-y(n))/s(m)+0.5*s(m)*d(2)
          go to 3
999       m=-iabs(m)
          return
          end


      subroutine splindt(m,x,y,a,b,c,xx,yy,deriv)
c  given cubic spline coefficients a,b,c, and m observation data arrays
c  x,y  splindt evaluates the piecewise cubic spline ordinate yy and its
c  derivative deriv at the abscissa xx, where xx is in the closed
c  interval (x(1),x(m)).
c
      dimension x(400),y(400),a(400),b(400),c(400)
      if(xx.lt.x(1) .or. xx.gt.x(m))go to 9
      ml=m-1
      do 1 i=1,ml
      j=i
      if(xx.le.x(i+1))go to 2
    1 continue
    9 write(6,60)xx,x(1),x(m)
   60 format(" error in splindt call--xx=",f10.1," not in closed
     & interval",f10.1,",",f10.1)
      return
    2 z=xx-x(j)
      yy=y(j)+((c(j)*z+b(j))*z+a(j))*z
      deriv=(3.*c(j)*z+2.*b(j))*z+a(j)
      return
      end


      subroutine term
      external print(descriptors)
      character*50 file
      write(6,1000)
 1000 format(" type input filename and extension: "$)
      read(5,1100) file
 1100 format(v)
c
c  NOTE: print is a system subroutine on the honeywell 68/88
c
      call print(file)
  990 return
      end


      subroutine waved(x,t,nobs,iu,id,delx,dint,nprint,xstar,
     & nstar,curv,nval,tstar,xprim,tprim,idim,isplin,bend)
c
c     given a set of t-x data points defined on horizon iu, subroutine
c     waved determines a new set of data points reduced to horizon id.
c     it also takes certain equally spaced points on the t-x
c     curve, denoted by xstar(i), and moves them down to the idth.
c     horizon, keeping track of their location and time for the
c     users possible consideration later on.
c
      dimension gt(50,24),gx(50,24),sx(50),sd(50),ng(50),nc(50),
     &cx(10,50),cy(10,50),cv(10,50),gtc(10,200),gxc(10,200),sxc(10),
     &sdc(10),ngc(10),offset(50),offsc(10)
       common/data1/ gt,gx,sx,sd,ng,nc,cx,cy,cv,gtc,gxc,sxc,sdc,ngc,
     & itty,jtty,offset,offsc
      dimension x(400),t(400),xstar(30),tstar(30),xprim(30),
     & tprim(30),xx(400),tt(400)
      dimension d(2),p(400),s(400)
      dimension a(400),b(400),c(400)
      character*3 curv,bend
      external raydl(descriptors)
      d(1)=0.
      d(2)=0.
c
c     initialize
c
      isplin=0
      idim=0
c
c     if the index of the upper layer exceeds or equals that of the lower
c     layer, the subroutine will not be executed.
c
      if(iu.ge.id)return
      if(nobs.le.400)go to 32
      write(jtty,1008)nobs
 1003 format(" waved condition 1. nobs=",i5)
      idim=1
      return
c
c     preserve iu.
c
   32 iq=iu
c
c     take the values of the special "star" points xstar(i), and preserve
c     them by putting them into array xprim(i).
c
      do 21 i=1,nstar
   21 xprim(i)=xstar(i)
c
c     subdivide the arrival-time curve into equally spaced intervals.
c
  500 call eqspace(x,t,nobs,dint,idim)
      if(idim.eq.0)go to 51
      write(jtty,1009)nobs
 1009 format(" waved condition 2. nobs=",i5)
      idim=1
      return
c
c     t-x points could be filtered at this point.
c      51 call fltr(x,t,nobs,flt)
c
```

```
   51 xmin=x(1)
      xmax=x(nobs)
      if(curv.ne."spl")go to 52
c
c     if spline function is used, determine their coefficients a, b, and c
c     for equally spaced data points.
c
      call splinl(nobs,0.,x,t,a,b,c,0,d,p,s)
      if(nobs.gt.0)go to 52
      write(jtty,1010)
 1010 format(" waved condition 3")
      isplin=1
      return
c
c     initialize
c
   52 ipoint=1
      ir=iq+1
      xp=xmin
    4 continue
      if(curv.ne."spl")go to 57
c
c     calculate the arrival-time and slope of curve at distance coordinate
c     xp using either the spline or line approximation.
c
      call splindt(nobs,x,t,a,b,c,xp,time,deriv)
      go to 55
   57 call lindt(nobs,x,t,xp,nval,time,deriv)
   55 call comp(xp,iq,elev,x1,y1,x2,y2,v0,j,iflag)
      if(iflag.eq.0)go to 24
      write(jtty,1000)xp,iq
 1000 format(" waved condition 4 at xp=",f10.1," horizon",i5)
      go to 26
c
c     calculate the angle of submergence at xp.
c
   24 f=v0*deriv
      if(abs(f).lt.1.)go to 1
      vel=1./deriv
      write(jtty,1001)xp,iq,vel
 1001 format(" waved condition 5 at xp=",f10.1," horizon",i5,
     & " velocity",f10.1)
      go to 26
    1 beta=asin(f)
c
c     initialize.
c
      taut=0.
c
c     move the ray down one layer.
c
      call raydl(xp,iq,ir,beta,xd,yd,taut,iflag,bend)
      if(iflag.eq.0)go to 25
      write(jtty,1002)xp,iq
 1002 format(" waved condition 6 at xp=",f10.1," horizon",i5)
      go to 26
c
c     put the distance coordinate and travel-time of the
c     new reduced point into arrays.
c
   25 xx(ipoint)=xd
      tt(ipoint)=time-taut
   26 if(xp.ge.xmax)go to 3
c
c     not all done.    increment.
c
      xp=xp+delx
      if(xp.gt.xmax)xp=xmax
      ipoint=ipoint+1
      if(ipoint.le.400)go to 4
      write(jtty,1014)iq,xp
 1014 format(" waved condition 7 from horizon",i5," at xp=",f10.1)
      idim=1
      return
c
c     all done for this layer.    re-arrange the new points in increasing
c     values of x and check for crossovers.
c
    3 ishift=0
   54 continue
      do 5 i=1,ipoint-1
      if(xx(i).ne.xx(i+1))go to 53
      xx(i+1)=xx(i+1)+.000001*abs(xx(i+1))
      ishift=ishift+1
   53 xsmall=aminl(xx(i),xx(i+1))
      if(xsmall.eq.xx(i))go to 5
      ishift=ishift+1
      tsmall=tt(i+1)
      xx(i+1)=xx(i)
      tt(i+1)=tt(i)
      xx(i)=xsmall
      tt(i)=tsmall
      go to 54
    5 continue
c
c     do the special "star" points by calculating the travel-times and
c     the starting and ending points for the xstar(i).
c
      if(nstar.eq.0)go to 35
      do 6 i=1,nstar
      xp=xprim(i)
      if(xp.ge.xmin .and. xp.le.xmax)go to 56
      write(jtty,1012)xp,iq,xstar(i)
 1012 format(" waved condition 8 at xp=",f10.1," horizon",i5,
     &" originating at xstr=",f10.1)
      xprim(i)=1.e6
      go to 6
c
```

```
c     calculate the arrival-time and slope of the t-x graph using either
c     the line or spline routine.
c
   56 if(curv.ne."spl")go to 58
      call splindt(nobs,x,t,a,b,c,xp,time,deriv)
      go to 59
   58 call lindt(nobs,x,t,xp,nval,time,deriv)
   59 if(iq.eq.iu)tstar(i)=time
      call comp(xp,iq,elev,x1,y1,x2,y2,v0,j,iflag)
      if(iflag.eq.0)go to 27
      write(jtty,1004)xp,iq,xstar(i)
 1004 format(" waved condition 9 at xp=",f10.1," horizon",i5,
     &" originating at",f10.1)
      xprim(i)=1.e6
      go to 6
c
c     calculate the submergence angle beta.
c
   27 f=v0*deriv
      if(abs(f).lt.1.)go to 28
      vel=1./deriv
      write(jtty,1005)xp,iq,vel,xstar(i)
 1005 format(" waved condition 10 at xp=",f10.1," horizon",i5,/,
     &" velocity",f10.1," originating at",f10.1)
      xprim(i)=1.e6
      go to 6
   28 beta=asin(f)
      taut=0.
c
c     and move point down one horizon.
c
      call raydl(xp,iq,ir,beta,xd,yd,taut,iflag,bend)
      if(iflag.eq.0)go to 29
      write(jtty,1006)xp,iq,xstar(i)
 1006 format(" waved condition 11 at xp=",f10.1," horizon",i5,
     &" originating at",f10.1)
      xprim(i)=1.e6
      go to 6
   29 xprim(i)=xd
      tprim(i)=time-taut
    6 continue
c
c     put the xx(i) into the x(i) array.
c
   35 do 34 i=1,ipoint
      x(i)=xx(i)
      t(i)=tt(i)
   34 continue
c
c     print the new t-x arrays if desired.
c
      if(nprint.eq.0)go to 30
      if(iq.gt.1)go to 31
      write(jtty,1013)
 1013 format(" intermediate arrival times from sub waved",/," layer  ",
     &4("distance time   "))
   31 write(jtty,1007)ir,(x(i),t(i),i=1,ipoint)
 1007 format(i5,/,(7x,4(2f7.1,3x)))
      write(jtty,1003)xstar(1),xprim(1)
 1003 format(" shot originally at x=",f8.1," now at x=",f8.1)
   30 nobs=ipoint
      iq=iq+1
c
c     all finished if iq=id.   otherwise do the next layer.
c
      if(iq.eq.id)return
      go to 500
      end


      subroutine xisect(xq,yq,aq,tq,vq,v2,xis,etas,xie,etae,taue,ib)
c
c     given an emergent ray defined with an upper end point (xq,yq) with
c     emergent angle aq and emergent velocity vq;   and also given the
c     critical subsurface point defined by (xis,etas) with a latterally
c     propagating ray of velocity v2 such that it intersects the
c     emergent ray resulting in a total travel time tq, between
c     (xis,etas) and (xq,yq);   calculate the intersection point
c     (xie,etae) between the two rays such that the total travel time
c     tq,  is satisfied.   also calculate the travel time taue of the
c     laterally propagating portion of the combined rays.
c     ib is normally zero but changes to some other valye
c     if no solution cal be found.
c
      jtty=6
      sq1=vq*cos(aq)
      if(sq1.eq.0.)go to 500
      v22=v2*v2
      taq=sin(aq)/cos(aq)
      u=yq/sq1-tq
      al=xis+yq*taq-xq
      c=etas*etas+al*al-v22*u*u
      b=2.*(etas+al*taq-u*v22/sq1)
      a=1.+taq*taq-v22/(sq1*sq1)
      sq2=b*b-4.*a*c
      if(sq2.lt.0.)go to 550
      if(a.eq.0.)go to 560
      etae=(b-sqrt(sq2))/(2.*a)
      xie=xq-taq*(yq-etae)
      taue=tq-(yq-etae)/sq1
      ib=0
 1    continue
      return
 500  write(jtty,501)
 501  format(" xisect error ")
      write(jtty,502)
 502  format(" xisect says vq*cos(aq)=0. ")
      ib=1
```

```
      return
550   write(jtty,501)
      write(jtty,551)
551   format(" xisect says imaginary solution")
      ib=2
      return
560   write(jtty,501)
      write(jtty,561)
561   format(" xisect says subquantity a=0. ")
      ib=3
      return
      end


      subroutine xtend(x,t,nobs,nopt,dx,fn,ioptl,iopt3,idim)
c
c     adds a few points on to arrays (x(i),t(i)) i=1,nobs according to
c     rules specified by nopt.
c     case i  if nopt=0, no points are added.
c     case ii  If nopt=1, add dx number of points contained in array
c     fn(i) to either the beginning or the end of the arrays x(i),t(i).
c     case iii  if nopt=2, add a single point to the array x(i),t(i)
c     according to the velocity specified in fn(1).  if dx is negative the
c     point is added to the beginning of the arrays and if it is negative,
c     to the end of the arrays.
c     case iv  If nopt=3, add a single point to the arrays using a least
c     squares straight line approximation through the number of points by
c     fn(1).  if dx is negative, the point is added to the beginning of the
c     arrays x(i),t(i)  and if it is negative, to the end of the arrays.
c     iopt2=1 if any of the points being added in case ii fall within
c     the range of the x(i).
c     iopt3=1 if fewer points exist in array than are to be used for
c     least squares fit in case iv.
c     idim=1 if program dimensions are exceeded.
c
      dimension x(400),t(400),fn(10),xx(20),tt(20)
c
c     initialize.   these variables change value if problems arise.
c
      idim=0
      ioptl=0
      iopt3=0
c
c     case i  nopt=0.  no points are added to the t-x arrays.
c
      if(nopt.eq.0)return
      if(nopt.ne.1)go to 100
c
c     case ii  nopt=1  extend the data to include points within the array fn(i).
c
      nx=dx
c
c     determine whether points will be added to beginning or end of arrays.
c
      if(fn(1).lt.x(1))go to 1
c
c     add nx points to the end.
c
      n=nobs+1
      xmax=x(nobs)
      nobs=nobs+nx
      j=1
      do 2 i=n,nobs
      x(i)=fn(j)
      if(x(i).gt.xmax)go to 21
      ioptl=1
      return
   21 t(i)=fn(j+nx)
      j=j+1
    2 continue
      go to 300
c
c     add nx points to the beginning.  first make room for these points.
c
    1 j=nobs+nx
      xmin=x(1)
      do 3 i=1,nobs
      x(j)=x(j-nx)
      t(j)=t(j-nx)
      j=j-1
    3 continue
      nobs=nobs+nx
      do 4 i=1,nx
      x(i)=fn(i)
      if(x(i).lt.xmin)go to 22
      ioptl=1
      return
   22 t(i)=fn(i+nx)
    4 continue
      go to 300
  100 if(nopt.ne.2)go to 101
c
c     case iii  nopt=2   add one point to either end using the velocity
c     specified in fn(1) at the interval in dx.
c
      v=fn(1)
      if(dx.lt.0.)go to 5
c
c     add apoint to the end of the array.
c
      x(nobs+1)=x(nobs)+dx
      t(nobs+1)=dx/v+t(nobs)
      nobs=nobs+1
      go to 300
c
c     add a point to the beginning of the array.  first make room for it.
c
    5 j=nobs+1
```

```
      do 6 i=1,nobs
      x(j)=x(j-1)
      t(j)=t(j-1)
      j=j-1
    6 continue
      x(1)=x(2)+dx
      t(1)=t(2)+dx/v
      nobs=nobs+1
      go to 300
c
c     case iv  nopt=3  add one point to either end using a least squares
c     straight line approximation thru fn(1) points at the interval dx.
c
  101 if(nopt.ne.3)go to 102
      nx=fn(1)
c
c     at least two points must be used but they may not exceed nobs.
c
      if(nx.le.nobs .and. nx.ge.2)go to 23
      iopt3=1
      return
   23 if(dx.gt.0.)go to 7
c
c     dx<0  add a point to the beginning of the arrays.  first put points
c     to be used into another array.
c
      do 8 i=1,nx
      xx(i)=x(i)
      tt(i)=t(i)
    8 continue
      go to 10
c
c     dx>0  add a point to the end of the array.
c
    7 j=nobs+1-nx
      do 9 i=1,nx
      xx(i)=x(j)
      tt(i)=t(j)
      j=j+1
    9 continue
c
c     calculate least squares straight line coefficients a1 and a2.
c     see sokolnikoff 1941 p539.
c
   10 c1=nx
      c2=0.
      c3=0.
      c5=0.
      c6=0.
      do 11 i=1,nx
      c2=c2+xx(i)
      c3=c3+tt(i)
      c5=c5+xx(i)*xx(i)
   11 c6=c6+xx(i)*tt(i)
      c4=c2
      a2=(c1*c6-c3*c4)/(c1*c5-c2*c4)
      a1=(c3-c2*a2)/c1
      if(dx.gt.0.)go to 12
c
c     add point to beginning of arrays.  first make room for it.
c
      j=nobs+1
      do 13 i=1,nobs
      x(j)=x(j-1)
      t(j)=t(j-1)
      j=j-1
   13 continue
      x(1)=x(2)+dx
      t(1)=a1+a2*x(1)
      t(2)=a1+a2*x(2)
      nobs=nobs+1
      go to 300
c
c     add point to end of arrays.
c
   12 nobs=nobs+1
      x(nobs)=x(nobs-1)+dx
      t(nobs)=a1+a2*x(nobs)
      t(nobs-1)=a1+a2*x(nobs-1)
  300 if(nobs.le.400)return
c
c     program dimensions are exceeded.
c
      idim=1
  102 return
      end
```

REFERENCES CITED

Anderson, W. L., 1971, Application of bicubic spline functions to two-dimensional gridded data, PB 203 579, NTIS, Springfield, VA., p. 61-63.

Hawkins, L. V., 1961, The reciprocal method of routine shallow seismic refraction investigations, Geophysics, v. 18, no. 3, p. 806-819.

Nettleton, L. L., 1940, Geophysical prospecting for oil, McGraw-Hill Book Co., Inc., NY, 443 p.

Rockwell, D. W., 1967, A general wavefront method in seismic refraction prospecting: SEG, Tulsa, p. 363-415.

Schenck, F. L., 1967, Refraction solutions and wavefront targeting: in seismic refraction prospecting: SEG, Tulsa, p. 416-425.

Scott, J. H., Tibbetts, B. L., Burdick, R. G., 1972, Computer analysis of seismic refraction data, RI 7595, U.S. Dept. of Interior, Bureau of Mines, Washington, D.C., 95 p.

Slotnick, M. M., 1950, A graphical method for the interpretation of refraction profile data: Geophysics, v. 15, no. 2, p. 163-180.

Slotnick, M. M., 1959, Lessons in seismic computing, edited by R. A. Geyer: SEG, Tulsa, 268 p.

Thornburgh, H. R., 1930, Wavefront diagrams in seismic interpretation: AAPG Bull., v. 14, p. 185-200.