NRL Report 7108

Computer Program for Calculating Propagation Loss into the Shadow Zone

H. N. Van Ness

Acoustic Warfare Branch
Acoustics Division

August 26, 1970

NAVAL RESEARCH LABORATORY
Washington, D.C.

This document has been approved for public release and sale; its distribution is unlimited.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Naval Research Laboratory Washington, D.C. 20390	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

COMPUTER PROGRAM FOR CALCULATING PROPAGATION LOSS INTO THE SHADOW ZONE

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

A final report on the problem.

5. AUTHOR(S) *(First name, middle initial, last name)*

Hanford N. Van Ness

6. REPORT DATE August 26, 1970	7a. TOTAL NO. OF PAGES 44	7b. NO. OF REFS 5
8a. CONTRACT OR GRANT NO. NRL Problem S01-31 b. PROJECT NO. Project SF 11-552-001-8608 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) NRL Report 7108 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Department of the Navy (Naval Ship Systems Command), Washington, D. C. 20360

13. ABSTRACT

A computer program has been developed for calculating the propagation loss of underwater sound into the region of the shadow zone. The calculations are based on a composite propagation scattering model. The computer program has been written partly in FORTRAN IV and partly in NELIAC. Copies of this program are available to interested parties.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Underwater sound propagation						
Acoustic scattering						
Sonar						
Sound speed profile						
Ray tracing						

DD FORM 1 NOV 65 1473 (BACK)
(PAGE 2)

Security Classification

CONTENTS

Abstract	ii
Problem Status	ii
Authorization	ii
INTRODUCTION	1
DESCRIPTION	1
GUIDE FOR USERS	3
FLOW DIAGRAM	4
REFERENCES	11
APPENDIX A – Program Listing for SURFSCAT	13

ABSTRACT

A computer program has been developed for calculating the propagation loss of underwater sound into the region of the shadow zone. The calculations are based on a composite propagation scattering model. The computer program has been written partly in FORTRAN IV and partly in NELIAC. Copies of this program are available to interested parties.

PROBLEM STATUS

This concludes the work on this problem, which was terminated June 30, 1969. Unless otherwise notified, this problem will be considered closed 30 days after the issuance of this report.

AUTHORIZATION

NRL Problem S01-31
Project SF-11-552-011-8608

Manuscript submitted March 31, 1970.

COMPUTER PROGRAM FOR CALCULATING PROPAGATION LOSS INTO THE SHADOW ZONE

INTRODUCTION

A program named Assured Range Studies was carried on by the Acoustic Warfare (formerly Techniques) Branch of the Acoustics Division. One of the results of the Assured Range Studies was the development of a means of computing the propagation loss of acoustic energy in the ocean from a source in the surface layer to any point below the layer. The purpose of this report is to describe the computer program for the calculation of this propagation loss.

It is a common occurrence for the ocean to develop a layer of water which is virtually isothermal. This layer of almost constant temperature starts just below the ocean surface and can extend downward for several hundred feet. The sound speed gradient in the isothermal layer is positive in the downward direction due to the increase in the hydrostatic water pressure with depth. Immediately below the layer the sound speed gradient is negative. This is caused by the relatively rapid decrease of water temperature with depth. Sound propagation in the layer, or surface duct as it is also called, is usually favorable.

For a sound source in the surface layer there exists a limit ray. The limit ray is the ray which leaves the source at an angle such that it crosses the bottom boundary of the duct at grazing incidence. Ray theory predicts no sound propagation below the duct boundary beyond the point of grazing incidence of the limit ray. This region is the shadow zone.

Experience has shown that the shadow zone is insonified, and the Assured Range Studies proposed a propagation scattering model for determining sound intensity there. Specifically, it was asserted that the principal mechanism for insonifying the region is the scattering of acoustic energy by the rough, irregular sea surface.

This propagation scattering model requires the knowledge of only those parameters that can be measured. The parameters are the sound speed variation with depth, wind speed, depth of the isothermal duct, depth of the sound source, range between source and receiver (or target), and depth of the receiver. (The terms *receiver* and *target* will be used interchangeably throughout this report.) However, the calculation of propagation loss is so intricate and lengthy that it is made practicable only by means of a digital computer. A brief description covering the important points of the propagation scattering model is given below.

DESCRIPTION

The propagation scattering model asserts that acoustic rays diverge from the source, where they are refracted in accordance with the sound speed gradient, and are then reflected and/or scattered from the surface into the shadow zone. The scattering occurs from an area between source and receiver. (It is assumed that backscattering is unimportant, and areas beyond the receiver make no significant contribution to the signal strength at the receiver. Also, there is no contribution by sound reflected from the ocean bottom.) The scattering strength of any point on the surface is a continuous

function of position with respect to the source and the receiver. But since the scattering cannot be computed over any continuous extent from a practical standpoint, the surface of the ocean between source and receiver is divided into a number of elementary scattering areas, in this case squares all of the same size. Such a division is diagrammed in Fig. 1, where S is the source and T is the target. The scattering strength is computed at the center of each of the elementary scattering areas, and it is assumed constant over the entire square.

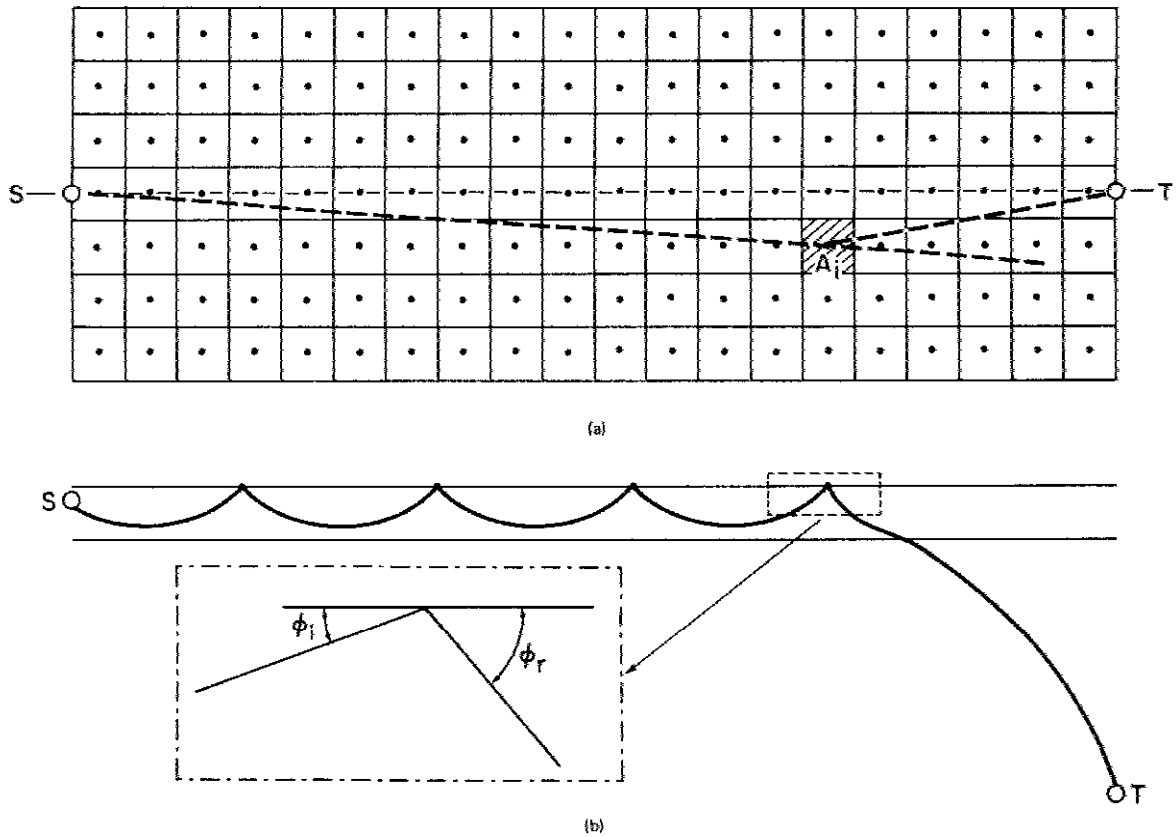

Fig. 1 - Diagram of scattered ray, (a) ocean surface divided into elementary scattering areas, (b) geometry of scattered ray

With the division of the ocean surface into the elementary scattering areas, a set of individual ray paths is thus defined. The individual ray path is the path from the source to the center of any scattering area and from there to the receiver. For the ith square the propagation loss H_i in decibels is computed and displayed in the computer output. Each value of H_i is converted to its equivalent intensity, and all the intensities are added together. From the sum of the equivalent intensities the total propagation loss is determined.

The number of ray paths or the total extent of the scattering surface considered is dependent on the limits set for scattering strength and the values of H_i for elementary scattering areas lying on the central axis between source and receiver. The scattering strength of a scattering area decreases as areas further from the central axis are considered. When the scattering strength falls below -60 dB, areas further off the central axis are not considered. The error caused by not considering a larger area is a small

fraction of a decibel. The computations start by considering the ocean surface at the receiver end of the propagation path and "working" back toward the source. The values of H_i for the elementary scattering areas reach a maximum, usually within 1000 yd of the target. That there is a maximum value for H_i is a conclusion made after several preliminary calculations had been made. When H_i for a square, whose center lies directly on the center line, is less than -150 dB, the computation of H_i for any other area is terminated and the summation of all the intensity equivalents of the H_i's is made. (Again, here, errors due to truncation are small.) This final value of H is the propagation loss to a point in the shadow zone, i.e., at the target. The signal strength at the receiver is due to the scattered field. This theory and the program do not consider the "direct" field that would be encountered in the duct, and the target below the duct receives only the energy in the scattered field. The computer printout also includes the scattering strength of each scattering area and the angle of arrival each ray makes with the target.

The propagation loss H_i, in decibels, of each ray is the sum of four terms, as given in Eq. (1).

$$H_i = TL_{1i} + 10 \log_{10} S_{si} + 10 \log_{10} A_i + TL_{2i} . \tag{1}$$

(The subscript i in Eq. (1) will not be used in the following discussion.) The loss TL_1 is the loss along the ray from the source to the scattering area. To compute this loss, the AMOS (1) formulations were used plus a surface-coupled loss, suggested by M. Schulkin (2), that the ray suffers on scattering from the surface.

The quantity S_s of Eq. (1) is the scattering coefficient of the scattering area, and 10 log S_s is called the scattering strength. S_s is computed using the scattering model developed by Beckmann-Spizzichino (3). Their development includes scattering from a moderately rough to rough, irregular surface depending on acoustic frequency, standard deviation of wave height, and the angle at which the target "sees" the scattering area. The elementary scattering area A is a square of 50 yd on a side.

The quantity TL_2 is the propagation loss the ray suffers from the area of scattering to the target. The loss TL_2 is found by a ray-tracing method outlined by Stewart (4). From bathymetric and salinity data the sound speed for any depth is computed using Wilson's (5) equations. Stewart's method generates a sound speed profile that is not only continuous at the layer boundaries, but the first derivative of the sound speed with respect to depth is also everywhere continuous. This method is adequate for computing TL_2 because the sound speed structure can be accurately portrayed below the duct as well as within the duct. In the region through which TL_1 is computed, the sound speed is a linear function of depth because of the isothermal water.

GUIDE FOR USERS

The input parameters for this computer program are all either chosen or measured except for PHISUB1 (see list below). These values are listed below with their equivalent FORTRAN designations.

This information goes on one data card which is read by a FORTRAN read instruction. This single data card is to be followed by the sound speed profile. The curvilinear sound speed profile is generated by another computer program SOUNDSPEED which is in the NELIAC compiler language. The inputs for the sound speed profile require either depth sound speed pairs or temperature and salinity data and their corresponding depths.

Preliminary calculations showed that the Beckmann-Spizzichino scattering model is quite insensitive to small changes of small angles of PHISUB1 (ϕ_i). Even for the deepest

Definition	FORTRAN Designation
Channel depth (yd)	HC
Receiver depth (yd)	RDEPTH
Angle of incidence of ray at surface (degrees)	PHISUB1
Wind (knots)	W
Acoustic frequency (kHz)	FREQ
Value of linear gradient in duct (1/sec)	GC
Range between source and target (yd)	RINITIAL
Source depth (yd)	ZSUB0
Surface temperature (°F)	TEMP

surface ducts PHISUB1 will be about 3.5 degrees at most. Therefore PHISUB1 can be arbitrarily set to any small angle less than 3.5 degrees. A value of 2 degrees was used by the author.

FLOW DIAGRAM

The flow diagram of Fig. 2 shows the general computation scheme. Figs. 3 through 6 are the major subroutines except for subroutines INITPTOP and PTOP. These latter two subroutines are in the NELIAC-N compiler language. Their purpose will be noted below.

The first five operations after start given in Fig. 2 read in all of the data except for the sound speed profile, print this data as an output, calculate several constants as noted in operations numbered 4 and 5, divide the ocean surface as illustrated in Fig. 1, and compute all ranges for the determinations of TL_1 and TL_2. Subroutine DATA, which computes the ranges for TL_1 and TL_2, is diagrammed in Fig. 3.

Subroutine INITPTOP calls subroutine RDPROFILE, which reads in and stores the sound speed profile. INITPTOP also calculates a table of sound speeds for use by PTOP. The quantities FLAG, I, MX, and N are counting and controlling indexes. The subroutine PTOP computes the propagation loss TL_2, the angle of scattering from the surface and the angle of arrival at the target. PTOP finds the solution ray subject to the constraints of the problem. The constraints placed on the ray are horizontal range, depth of the source (which is the surface), depth of the target, and the sound speed gradient. The angle of scattering from the surface comes about by selecting only that ray that leaves the surface such that it will arrive at the target, given those constraints. This angle is required for subroutine SSS, and it is designated ϕ_r in Fig. 1.

The subroutine SSS computes the scattering strength $10 \log_{10} S_s$ of Eq. (1) and subroutine CMPUTE H calculates TL_1 and makes the addition of the quantities indicated in Eq. (1). Subroutine TABLE(MX) prints the output of the program in tabular form. The output table format displays the quantities H, $10 \log_{10} S_s$, and ANGRCVR (angle of arrival at the target) for each elementary scattering area. The numbers are arranged on

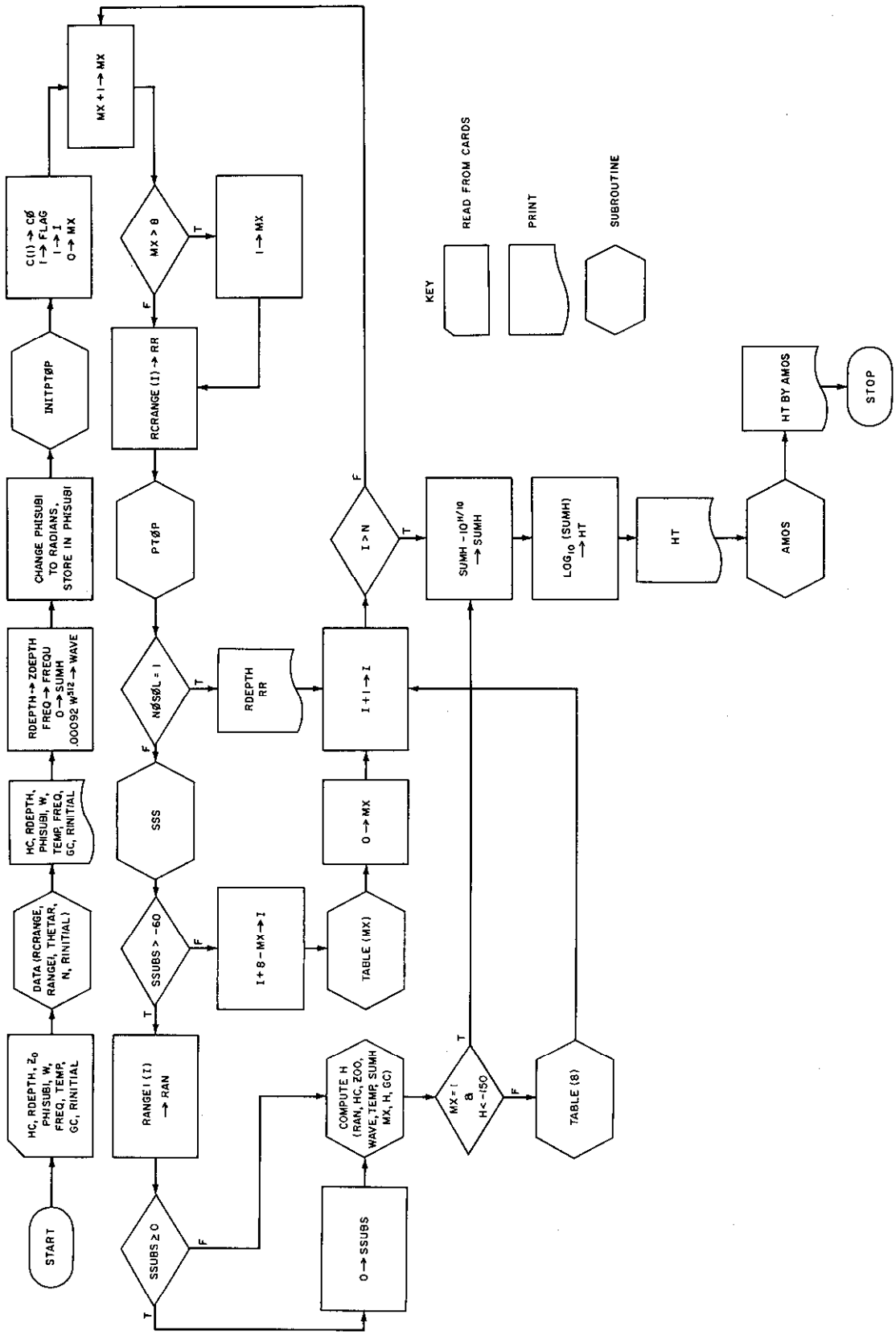

Fig. 2 - Flow diagram of main program for SURFSCAT

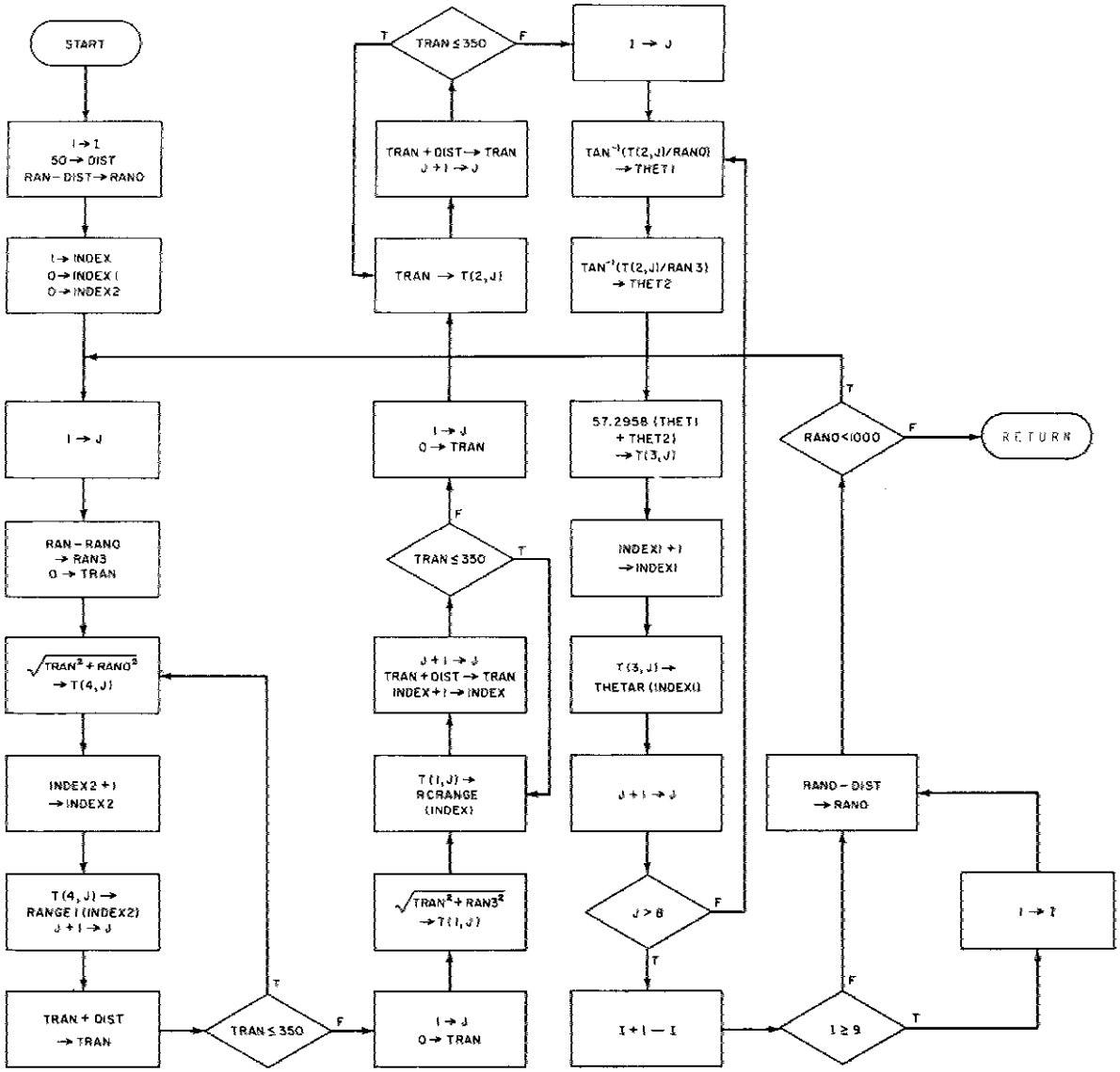

Fig. 3 - Flow diagram for range calculation for computing TL_1 and TL_2 and for calculation of θ_r

the page to correspond to the relative location of their position on the sea surface. But since the manner in which the surface is divided gives asymmetry with respect to the axis connecting source and target, only those values to one side (the right one) of this axis are printed.

The main program SURFSCAT computes the effective intensity for each ray, normalized to the source level of 1 yd, by taking $\log_{10}^{-1}(H_i/10)$. These effective intensities are summed, and then the total propagation loss between the source and the target is computed as $10 \log_{10}$ of the summed effective intensities.

Subroutine AMOS computes this same quantity using only the AMOS formulations throughout the entire extent of the transmission path.

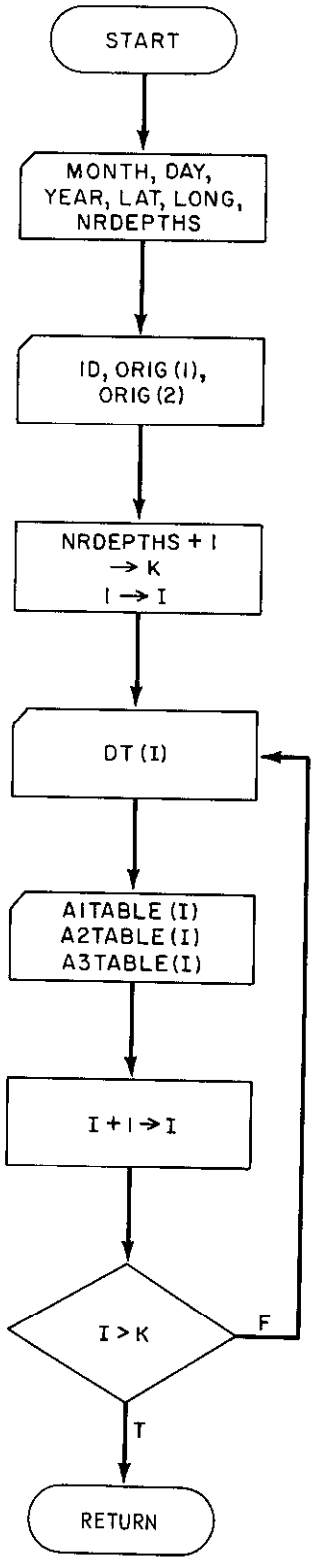

Fig. 4 - Flow diagram for reading the sound velocity profile

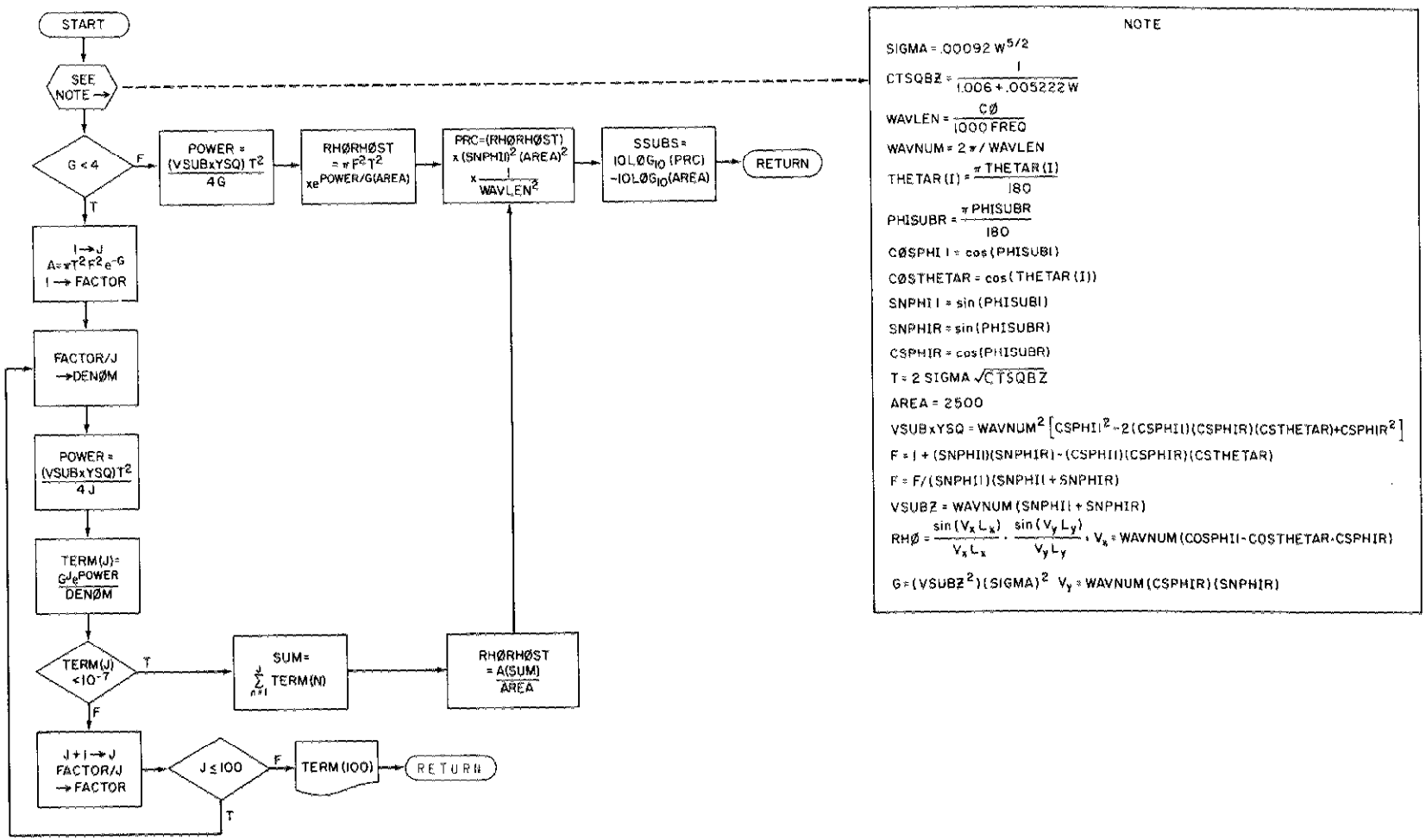

Fig. 5 - Flow diagram for calculation of $10 \log_{10} S_s$

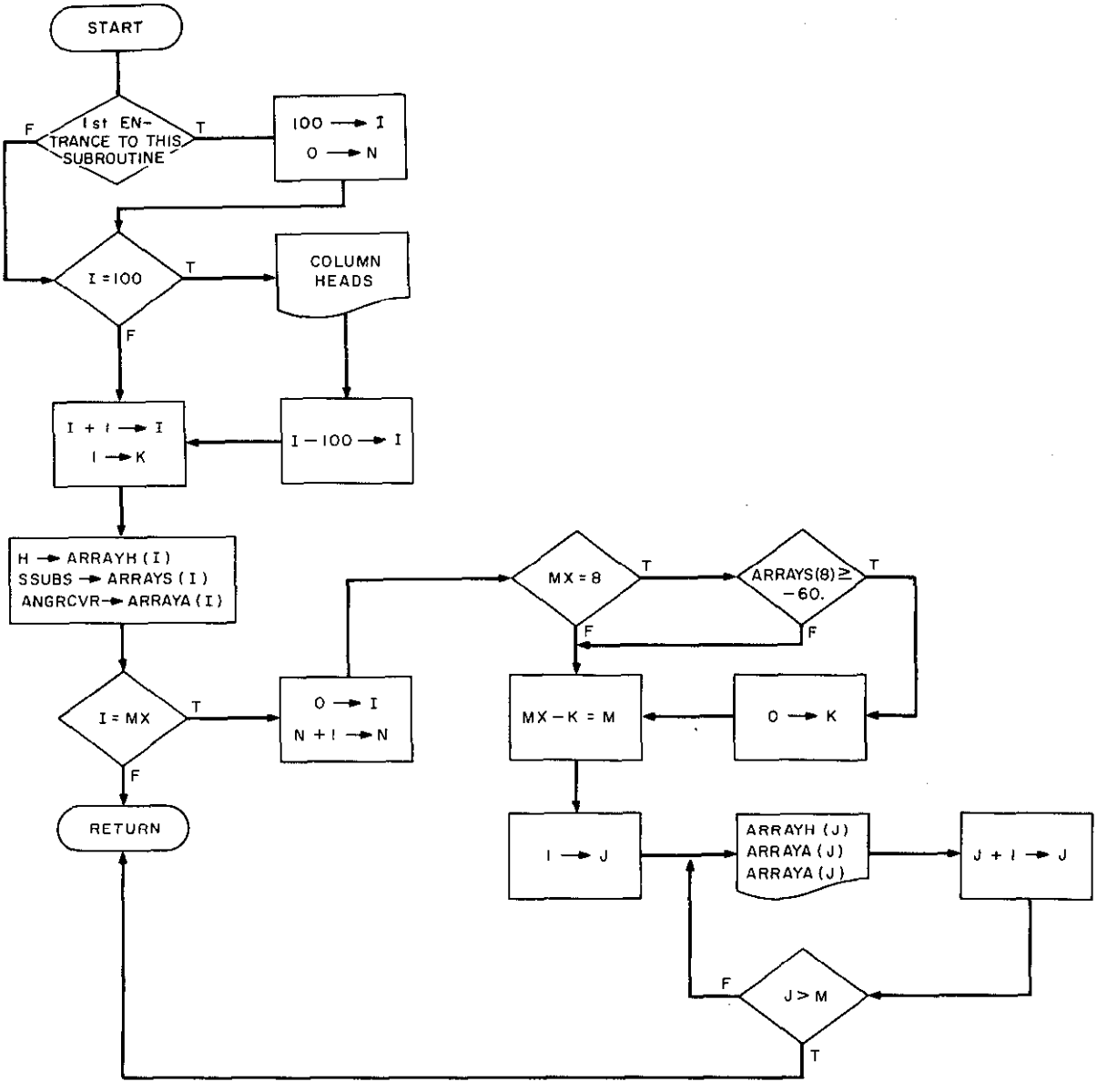

Fig. 6 - Flow diagram of printout

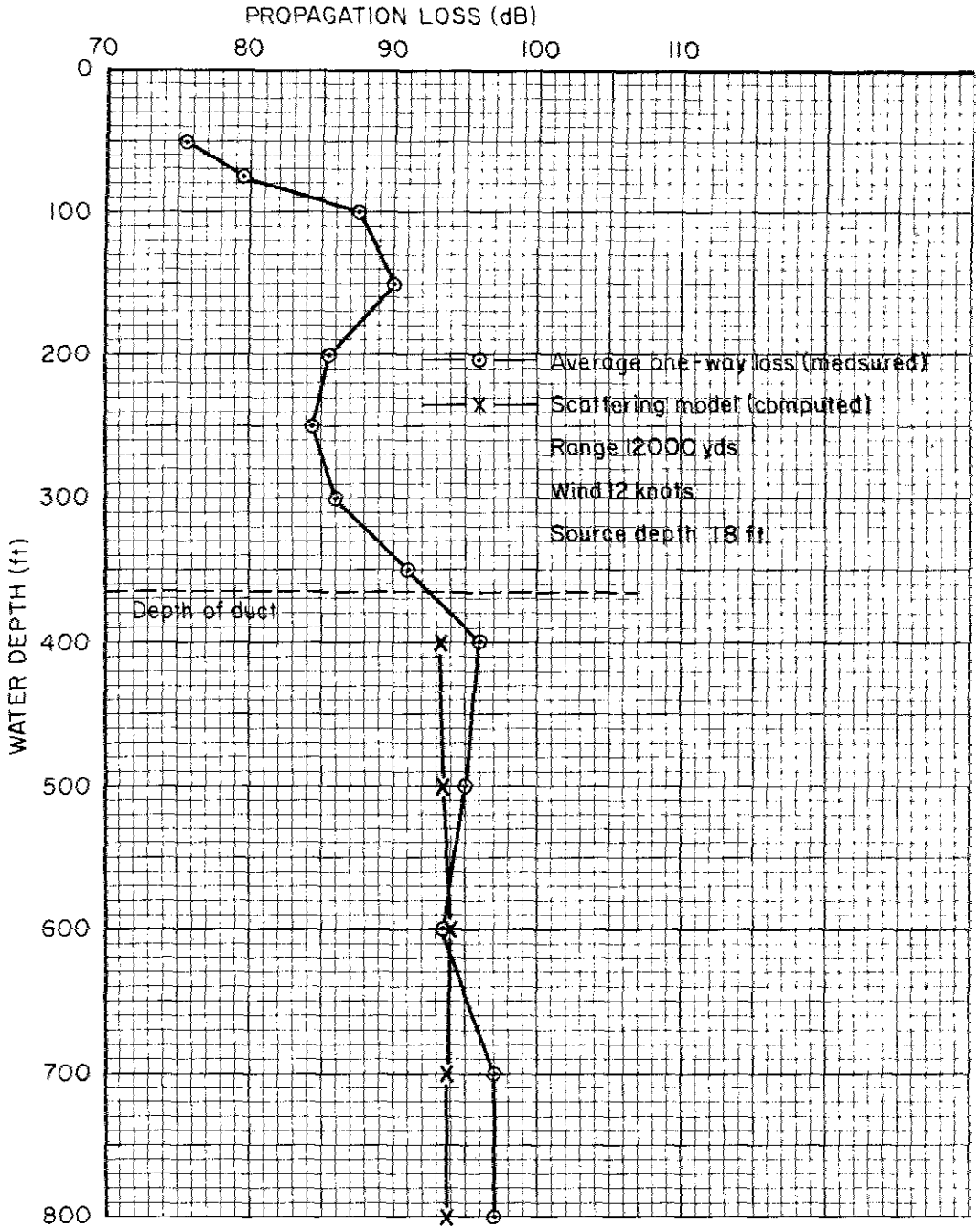

Fig. 7 - Propagation loss vs depth

A complete program listing is given in the Appendix. This computer program has been run repeatedly using field data collected during experiments at sea. These data were restricted to one frequency, but there were variations of range, target depth, surface duct depth, sound speed profile, and wind speed. The difference between computed and measured values of propagation loss never exceeded ±3.5 dB, where the measured value was based on the average loss for 20 to 30 pings. Shown in Fig. 7 are some typical propagation loss measurements for various depths for a constant range of 12,000 yards. The source depth is about 6 yd.

REFERENCES

1. March, H.W., Jr., and Schulkin, M., "Report on the Status of Project AMOS," USL Report 255A, May 9, 1967

2. Schulkin, M., J. Acoust. Soc. Amer. (Letter to the Editor), 44:1152-1154 (1968)

3. Beckmann, P., and Spizzichino, A., "The Scattering of Electromagnetic Waves from Rough Surfaces," New York:Pergamon, 1963

4. Stewart, K.R., J. Acoust. Soc. Amer. 38:339-347 (1965)

5. Wilson, W.D., J. Acoust. Soc. Amer. 32:641-644 (1960)

Appendix A

Program Listing for SURFSCAT

```
      PROGRAM SURFSCAT                                                  SUR00100
C                                                                       SUR00200
C  SURFSCAT COMPUTES THE PROPAGATION LOSS OF ACOUSTIC ENERGY EMITTED    SUR00300
C  FROM A SOURCE IN THE SURFACE DUCT SCATTERED INTO THE SHADOW ZONE.    SUR00400
C  THIS LOSS IS A FUNCTION OF WIND VELOCITY, TEMPERATURE, SOURCE AND    SUR00500
C  RECEIVER DEPTHS, DUCT DEPTH, FREQUENCY, RANGE, AND THE SOUND VELOCITYSUR00600
C  PROFILE.                                                             SUR00700
C                                                                       SUR00800
C  THE SCATTERING SURFACE IS DIVIDED INTO SQUARES OF EQUAL SIZE.  THE   SUR00900
C  SCATTERING STRENGTH IS COMPUTED AT THE CENTER OF EACH SQUARE, AND IT SUR01000
C  IS ASSUMED CONSTANT OVER THE ENTIRE SQUARE.                          SUR01100
C                                                                       SUR01200
      DIMENSION RCRANGE(3000),THETAR(3000)  ,RANGE1(3000)               SUR01300
      COMMON/SSS DATA/ANGSRCE,ANGRCUR,PLOSS,RR,RDEPTH,FREQ,NOSOL,FLAG,  SUR01400
     1CO,SSUBS                                                          SUR01500
      COMMON/SSS PRAM/W,PHISUB1,THETAR,I                                SUR01600
      COMMON/A/H                                                        SUR01700
      COMMON/PROFILE/MONTH,DAY,YEAR,LAT,LONG,ID,ORIG(2),NRDEPTHS,RT,    SUR01800
     *              DONC,MTHD,                                          SUR01900
     1DT(800), C(800), A1TABLE(800), A2TABLE(800), A3TABLE(800)          SUR02000
      COMMON/ANDY/HC, ZSUBC, ZDEPTH, RINITIAL, FEQU, WAVE, TEMP         SUR02100
C                                                                       SUR02300
C  RCRANGE(I) IS THE STRAIGHT LINE DISTANCE BETWEEN THE I TH SCATTERING SUR02400
C  POINT AND THE POINT ON THE SURFACE DIRECTLY ABOVE THE RECEIVER       SUR02500
C  (POINT A).                                                           SUR02600
C                                                                       SUR02700
C  RANGE1(I) IS THE DISTANCE BETWEEN THE SOURCE AND THE I TH SCATTERING SUR02800
C  POINT.                                                               SUR02900
C                                                                       SUR03000
C  THETAR(I) IS THE ANGLE BETWEEN THE VECTOR FROM THE SOURCE TO THE I THSUR03100
C  SCATTERING POINT AND THE VECTOR FROM THE I TH SCATTERING POINT TO    SUR03200
C  POINT A.                                                             SUR03300
C                                                                       SUR03400
C  RDEPTH IS THE RECEIVER DEPTH IN YARDS.                               SUR03500
C                                                                       SUR03600
C  HC IS THE CHANNEL DEPTH IN YARDS.                                    SUR03700
C                                                                       SUR03800
C  PHISUB1 IS THE ANGLE OF INCIDENCE A RAY MAKES WITH A SCATTERING      SUR03900
C  POINT.  IT IS ASSUMED CONSTANT FOR ALL RAYS AND SCATTERING           SUR04000
C  POINTS (USUALLY ABOUT 2 DEGREES).                                    SUR04100
C                                                                       SUR04200
C  W IS WIND SPEED IN KNOTS.                                            SUR04300
C                                                                       SUR04400
C  FREQ IS THE FREQUENCY OF THE ACOUSTIC SIGNAL IN KHZ.                 SUR04500
C                                                                       SUR04600
C  GC IS THE SOUND VELOCITY GRADIENT IN THE DUCT (A CONSTANT BY         SUR04700
C  DEFINITION OF THE DUCT) IN 1/SECS.                                   SUR04800
C                                                                       SUR04900
C  RINITIAL IS THE DISTANCE BETWEEN THE SOURCE AND POINT A.             SUR05000
C                                                                       SUR05100
C  ZSUBO IS THE DEPTH OF THE SOURCE IN YDS.                             SUR05200
C                                                                       SUR05300
C TEMP IS THE TEMPERATURE IN THE DUCT (ASSUMED CONSTANT).               SUR05400
C                                                                       SUR05500
      READ 1, HC, RDEPTH, PHISUB1, W, FREQ, GC, RINITIAL, ZSUBO, TEMP   SUR05600
    1 FORMAT(9F8)                                                       SUR05700
C                                                                       SUR05800
```

```
C  DATA COMPUTES THE VALUES OF RCRANGE(I), THETAR(I), RANGE1(I), AND     SUR05900
C  N, WHERE N IS THE TOTAL NUMBER OF SCATTERING POINTS TO BE CONSIDERED.SUR06000
C                                                                        SUR06100
      CALL DATA(RCRANGE,RANGE1,THETAR,N,RINITIAL)                        SUR06200
      PRINT 9                                                            SUR06300
    9 FORMAT(1X*INPUT DATA---*/)                                         SUR06400
      PRINT 10, HC, RDEPTH, PHISUB1, W, TEMP, FREQ, GC, RINITIAL         SUR06500
   10 FORMAT(5X*CHANNEL DEPTH*5X*RECEIVER DEPTH*5X*PHISUBI*5X*W*8X*TEMP*SUR06600
     1 10X*FREQ*5X*GAMMA CONSTANT*5X*INITIAL RANGE*/                     SUR06700
     25X,F10.3,10X,F10.3,3X,F10.3,3X,F6.3,5X,F6.3,5X,F8.3,3X,F10.5,10X,  SUR06800
     3F10.3//)                                                           SUR06900
      ZDEPTH = RDEPTH                                                    SUR07000
      FEQU = FREQ                                                        SUR07100
      SUMH = 0                                                           SUR07200
      WAVE = .00092*W**(5./2.)                                           SUR07300
      PI = 3.1415926535                                                  SUR07400
      CON = PI/180.0                                                     SUR07500
      PHISUB1 = PHISUB1*CON                                              SUR07600
C                                                                        SUR07700
C  INITPTOP IS A NELIAC SUBPROGRAM WHICH CALLS RDPROFILE AND RETURNS     SUR07800
C  DATA FOR PTOP.  RDPROFILE READS THE SOUND SPEED PROFILE WHICH MAY     SUR07900
C  BE GENERATED BY SOUND SPEED AVAILABLE THROUGH KEN MORIN, NRL.         SUR08000
C                                                                        SUR08100
      CALL INITPTOP                                                      SUR08200
      C0 = C(1)                                                          SUR08300
      FLAG = 1.0                                                         SUR08400
      MX = 0                                                             SUR08500
C                                                                        SUR08600
C  THE FOLLOWING LOOP COMPUTES THE TOTAL SCATTERING LOSS FROM SOURCE TO SUR08700
C  RECEIVER FOR EACH SCATTERING POINT IN DB (H), THE SCATTERING          SUR08800
C  STRENGTH OF EACH SCATTERING POINT IN DB (10LOG(SSUBS)), AND THE       SUR08900
C  ANGLE OF INCIDENCE OF THE RAY FROM EACH SCATTERING POINT AT THE       SUR09000
C  RECEIVER (ANGRCVR) IN DEGREES AND PRINTS THESE OUT IN TABULAR FORM.   SUR09100
C                                                                        SUR09200
      DO 50 I=1,N                                                        SUR09300
      MX = MX + 1                                                        SUR09400
      IF (MX.GT.8) MX = 1                                                SUR09500
      RR = RCRANGE(I)                                                    SUR09600
C                                                                        SUR09700
C  PTOP IS A NELIAC SUBPROGRAM WHICH COMPUTES THE PROPAGATION LOSS       SUR09800
C  FROM A SCATTERING POINT TO THE RECEIVER (TL2) AND THE ANGLE OF        SUR09900
C  DEPARTURE OF THE SCATTERED RAY FROM A SCATTERING POINT (PHISUBR).     SUR10000
C                                                                        SUR10100
      CALL PTOP                                                          SUR10200
      IF (NOSOL.EQ.1) PRINT 30,RDEPTH,RR                                 SUR10300
      IF (NOSOL.EQ.1) GO TO 50                                           SUR10400
   30 FORMAT(20X,*THERE IS NO SOLUTION FOR THIS DEPTH AND RANGE*,5X,     SUR10500
     *   F7.2,5X,F7.2//)                                                 SUR10600
C                                                                        SUR10700
C  SSS COMPUTES THE SCATTERING STRENGTH AT THE CENTER OF ONE OF THE      SUR10800
C  SCATTERING AREAS.                                                     SUR10900
C                                                                        SUR11000
      CALL SSS                                                           SUR11100
      IF (SSUBS.GT.-60.0) GO TO 40                                       SUR11200
      ADD = (8-MX)                                                       SUR11300
      I = I + ADD                                                        SUR11400
C                                                                        SUR11500
```

```
C   TABLE PRINTS OUT VALUES OF H, 10LOG(SSUBS), AND ANGRCVR IN          SUR11600
C   TABULAR FORM.                                                       SUR11700
C                                                                       SUR11800
      CALL TABLE(MX)                                                    SUR11900
      MX = 0                                                            SUR12000
      GO TO 50                                                          SUR12100
   40 RAN = RANGE1(I)                                                   SUR12200
C                                                                       SUR12400
C   COMPUTE H COMPUTES THE SCATTERING LOSS FROM THE SOURCE TO A         SUR12500
C   SCATTERING POINT (TL1), H AND SUMS THE EQUIVALENT INTENSITIES,      SUR12600
C   10**(H/10).                                                         SUR12700
C                                                                       SUR12800
      CALL CMPUTE H(RAN, HC, ZSUBOO, WAVE, TEMP, SUMH, MX, H, GC)       SUR12900
      IF(MX.EQ.1.AND.H.LT.-150.0) GO TO 55                              SUR13000
      CALL TABLE(H)                                                     SUR13100
   50 CONTINUE                                                          SUR13200
   55 SUMH = SUMH - 10**(H/10.0)                                        SUR13300
C   HT IS THE TOTAL PROPAGATION LOSS FOR ALL OF THE RAYS TRACES,        SUR13400
      HT = 10*ALOG10(SUMH)                                              SUR13500
      PRINT 60, HT                                                      SUR13600
   60 FORMAT(////20X* THE TOTAL PROPAGATION LOSS (HT) IS *F10.3/)       SUR13700
C                                                                       SUR13800
C   AMOS COMPUTES ANOTHER VALUE FOR HT USING THE AMOS FORMULATIONS.     SUR13900
C                                                                       SUR14000
      CALL AMOS                                                         SUR14100
      END                                                               SUR14200
```

```
      SUBROUTINE RDPROFLE                                               RDP 0100
C                                                                       RDP 0200
C   RDPROFILE READS THE SOUND SPEED PROFILE WHICH MAY BE GENERATED BY   RDP 0300
C   SOUND SPEED AVAILABLE THROUGH KEN MORIN, NPL.                       RDP 0400
C                                                                       RDP 0500
      COMMON/PROFILE/MONTH,DAY,YEAR,LAT,LONG,ID,ORIG(2),NRDEPTHS,       RDP 0600
     *            BT,DONC,MTHD,                                         RDP 0700
     1DT(800), C(800), A1TABLE(800), A2TABLE(800), A3TABLE(800)         RDP 0800
    1 READ 2,MONTH,DAY,YEAR,LAT,LONG,ID,ORIG(1),ORIG(2),NRDEPTHS        RDP 0900
    2 FORMAT(1X,I3,1X,I3,1X,I6,1X,F8,1X,F9,1X,A8,1X,2A8/I5)             RDP 1000
      K=NRDEPTHS+1                                                      RDP 1100
    5 READ 6,(DT(I),I=1,K)                                              RDP 1200
    6 FORMAT(4E18.9)                                                    RDP 1300
      READ 6,(C(I),I=1,K)                                               RDP 1400
      READ 6,(A1TABLE(I),I=1,K)                                         RDP 1500
      READ 6,(A2TABLE(I),I=1,K)                                         RDP 1600
      READ 6,(A3TABLE(I),I=1,K)                                         RDP 1700
      END                                                               RDP 1800
```

```
      SUBROUTINE SSS                                                    SSS 0100
C                                                                       SSS 0200
C  SUBROUTINE SSS COMPUTES THE SURFACE SCATTERING STRENGTH OF           SSS 0300
C  THE SURFACE WITH MEAN WAVE HEIGHT SIGMA AND MEAN SURFACE             SSS 0400
C  SLOPE 1./CTSQBZ, BOTH FUNCTIONS OF WIND SPEED W IN KNOTS             SSS 0500
C                                                                       SSS 0600
      DIMENSION TERM(100)                                               SSS 0700
      COMMON/SSS PRAM/W,PHISUB1,THETAR(3000),I                          SSS 0800
      COMMON/SSS DATA/PHISUBR,ANGRCUR,PLOSS,RR,RDEPTH,FREQ,NOSOL,FLAG   SSS 0900
     *,CO,SSUFS                                                         SSS 1000
      SIGMA = .00092*W**(5./2.)                                         SSS 1100
      CTSQBZ = 1./(.006 + .005202*W)                                    SSS 1200
      WAVLEN = CO/(1000.*FREQ)                                          SSS 1300
      PI = 3.1415926535                                                 SSS 1400
      WAVNUM = 2.*PI/WAVLEN                                             SSS 1500
      CON = PI/180.                                                     SSS 1600
      THETAR(I) = THETAR(I)*CON                                         SSS 1700
      PHISUBR = PHISUBR*CON                                             SSS 1800
      CSPHI1 = COS(PHISUB1)                                             SSS 1900
      SNTHETAR = SIN(THETAR(I))
      CSTHETAR = COS(THETAR(I))                                         SSS 2000
      SNPHI1 = SIN(PHISUB1)                                             SSS 2100
      SNPHIR = SIN(PHISUBR)                                             SSS 2200
      CSPHIR = COS(PHISUBR)                                             SSS 2300
      T = 2.*SIGMA*SQRTF(CTSQBZ)                                        SSS 2400
      AREA = 2500.                                                      SSS 2500
      VSUBXYSQ = WAVNUM**2*(CSPHI1**2 - 2.*CSPHI1*CSPHIR*CSTHETAR + CSPHSSS 2600
     *IR**2)                                                            SSS 2700
      F = 1. + SNPHI1*SNPHIR - CSPHI1*CSPHIR*CSTHETAR                   SSS 2800
      F = F/(SNPHI1*(SNPHI1 + SNPHIR))                                  SSS 2900
      VSUBX = (CSPHI1 - CSPHIR*CSTHETAR)*WAVNUM
      VSUBY = -CSPHIR*SNTHETAR*WAVNUM
      VSUBZ = WAVNUM*(SNPHI1 + SNPHIR)                                  SSS 3000
      IF(VSUBX.EQ.0.0) RHO1 = 1.
      IF(VSUBX.EQ.0.0) GO TO 8
      RHO1 = SIN(50.*VSUBX)/(50.*VSUBX)
    8 IF(VSUBY.EQ.0.0) RHO2 = 1.0
      IF(VSUBY.EQ.0.0) GO TO 9
      RHO2 = SIN(50.*VSUBY)/(50.*VSUBY)
    9 RHO = RHO1*RHO2
C                                                                       SSS 3200
C  ROUGHNESS CRITERION IS DETERMINED                                    SSS 3300
C  THIS IS ONLY FOR A ROUGH AND A MODERATELY ROUGH SEA SURFACE          SSS 3400
C                                                                       SSS 3500
      G = VSUBZ*VSUBZ*SIGMA*SIGMA                                       SSS 3600
C                                                                       SSS 3700
C  G MUCH GREATER THAN ONE CORRESPONDS TO A ROUGH SURFACE WHILE G       SSS 3800
C  APPROXIMATELY EQUAL TO ONE CORRESPONDS TO A MODERATELY ROUGH         SSS 3900
C  SURFACE.                                                             SSS 4000
C                                                                       SSS 4100
      IF(G.LT.4.) GO TO 1                                               SSS 4200
      POWER = -1.*(VSUBXYSQ*T*T/(4.*G))                                 SSS 4300
      RHORHOST = PI*F*F*T*T*EXPF(POWER)/(G*AREA)                        SSS 4400
      GO TO 5                                                           SSS 4500
    1 J = 1                                                             SSS 4600
      A = PI*T*T*F*F*EXPF(-G)                                           SSS 4700
      FACTOR = 1/J                                                      SSS 4800
```

```
 2 DENOM = FACTOR/J                                                     SSS 4900
   POWER = -1.*(VSUBXYSQ*T*T/(4.*J))                                    SSS 5000
   TERM(J) = G**J*EXPF(POWER)*DENOM                                     SSS 5100
   IF(TERM(J).LE..0000001) GO TO 3                                      SSS 5200
   J = J + 1                                                            SSS 5300
   FACTOR = FACTOR/J                                                    SSS 5400
   IF(J - 100) 2, 2, 6                                                  SSS 5500
 3 SUM = 0                                                              SSS 5600
   DO 4 N=1, J                                                          SSS 5700
   SUM = SUM + TERM(N)                                                  SSS 5800
 4 CONTINUE                                                             SSS 5900
   RHORHOST = A*SUM/AREA + EXPF(-G)*RHO**2
 5 PRC = RHORHOST*SNPHI1**2*AREA/WAVLEN**2
   SSUBS = 10.*ALOG10(PRC)
   RETURN                                                               SSS 6300
 6 PRINT 7, TERM(100)                                                   SSS 6400
 7 FORMAT(* SERIES FOR SSUBS FAILED TO CONVERGE RAPIDLY ENOUGH  *,      SSS 6500
  1E11.4)                                                               SSS 6600
   STOP                                                                 SSS 6700
   END                                                                  SSS 6800
```

```
      SUBROUTINE DATA(RCRANGE,RANGE1,THETAR,INDEX1,RAN)
C
C  DATA COMPUTES THE VALUES OF RCRANGE(I), THETAR(I), RANGE1(I), AND
C  N, WHERE N IS THE TOTAL NUMBER OF SCATTERING POINTS TO BE CONSIDERED.
C
      DIMENSION RCRANGE(1),THETAR(1),T(4,8),RANGE1(1)
      I = 1
      DIST = 50
      RANO = RAN - DIST
      INDEX = 1
      INDEX1 = 0
      INDEX2 = 0
    2 J = 1
      RAN3 = RAN - RANO
      TRAN = 0
C
C   THIS COMPUTES RANGE1 USED BY ROUTINE THAT COMPUTES H (PROPAGATION
C   LOSS OF ANY ONE RAY).
C
    1 T(4,J) = SQRT(TRAN*TRAN + RANO*RANO)
      INDEX2 = INDEX2 + 1
      RANGE1(INDEX2) = T(4,J)
      J = J + 1
      TRAN = TRAN + DIST
      IF (TRAN.LE.350) GO TO 1
      J = 1
      TRAN = 0
    3 T(1,J) = SQRT(TRAN*TRAN + RAN3*RAN3)
C
C  THIS PUTS RANGES IN PROPER ARRAY TO BE USED BY SURFSCAT
C
      RCRANGE(INDEX) = T(1,J)
      J = J + 1
      TRAN = TRAN + DIST
      INDEX = INDEX + 1
      IF(TRAN.LE.350) GO TO 3
      J = 1
      TRAN = 0
    4 T(2,J) = TRAN
      TRAN = TRAN  + DIST
      J = J + 1
      IF(TRAN.LE.350) GO TO 4
      DO 6 J=1,8
      THET1 = ATAN(T(2,J)/RANO)
      THET2 = ATAN(T(2,J)/RAN3)
      T(3,J) = 57.2958*(THET1 + THET2)
C
C  THIS PUTS THETAR IN PROPER ARRAY TO BE USED BY SURFSCAT
C
      INDEX1 = INDEX1 + 1
    6 THETAR(INDEX1) = T(3,J)
      I = I + 1
      IF(I-9)9,10,10
    9 RANO = RANO - DIST
      IF(1000 - RANO)2,2,112
   10 I = 1
      GO TO 9
  112 CONTINUE
      END
```

```
      SUBROUTINE CMPUTE H(RAN, HC, ZSUB0, WAVE, TEMP, SUMH, MX, H, GC)  H    0100
C                                                                       H    0200
C  COMPUTE H COMPUTES THE SCATTERING LOSS FROM THE SOURCE TO A          H    0300
C  SCATTERING POINT (TL1), H AND SUMS THE EQUIVALENT INTENSITIES,       H    0400
C  10**(H/10).                                                          H    0500
C                                                                       H    0600
      COMMON/SSSDATA/DUM1,DUM2,TL2,DUM3,DM,FREQ,DUM4,DUM5,C0,SSUBS      H    0700
      NR   = RAN*SQRTF(GC)/SQRTF(8.*HC*C0)                              H    0800
      QUAN = NR                                                         H    0900
      IF(QUAN.LT.1.) QUAN = 1.                                          H    1000
    5 R = RAN/(1000.*SQRTF(3.*HC))                                      H    1100
      RAN = RAN/1000.                                                   H    1200
      ZSUB0 = SQRTF(ZSUB0/HC)                                           H    1300
      ZEE = .1                                                          H    1400
      ASUBS = 3.26*SQRTF(WAVE*FREQ)/SQRTF(3.*HC)                        H    1500
      EXP = 6. - (2100./(TEMP + 459.6))                                 H    1600
      FSUBT = 1.23*10.**EXP                                             H    1700
      ALPHA = .651*FSUBT*FREQ**2/(FREQ**2 + FSUBT**2) + .0269*FREQ**2/FSH    1800
     *UBT                                                               H    1900
      G0FZ = .1*10.**(2.3*(ZEE - ZSUB0))*(FREQ/25.)**(1./3.)            H    2000
      F0FZEE = .4*10.**ZEE                                              H    2100
      F0FZSUB0 = .4*10.**ZSUB0                                          H    2200
      F0FZED = .4*10.**(ZEE - ZSUB0)                                    H    2300
      FACT0R = (FREQ/8.)**(1./3.)                                       H    2400
      IF(FREQ.LE.8.) FACT0R = 1.                                        H    2500
      H0FZZ0 = (F0FZEE + F0FZSUB0 + F0FZED)*FACT0R                      H    2600
      RSUB1 = (2. - ZEE - ZSUB0)/4.                                     H    2700
      IF(RAN.GE.RSUB1) GO TO 1                                          H    2800
      TL1 = 20.*ALOG10(RAN) + ALPHA*RAN + G0FZ*R/RSUB1 + 60.            H    2900
      GO TO 3                                                           H    3000
    1 IF(R.GE.RSUB1 + .5) GO TO 2                                       H    3100
      TL1 = 20.*ALOG10(RAN) + ALPHA*RAN + 2.*(R - RSUB1)*H0FZZ0 + (1. - H    3200
     $2.*(R - RSUB1))*G0FZ  + 60.                                       H    3300
      GO TO 3                                                           H    3400
    2 ARG1 = (RSUB1 + .5)*SQRTF(3.*HC)                                  H    3500
      TL1 = 10.*ALOG10(RAN) + RAN*(ALPHA + ASUBS) + H0FZZ0 - ASUBS*ARG1 H    3600
     1+ 10.*ALOG10(ARG1) + 60.                                          H    3700
    3 H = -TL1 - TL2 + SSUBS + 33.98 - QUAN*1.64*SQRTF(WAVE*FREQ*3.)    H    3800
C                                                                       H    3900
C  H IS INCREMENTED BY 10*ALOG10(2) FOR SCATTERING POINTS OFF THE       H    4000
C  CENTER LINE TO ACCOUNT FOR THE PROPAGATION LOSS FROM THE SYMMETRIC   H    4100
C  SQUARE ON THE OPPOSITE SIDE OF THE CENTER LINE.                      H    4200
C                                                                       H    4300
      IF(MX.NE.1) H = H + 10*ALOG10(2.)                                 H    4400
      SUMH = SUMH + 10**(H/10)                                          H    4500
      RAN = 1000.*RAN                                                   H    4600
      END                                                               H    4700
```

```
      SUBROUTINE TABLE(MX)                                              TAB 0100
C                                                                       TAB 0200
C  TABLE PRINTS OUT VALUES OF H, 10LOG(SSUBS), AND ANGRCVR IN           TAB 0300
C  TABULAR FORM.                                                        TAB 0400
C                                                                       TAB 0500
      COMMON/SSS DATA/ANGSRCE,ANGRCVR,PLOSS,RR,RDEPTH,FREQ,NOSOL,FLAG,  TAB 0600
     1CO,SSUBS                                                          TAB 0700
      COMMON/A/H                                                        TAB 0800
      DIMENSION ARRAYH(8),ARRAYS(8),ARRAYA(8)                           TAB 0900
      DATA(I = 100), (LINE = 21)
      DATA( N = 0 )                                                     TAB 1100
      IF( I - 100 )5 , 7                                                TAB 1200
    7 PRINT 30                                                          TAB 1300
  300FORMAT( 40X , 1HM //            19X , 1H0 , 9X , 1H1 , 9X , 1H2 ,  TAB 1400
     19X , 1H3 , 9X , 1H4 , 9X , 1H5 , 9X , 1H6 , 9X , 1H7 / 5X , 1HN /)TAB 1500
      I = I - 100                                                       TAB 1600
    5 I = I + 1                                                         TAB 1700
      K = 1
      ARRAYH(I) = h                                                     TAB 1800
      ARRAYS(I) = SSUBS                                                 TAB 1900
      ARRAYA(I) = ANGRCVR                                               TAB 2000
      IF (I.EQ.MX) GO TO 20                                             TAB 2100
   10 RETURN                                                            TAB 2200
   20 I = 0                                                             TAB 2300
      N = N + 1                                                         TAB 2400
      IF(MX.EQ.8.AND.ARRAYS(8).GE.-60.) K = 0
      M = MX - K
      PRINT 40, (ARRAYH(J), J = 1, M)
      PRINT 41, N, (ARRAYS(J), J = 1, M)
      PRINT 42, (ARRAYA(J), J = 1, M)
      LINE = LINE + 4
      IF(LINE.GE.53) PRINT 44
      IF(LINE.GE.53) LINE = 0
   40 FORMAT(/, 13X, 3HH   , 8(F8.2,2X))                                TAB 2900
   41 FORMAT (1X,I5,3X,7HSSUBS   ,8(F8.2,2X))                           TAB 3000
   42 FORMAT(7X, 9HANGRCVR   , 8(F8.2, 2X))                             TAB 3100
   44 FORMAT(1H1)
      END                                                               TAB 3200
```

```
      SUBROUTINE AMOS                                                   AMO 0100
C                                                                       AMO 0200
C  AMOS COMPUTES ANOTHER VALUE FOR HT USING THE AMOS FORMULATIONS.      AMO 0300
C                                                                       AMO 0400
      COMMON/ANDY/H, ZSUBO, ZEE, RAN, FREQ, WAVE, TEMP                  AMO 0500
      REAL LOSS1, LOSS2                                                 AMO 0600
      FREQ =  1000.*FREQ                                                AMO 0700
      ALPHAS = 3.28*SQRTF(WAVE*FREQ/(3000.*H))                          AMO 0800
      R = RAN/(SQRTF(3.*H)*1000.)                                       AMO 0900
      ZZOO = ZSUBO                                                      AMO 1000
      ZSUBO = SQRTF(ZSUBO/H)                                            AMO 1100
      ZENO = ZEE                                                        AMO 1200
      ZEE = SQRTF(ZEE/H)                                                AMO 1300
      ZED = ZEE - ZSUBO                                                 AMO 1400
      FREQO = (FREQ/25000.)**(1./3.)                                    AMO 1500
      IF(ZED.GT.1.) GO TO 2                                             AMO 1600
      GOFZED = (.1*10.**(2.3*ZED))*FREQO                                AMO 1700
      GO TO 3                                                           AMO 1800
    2 GOFZED = 20.*FREQO                                                AMO 1900
    3 FOFZED = .4*10.**ZED                                              AMO 2000
      FOFZEE = .4*10.**ZEE                                              AMO 2100
      FOFZO = .4*10.**ZSUBO                                             AMO 2200
      HOFZ = FOFZED + FOFZEE + FOFZO                                    AMO 2300
      IF(FREQ.LT.8000.) GO TO 4                                         AMO 2400
      HOFZ = HOFZ*(FREQ/8000.)**(1./3.)                                 AMO 2500
    4 POWER = 6. - (2100./(TEMP + 459.6))                               AMO 2600
      FSUBT =  1.23*(10.**POWER)                                        AMO 2700
      A = (.651*((FREQ/1000.)**2)*FSUBT/((FREQ/1000.)**2 + FSUBT**2)) + AMO 2800
     *.0269*(FREQ/1000.)**2/FSUBT                                       AMO 2900
      IF(ZSUBO.GT.1.) GO TO 6                                           AMO 3000
      IF(ZEE.GT.1.) GO TO 5                                             AMO 3100
      RSUB1 = (2. - ZEE - ZSUBO)/4.                                     AMO 3200
      GO TO 7                                                           AMO 3300
    5 RSUB1 = (1. - ZSUBO)/4. + SQRTF(ZEE**2 - 1.)/5.                   AMO 3400
      GO TO 7                                                           AMO 3500
    6 RSUB1 = (SQRTF(ZSUBO**2 - 1.) + SQRTF(ZEE**2 - 1.))/5.            AMO 3600
    7 VAL = ABSF(3.*ZENO - 3.*H)                                        AMO 3700
      QUA = ABSF(3.*ZZOO - 3.*H)                                        AMO 3800
      TEST = 25. - SQRTF(VAL) - SQRTF(QUA) + 5.*RAN/1000.               AMO 3900
      IF(TEST.LE.0.) TEST = 0.0                                         AMO 4000
      LOSS2 = 20.*ALOG10(RAN) + A*(RAN/1000.) + TEST*FREQO              AMO 4100
      IF(R.GT.RSUB1) GO TO 10                                           AMO 4200
      LOSS1 = 20.*ALOG10(RAN) + A*(RAN/1000.) + GOFZED*R/RSUB1          AMO 4300
      IF(ZEE.LE.1.AND.ZSUBO.LE.1.) GO TO 8                              AMO 4400
      IF(LOSS1 - LOSS2)   8, 9, 9                                       AMO 4500
    8 PLOSS = LOSS1                                                     AMO 4600
      GO  TO 16                                                         AMO 4700
    9 PLOSS  = LOSS2                                                    AMO 4800
      GO TO 16                                                          AMO 4900
   10 IF(R.GE.RSUB1+.5) GO TO 14                                        AMO 5000
      PLOSS = 20.*ALOG10(RAN) + A*RAN/1000. + 2.*(R - RSUB1)*HOFZ +     AMO 5100
     *(1. - 2.*(R - RSUB1))*GOFZED                                      AMO 5200
      IF(ZSUBO.LE.1.) GO TO 16                                          AMO 5300
   11 IF(PLOSS - LOSS2) 12, 13, 13                                      AMO 5400
   12 GO TO 14                                                          AMO 5500
   13 PLOSS = LOSS2                                                     AMO 5600
      GO TO 16                                                          AMO 5700
```

```
14 PLOSS = 10.*ALOG10(RAN) + (A + ALPHAS)*RAN/1000. + HOF7 - ALPHAS   AMO 5800
  **SQRTF(3.*H)*(RSUB1 + .5) + 10.*ALOG10(SQRTF(3.*H)*(RSUB1 + .5)) *AMO 5900
  *30.                                                               AMO 6000
   IF(ZSUBR.GT.1.) GO TO 15                                          AMO 6100
   GO TO 16                                                          AMO 6200
15 IF(PLOSS - LOSS2) 16, 17, 17                                      AMO 6300
17 PLOSS = LOSS2                                                     AMO 6400
16 PRINT 90, PLOSS                                                   AMO 6500
90 FORMAT(5X*AMOS FORMULATIONS GIVE *,F5.1,* DB LOSS*)               AMO 6600
   RETURN                                                            AMO 6700
   END                                                               AMO 6800
```

```
ENTRY:INITPTOP,PTOP ^                                                   PTP00100
LIBRARY≠        RDPROFLE^                                               PTP00200
(:THIS COMMON BLOCK GIVES THESE TWO VALUES TO SUBROUTINE CMPUTE H)      PTP00300
COMMON:SNDSPDS(2):SURFACE VELOCITY,BOTTOM VELOCITY.^                    PTP00400
COMMON;SSSDATA(8);ANGSRCE.ANGRCVR.PLOSS,RR,RDEPTH,FREQ.NOSOL.FLAG.^     PTP00500
COMMON≠PROFILE(4012)≠                                                   PTP00600
                  MONTH,DAY,YEAR,LATITUDE.LONGITUDE.IDNO,               PTP00700
   ORIGINATOR(2),                                                       PTP00800
NROFDEPTHS,   BT,DONC,MTHD,                                             PTP00900
DEPTH TABLE≠DT(800), C≠SOUNDSPEED TABLE(800). ASUBONE≠A1TABLE (800),    PTP01000
ASUBTWO≠A2TABLE(800), ASUBTHREE≠A3TABLE(800).^..                        PTP01100
TX DEPTH≠ Z SUBTX = 000000.00000,                                       PTP01200
RECEIVER DEPTH≠ RCVR DEPTH≠ ZSUBRC = 000000.00000,                      PTP01300
RECEIVER RANGE = 000000.0000,                                           PTP01400
MODE,ANG,                                                               PTP01500
NR OF A1S≠ NMBR OF A1S,                                                 PTP01600
(≠PAGE F5273-2)                                                         PTP01700
C OF SOURCE≠ TX SOUND SPEED = 0000.0000,                                PTP01800
MODE DIRECTION,                                                         PTP01900
MODE NUMBER = 00,                                                       PTP02000
RANGE≠ R(2)=0*0,                                                        PTP02100
DERIVATIVE≠ D(2)=0*0,                                                   PTP02200
RANGE SIGN ≠ RS (2),                                                    PTP02300
DERIVATIVE SIGN≠ DS(2),                                                 PTP02400
EPSILON,                                                                PTP02500
DELTAV=0000.0000,                                                       PTP02600
CASE NUMBER≠ CN,                                                        PTP02700
VRR (6),                                                                PTP02800
VELOCITY≠ V(2)=0*0,                                                     PTP02900
                  V MAX≠ MAXIMUM VELOCITY, INIVL.                       PTP03000
INITIAL VELOCITY=0000.0000,                                             PTP03100
MAXVONPROFILE=0000.0000, MAXPROFILEDEPTH=0000.0000, 11,                 PTP03200
CMXZTAB (100),                                                          PTP03300
CMT≠ C MAX TABLE (100).^                                                PTP03400
MAIN PROGRAM DIMENSIONING≠..                                            PTP03500
INTERVAL SIZE=4.0 ,                                                     PTP03600
ATTN FACTOR, FSQ,^                                                      PTP03700
INITPTOP:=1.0|TXDEPTH, 1.0*37|RM,                                       PTP03800
        FLAG = 0 : RDPROFLE ^ ^                                         PTP03900
97421.2034|VMAX, 1.0*-5|EPSILON, PROFILE LABEL,                         PTP04000
COMPUTE ATTN FACTOR, NR OF DEPTHS -1 | NR OF A1S,                       PTP04100
RDEPTH|RCVRDEPTH,                                                       PTP04200
SEARCH, NROFA1S|BOTTOM INDEX, C[BOTTOM INDEX]|BOTTOM VELOCITY,          PTP04300
(MAXVONPROFILE-INITIALVELOCITY)/INTERVALSIZE|DELTAV,                    PTP04400
ADPROINFO,                                                              PTP04500
C[TBI]|TXSNDSPD|TXSOUNDSPEED,  C[RRI]|RCSNDSPD,                         PTP04600
1|MODE,                                                                 PTP04700
PTOP:=RR|RECEIVERRANGE,NEWMODE,EXITPTOP:NEXTTRC:NEXTMODE:,              PTP04800
END OF RUN≠..                                                           PTP04900
LSTVINCFLAG^                                                            PTP05000
NEW MODE≠                                                               PTP05100
→INITIAL VELOCITY + EPSILON | V + DELTA V | V[1], 1 | NOSOL,            PTP05200
0|LSTVINCFLAG,                                                          PTP05300
MODE < 0≠                                                               PTP05400
   - 1 | MODE DIRECTION, $ MODE % | MODE NUMBER^ (≠ UP)                 PTP05500
      1 | MODE DIRECTION, MODE | MODE NUMBER^        (≠ DOWN)           PTP05600
TRACE RAY (V, 0^ RANGE, DERIVATIVE, RANGE SIGN, DERIVATIVE SIGN),       PTP05700
```

```
DETERMINE CASE≠                                                         PTP05800
TRACE RAY (V[1], 0⍉ RANGE[1], DERIVATIVE [1], RANGE SIGN [1], DERIVATIVPTP05900
E SIGN[1]),                                                             PTP06000
RAY TRACE MATRIX (RS, RS[1], DS, DS[1]⍉ CASE NUMBER),                   PTP06100
CASE SWITCH [CN],                                                       PTP06200
CASE SWITCH≠                                                            PTP06300
CASE 0,                                                                 PTP06400
CASE 1,                                                                 PTP06500
CASE 20,                                                                PTP06600
CASE 31,                                                                PTP06700
CASE 0≠ NEXT INCREMENT,                                                 PTP06800
(≠PAGE F5273-5)                                                         PTP06900
CASE 1≠                                                                 PTP07000
CASE 1 FUNCTION (V, V[1], R, R[1], RS, RS[1]⍉ VRR, VRR[1]),             PTP07100
TRACE SOLUTION RAYS (VRR, VRR[1]⍉), NEXT INCREMENT.                     PTP07200
CASE 20≠                                                                PTP07300
CASE 20 FUNCTION (V, V[1], R, R[1], RS, RS[1], DS, DS[1]⍉               PTP07400
                 VRR, VRR[1], VRR[2], VRR[3]),                          PTP07500
VRR = 0≠                                                                PTP07600
   NEXT INCREMENT.⍉                                                     PTP07700
TRACE SOLUTION RAYS (VRR, VRR[1]⍉),                                     PTP07800
TRACE SOLUTION RAYS (VRR[2], VRR[3]⍉), NEXT INCREMENT,                  PTP07900
CASE 31≠                                                                PTP08000
CASE 31 FUNCTION (V, V[1], R, R[1], RS, RS[1], DS, DS[1]⍉               PTP08100
      VRR, VRR[1], VRR[2], VRR[3], VRR[4], VRR[5]),                     PTP08200
TRACE SOLUTION RAYS (VRR, VRR[1]⍉),                                     PTP08300
VRR [2] = 0≠                                                            PTP08400
   NEXT INCREMENT.⍉                                                     PTP08500
TRACE SOLUTION RAYS (VRR[2], VRR[3]⍉),                                  PTP08600
TRACE SOLUTION RAYS (VRR[4], VRR[5]⍉), NEXT INCREMENT,                  PTP08700
(≠PAGE F5273-6)                                                         PTP08800
NEXT INCREMENT≠                                                         PTP08900
V[1] | V + DELTA V | V [1], RANGE [1] | RANGE,                          PTP09000
DERIVATIVE[1] | DERIVATIVE, RANGE SIGN [1] | RANGE SIGN,                PTP09100
DERIVATIVE SIGN [1] | DERIVATIVE SIGN,                                  PTP09200
V[1] < MAX V ON PROFILE≠                                                PTP09300
   DETERMINE CASE.⍉                                                     PTP09400
LSTVINCFLAG=0≠                                                          PTP09500
   1|LSTVINCFLAG, MAXVONPROFILE|V[1],                                   PTP09600
   DETERMINE CASE.⍉                                                     PTP09700
LAST VINCREMENT≠                                                        PTP09800
TRACE RAY (VMAX, 0⍉ RANGE[1],, RANGE SIGN[1]),                          PTP09900
RANGE SIGN [1] = RANGE SIGN≠                                            PTP10000
   END OF MODE.⍉                                                        PTP10100
CASE 1 FUNCTION (V, VMAX, R, R[1], RS, RS[1]⍉ VRR, VRR[1]),             PTP10200
TRACE SOLUTION RAYS (VRR, VRR[1]⍉),                                     PTP10300
END OF MODE≠⍟,,                                                         PTP10400
AV=0000.000,⍉                                                           PTP10500
TRACE SOLUTION RAYS (V (2), R (2), TT (2), SL (2),                      PTP10600
ATTN (2), PROP (2), ANGLE. DV (2) = 0*0, TSTEMP,)≠                      PTP10700
→TRACE RAY(V,1⍉R),TRACE RAY(V[1],1⍉R[1]),                               PTP10800
(RECEIVER RANGE - R) / (R [1] - R) | TSTEMP,                            PTP10900
INTERP (TS TEMP, V, V [1]⍉ V),                                          PTP11000
TRACE RAY (V, 1⍉ R, DV,,, TT, SL, ATTN),                                PTP11100
FIND ANGLE (V⍉ ANGLE), ATTN + SL | PROP,                                PTP11200
0 | NOSOL,                                                              PTP11300
ANGLE|ANGSRCE,PROP|PLOSS,EXITPTOP.⍟                                     PTP11400
```

```
FIND ANGLE (V, ANG.)≠                                                   PTP11500
→TX SOUND SPEED / V | ANG, ARCCOS (ANG↑ ANG),                           PTP11600
ANG * 57.2957795 | ANG.                                                 PTP11700
INTERP (RATIO, X (2), VAL)≠ →RATIO * (X [1] - X) * X | VAL. ..          PTP11800
DD=00,MM=00,YY=0000,LONG=000.00,LAT=000.00,ID=[1],BTNO=000,             PTP11900
ORIG(2) = [≠AAAAAA],                                                    PTP12000
DONC BUFF=000000,                                                       PTP12100
MTHD BUFF=00,                                                           PTP12200
FREQUENCY = 00000.000,↑                                                 PTP12300
PROFILE LABEL≠→                                                         PTP12400
DAY(DD,MONTH(MM,YEAR(YY,LONGITUDE(LONG,LATITUDE(LAT,IDNO(ID,            PTP12500
BT(BTNO, ORIGINATOR(ORIG,                                               PTP12600
ORIGINATOR[1] | ORIG[1],                                                PTP12700
DONC(DONCBUFF, MTHD | MTHDBUFF,                                         PTP12800
 ,→PRINTER<>↑ << POINTS TO% POINTS RAYS TRACING >>.,                    PTP12900
<< PROFILES INFORMATION >>,                                             PTP13000
<<?DATE>$MM(DD(YY>                                                      PTP13100
<<?LATITUDE>$LAT>                                                       PTP13200
<<?LONGITUDE>$LONG>                                                     PTP13300
<<?ID$NO,>ID>                                                           PTP13400
<<ORIGINATOR>$ORIG(ORIG[1]>..                                           PTP13500
ADPROINFO≠→                                                             PTP13600
,→<<MAX$V$ON$PROFILE$$>MAXVONPROFILE>                                   PTP13700
<<DEPTH$OF$MAX$V$ON$PROFILE$$>MAXPROFILEDEPTH>                          PTP13800
<<INITIAL$VELOCITY$$>INITIALVELOCITY>                                   PTP13900
<<DELTA$VELOCITY$$>DELTAV>                                              PTP14000
<<INTERVAL$SIZE$$>INTERVALSIZE>..                                       PTP14100
COMPUTE ATTN FACTOR≠                                                    PTP14200
→FREQ * FREQ | FSQ * 0.04 / (4100.0 + FSQ) + 2.75 * -7 * FSQ            PTP14300
| ATTN FACTOR..                                                         PTP14400
7                                                                       PTP14500
(≠PAGE F5494-1)                                                         PTP14600
RANGE MAXIMUM≠ RM,                                                      PTP14700
PATH LENGTH,                                                            PTP14800
MODE COUNT,                                                             PTP14900
COS THETA SUBO, TAN THETA SUBO,                                         PTP15000
TEMP (5),                                                               PTP15100
UP OR DOWN, HI, LO,                                                     PTP15200
LBI, UBI,              (≠LOWER AND UPPER BOUNDARY INDICES)              PTP15300
VERTEX FLAG,                                                            PTP15400
ZR FLAG,                                                                PTP15500
BOUNCE FLAG,                                                            PTP15600
ZR1STINC FLAG,                                                          PTP15700
APEX, NADIR,                                                            PTP15800
ABI, NBI,        (≠APEX AND NADIR BOUNDARY INDICES)                     PTP15900
BOTTOM INDEX, SURFACE INDEX,                                            PTP16000
HALF CYCLE FLAG,                                                        PTP16100
RANGE TABLE (100),                                                      PTP16200
TRAVEL TIME TABLE (100),                                                PTP16300
DERIVATIVE TABLE (100),                                                 PTP16400
PATH LENGTH TABLE (100).↑                                               PTP16500
(≠PAGE F5494-2)                                                         PTP16600
TRACE RAY (V, SOLUTION FLAG, RANGE, DERIVATIVE,                         PTP16700
RANGE SIGN, DERIVATIVE SIGN, TRAVELTIME, SPRD LOSS, ATTENUATION.)≠      PTP16800
→0 | RANGE | DERIVATIVE | MODE COUNT | HALF CYCLE FLAG | APEX | NADIR |PTP16900
 NBI                                                                    PTP17000
| VERTEX FLAG | BOUNCE FLAG | ZRFLAG | ZR1STINC FLAG,                   PTP17100
```

```
| TRAVEL TIME | SURFACE INDEX | PATH LENGTH, -1 | ABI,                          PTP17200
V = TX SOUND SPEED≠                                                               PTP17300
FIND DIRECTION≠                                                                   PTP17400
   V > CMT [TBI - 1]≠                                                             PTP17500
      V < CMT [TBI + 1]≠                                                          PTP17600
         - 1 | MODE DIRECTION^ ERROR INC.^ (≠NADIR)                               PTP17700
      V > CMT [TBI+1]≠                                                            PTP17800
         1 | MODE DIRECTION^ ERROR INC.^ (≠APEX)                                  PTP17900
   1 | ZR FLAG | ZR1STINC FLAG, 1.0 | COSTHETA SUB0,                              PTP18000
   0 | TANTHETA SUB0, CONT.^                                                      PTP18100
C OF SOURCE / V | COS THETA SUB0, 1.0 / COS THETA SUB0 | TEMP *                   PTP18200
TEMP - 1.0 | TEMP [1], SQRT (TEMP [1]^ TAN THETA SUB0),                            PTP18300
CONT≠                                                                             PTP18400
MODE DIRECTION | UP OR DOWN = 1≠       (≠I. E. DOWN)                              PTP18500
TBI = BOTTOM INDEX≠                                                               PTP18600
   -UP OR DOWN | UP OR DOWN, UP, DOWN.^                                           PTP18700
TBI = 0≠                                                                          PTP18800
   -UP OR DOWN | UP OR DOWN, DOWN. UP.^                                           PTP18900
   DOWN≠ TBI | LBI | L0 | HI + 1 | UBI, CONT1,                                    PTP19000
   UP≠ TBI | UBI | L0 | HI -1 | LBI,                                              PTP19100
CONT1≠                                                                            PTP19200
ZR1STINC FLAG ( 0≠ - UP OR DOWN | UP OR DOWN^^                                    PTP19300
(≠PAGE F5494-3)                                                                   PTP19400
NEXT LAYER≠                                                                       PTP19500
UP OR DOWN = 1≠                                                                   PTP19600
   LBI = RBI≠                                                                     PTP19700
      INCREASE MODE COUNT.^^                                                      PTP19800
   UBI = RBI≠                                                                     PTP19900
      INCREASE MODE COUNT.^^                                                      PTP20000
CONTINUE TRACING,                                                                 PTP20100
INCREASE MODE COUNT≠                                                              PTP20200
MODE COUNT + 1 | MODE COUNT ≥ MODE NUMBER≠                                        PTP20300
   SET SIGNS.^                                                                    PTP20400
CONTINUE TRACING≠                                                                 PTP20500
HALF CYCLE FLAG ( 0≠                                                              PTP20600
   BRIC,^                                                                         PTP20700
LBI ≥ L0 | UBI ≤ HI≠                                                              PTP20800
   BRIC,^                                                                         PTP20900
SOLUTION FLAG = 1≠                                                                PTP21000
   COMPUTE RANGE (V, LBI, SOLUTION FLAG^                                          PTP21100
   RANGE TABLE [LBI], DERIVATIVE TABLE [LBI],                                     PTP21200
   TRAVEL TIME TABLE [LBI], PATH LENGTH TABLE [LBI], VERTEX FLAG)^                PTP21300
   COMPUTE RANGE (V, LBI, 0^ RANGE TABLE [LBI],                                   PTP21400
   DERIVATIVE TABLE [LBI],,, VERTEX FLAG)^                                        PTP21500
(≠PAGE F5494-4)                                                                   PTP21600
ZR1ST INC FLAG ( 0≠                                                               PTP21700
   0 | ZR1STINC FLAG,                                                             PTP21800
   MODE DIRECTION = 1≠                                                            PTP21900
      HI + 1 | HI | L0^                                                           PTP22000
      HI - 1 | HI | L0^                                                           PTP22100
      BRIC,^                                                                      PTP22200
   UP OR DOWN = 1≠                                                                PTP22300
      HI + 1 | HI^                                                                PTP22400
      L0 - 1 | L0^                                                                PTP22500
VERTEX FLAG = 0≠                                                                  PTP22600
      BRIC,^                                                                      PTP22700
UP OR DOWN = 1≠          (≠VERTEXED)                                              PTP22800
```

```
      1 | NADIR, LBI | NBI ≤ RBI≠                                          PTP22900
SOLUTION FLAG(0≠ RCVR OUTSIDE.⌐                                            PTP23000
          NEXT INCREMENT.⌐⌐                                                PTP23100
      1 | APEX, LBI | ABI > RBI≠                                           PTP23200
SOLUTION FLAG(0≠ RCVR OUTSIDE.⌐                                            PTP23300
          NEXT INCREMENT.⌐⌐                                                PTP23400
CONT VERTEX≠                                                               PTP23500
APEX = NADIR ? BOUNCE FLAG = 1≠                                            PTP23600
        1 | HALF CYCLE FLAG⌐⌐                                              PTP23700
0 | VERTEX FLAG,                                                           PTP23800
BYPASS RANGE INCREMENT COMPUTATION≠ BRIC≠                                  PTP23900
RANGE + RANGE TABLE [LBI] | RANGE > RANGE MAXIMUM≠                         PTP24000
   NEXT INCREMENT.⌐                                                        PTP24100
ZR FLAG ( 0≠                                                               PTP24200
   ATT.⌐                                                                   PTP24300
SOLUTION FLAG (0≠                                                          PTP24400
   PATH LENGTH + PATH LENGTH TABLE [LBI] | PATH LENGTH⌐⌐                   PTP24500
DERIVATIVE TABLE [LBI] + DERIVATIVE | DERIVATIVE,                          PTP24600
ATT≠ TRAVEL TIME + TRAVEL TIME TABLE [LBI] | TRAVEL TIME,                  PTP24700
(≠PAGE F 5494-5)                                                           PTP24800
UBI = NBI ? LBI = ABI≠                                                     PTP24900
   MCAVTX, (≠RAY HAS VERTEXED)                                             PTP25000
UBI ≥ BOTTOM INDEX≠                                                        PTP25100
   APEX = 1≠                                                               PTP25200
      1 | HALF CYCLEFLAG⌐⌐ 1 | BOUNCE FLAG, RHB,                           PTP25300
LBI ≤ SURFACE INDEX≠                                                       PTP25400
   NADIR = 1≠                                                              PTP25500
      1 | HALF CYCLE FLAG⌐⌐ 1 | BOUNCE FLAG, RHB. NBORV.⌐                   PTP25600
RHB≠                                                                       PTP25700
      (≠RAY HAS BOUNCED)                                                   PTP25800
RANGE TABLE [LBI] + RANGE | RANGE,                                         PTP25900
DERIVATIVE TABLE [LBI] + DERIVATIVE | DERIVATIVE,                          PTP26000
SOLUTION FLAG ( 0≠                                                         PTP26100
   TRAVEL TIME + TRAVEL TIME TABLE [LBI] | TRAVEL TIME,                    PTP26200
   PATH LENGTH + PATH LENGTH TABLE [LBI] | PATH LENGTH⌐⌐                   PTP26300
(≠MODE COUNT MAY NEED TO BE INCREASED IF RAY HAS VERTEXED OR BOUNCED ATPTP26400
 THE RECEIVER DEPTH NECESSITATING THIS DOUBLE CHECK WHICH IS DONE         PTP26500
 AT THE ENTRY POINT  **NEXT LAYER**)                                       PTP26600
 MCAVTX≠MODE CHECK AT VERTEX≠                                              PTP26700
 →UPORDOWN|UPORDOWN=1≠                                                     PTP26800
    LBI(RBI≠NBORV.⌐⌐                                                       PTP26900
    UBI(RBI≠NBORV.⌐⌐                                                       PTP27000
MODE COUNT+1|MODE COUNT≥MODE NUMBER≠                                       PTP27100
   SET SIGNS.⌐                                                             PTP27200
NO BOUNCE OR VERTEX≠ NHORV≠                                                PTP27300
(≠PAGE F5494-6)                                                            PTP27400
UP OR DOWN = 1≠                                                            PTP27500
   UBI | LBI + 1 | UBI⌐                                                    PTP27600
   LBI | UBI - 1 | LBI⌐                                                    PTP27700
NEXT LAYER,                                                                PTP27800
SET SIGNS≠                                                                 PTP27900
ZR FLAG ( 0≠                                                               PTP28000
   0 | ZRFLAG,                                                             PTP28100
   TRACE RAY EXIT.⌐                                                        PTP28200
RANGE ≥ RECEIVER RANGE≠                                                    PTP28300
   1 | RANGE SIGN⌐                                                         PTP28400
   -1 | RANGE SIGN⌐                                                        PTP28500
```

```
DERIVATIVE ≥ 0≠                                                         PTP28600
   1 | DERIVATIVE SIGN^                                                  PTP28700
   -1 | DERIVATIVE SIGN^                                                 PTP28800
SOLUTION FLAG = 0≠^                                                      PTP28900
COMPUTE SPREADING LOSS≠                                                  PTP29000
UP OR DOWN ( 1≠                                                          PTP29100
(COMMENT≠ THESE ALTERNATIVES ARE REVERSED WHEN COMPARED TO THE ONES      PTP29200
IN THE RAY TRACE PROGRAM.  THIS IS DUE TO THE FACT THAT THE              PTP29300
LBI AND UBI HAVE ALREADY BEEN INCREMENTED (DECREMENTED) PRIOR TO         PTP29400
THE MODE COUNT CHECK.)                                                   PTP29500
   C (UBI) / V | TEMP^                                                   PTP29600
   C(LBI)/V|TEMP^                                                        PTP29700
ARCCOS(TEMP^ANGRCVR)  , ANGRCVR*57.2957795|ANGRCVR.                      PTP29800
TEMP=1.0≠                                                                PTP29900
TRACE RAY EXIT.^                                                         PTP30000
1.0 - TEMP * TEMP | TEMP, SQRT (TEMP^ TEMP),                             PTP30100
$RANGE * DERIVATIVE * TEMP/COSTHETA SUB 0$ | TEMP,                       PTP30200
LOG (TEMP^ TEMP), 10.0 * TEMP | SPRD LOSS,                               PTP30300
COMPUTE ATTENUATION≠                                                     PTP30400
ATTN FACTOR * PATH LENGTH | ATTENUATION^                                 PTP30500
TRACE RAY EXIT.                                                          PTP30600
(≠PAGE F5494-7)                                                          PTP30700
ERROR INC≠ ,→PRINTER <>^ <<YOU$HAVE$CHOSEN$A$RELATIVE$                   PTP30800
MAXIMUM$OF$MINIMUM$ON$THE$PROFILE>> <<AS$YOUR$                           PTP30900
VERTEX$VELOCITY.>>*                                                      PTP31000
TRACE RAY EXIT.                                                          PTP31100
RCVR OUTSIDE≠                                                            PTP31200
(≠IT IS POSSIBLE THAT THE RAY COULD HAVE VERTEXED AT THE RECEIVER LAYERPTP31300
 AND THEREFORE IT SHOULD BE CHECKED BEFORE SAYING THE RECEIVER IS        PTP31400
 OUTSIDE THE RAY PATH.)                                                  PTP31500
UPORDOWN=1≠                                                              PTP31600
   $Z(1)→RCVR DEPTH$<0.1≠                                                PTP31700
      CONT VERTEX.^^                                                     PTP31800
   $Z-RCVR DEPTH$<0.1≠                                                   PTP31900
      CONT VERTEX.^^                                                     PTP32000
            ,→PRINTER<<THE$RECEIVER$IS$LOCATED$EITHER$ABOVE$OR$          PTP32100
BELOW$THE$RAY$PATH>>*                                                    PTP32200
TRACE RAY EXIT≠*,.                                                       PTP32300
RCK(4),                                                                  PTP32400
RKEY,                                                                    PTP32500
Z (2),                                                                   PTP32600
A1, A2, A3,                                                              PTP32700
VSQ,                                                                     PTP32800
ALPHA, BETA (2),                                                         PTP32900
RENO,                                                                    PTP33000
TANTHETA (2),                                                            PTP33100
BETATS,                                                                  PTP33200
ROOT1 = 0*0, ROOT2 = 0*0,                                                PTP33300
CMAXIN LAYER.^                                                           PTP33400
(≠THIS FLOWCHART CONTAINS COMPUTE RANGE FUNCTIONS                        PTP33500
WHICH COMPUTE RANGE, DERIVATIVE AND TRAVEL TIME)                         PTP33600
(≠PAGE F5494-8)                                                          PTP33700
COMPUTE RANGE (V SUBX. TABLE INDEX≠TI, SF, RSUBI. DSUBI, TSUBI, PL SUBIPTP33800
, VF)≠                                                                   PTP33900
→DEPTH TABLE (TI) | Z, DEPTH TABLE (TI+1) | Z(1),                        PTP34000
A1TABLE (TI) | A1, A2 TABLE (TI) | A2, A3 TABLE (TI) | A3,               PTP34100
FIND ALPHA≠                                                              PTP34200
```

```
4.0 * A1 | TEMP, A2 * A2 | TEMP [1],                                  PTP34300
V SUBX * V SUBX | VSQ * A3 | TEMP[2],                                 PTP34400
TEMP * (TEMP[2] * 1.0) - VSQ * TEMP[1] | ALPHA,                       PTP34500
(≠THE FØLLØWING 9 LINES CHECK FØR VERTEXING BETWEEN LAYERS)           PTP34600
A1 < 0≠                                                               PTP34700
   VT1≠ VSUBX > C[TI+1] | VSUBX > C[TI]≠                              PTP34800
      NØ VERTEX IN LAYER.                                             PTP34900
      VERTEX IN LAYER.⍟                                               PTP35000
   -A2/(2.0 * A1) | TEMP[3],                                          PTP35100
   Z < TEMP[3] < Z[1]≠⍟ VT1.                                          PTP35200
   C MAX TABLE [TI] | C MAX IN LAYER ≥ V SUBX≠                        PTP35300
      VERTEX IN LAYER≠ 1 | VF⍟                                        PTP35400
     NØ VERTEX IN LAYER≠ 0 | VF⍟⍟                                     PTP35500
(≠PAGE F5494-9)                                                       PTP35600
(≠THE FØLLØWING 13 LINES DETERMINE WHICH RANGE EQUATIØN               PTP35700
SHØULD BE USED)                                                       PTP35800
VF = 0≠                                                               PTP35900
   A1 > 0≠                                                            PTP36000
   1 | RENØ⍟                                                          PTP36100
      2 | RENØ⍟⍟                                                      PTP36200
   A1 > 0≠                                                            PTP36300
      UP ØR DØWN = 1≠                                                 PTP36400
         3 | RENØ⍟                                                    PTP36500
         6 | RENØ⍟⍟                                                   PTP36600
      UP ØR DØWN = 1≠                                                 PTP36700
         5 | RENØ⍟                                                    PTP36800
         4 | RENØ⍟⍟⍟                                                  PTP36900
(≠PAGE F5494-10)                                                      PTP37000
RANGESWI≠                                                             PTP37100
FIND BETAS≠                                                           PTP37200
2.0 * A1 | TEMP[4] * Z + A2 | BETA,                                   PTP37300
TEMP [4] * Z[1] + A2 | BETA[1],                                       PTP37400
RANGE SWITCH [RENØ-1],                                                PTP37500
RANGE SWITCH≠                                                         PTP37600
RANGE EQUATIØN 1,                                                     PTP37700
RANGE EQUATIØN 2,                                                     PTP37800
RANGE EQUATIØN 3,                                                     PTP37900
RANGE EQUATIØN 4,                                                     PTP38000
RANGE EQUATIØN 5,                                                     PTP38100
RANGE EQUATIØN 3A,                                                    PTP38200
(≠PAGE F5494-11)                                                      PTP38300
RANGE EQUATIØN 1≠                                                     PTP38400
RANGE FUNCTIØN 1(V SUBX, BETA, BETA [1], ALPHA⍟ RSUBI), CD,           PTP38500
RANGE EQUATIØN 2≠                                                     PTP38600
RANGE FUNCTIØN 2(VSUBX, BETA, BETA[1], ALPHA⍟ RSUBI), CD,             PTP38700
RANGE EQUATIØN 3≠                                                     PTP38800
RANGE FUNCTIØN 3(VSUBX, BETA, ALPHA⍟ RSUBI), CD,                      PTP38900
RANGE EQUATIØN 4≠                                                     PTP39000
RANGE FUNCTIØN 4(VSUBX, BETA [1], ALPHA⍟ RSUBI), CD,                  PTP39100
RANGE EQUATIØN 5≠                                                     PTP39200
RANGE FUNCTIØN 5(VSUBX, BETA, ALPHA⍟ RSUBI), CD,                      PTP39300
RANGE EQUATIØN 3A≠                                                    PTP39400
RANGE FUNCTIØN 3(VSUBX, BETA[1], ALPHA⍟ RSUBI),                       PTP39500
(≠PAGE F5494-12)                                                      PTP39600
CØMPUTE DERIVATIVE≠ CD≠                                               PTP39700
VF = 1≠                                                               PTP39800
   UP ØR DØWN = 1≠                                                    PTP39900
```

```
      TAN THETA FUNCTION (Z∘ TAN THETA), BETA | BETA TS∘                PTP40000
      TAN THETA FUNCTION (Z [1]∘ TAN THETA), BETA [1] | BETA TS∘        PTP40100
      ZR FLAG ( 0≠                                                      PTP40200
      COMPUTE TRAVEL TIME.∘                                             PTP40300
$TAN THETA SUB0 * (RSUBI + 4.0 * BETA TS / (ALPHA * TAN THETA))$ | DSUBPTP40400
I∘                                                                      PTP40500
   TAN THETA FUNCTION (Z∘ TAN THETA),                                   PTP40600
TANTHETA=0≠                                                             PTP40700
 ,→<<TAN$THETA$ZERO.>>, 0∨82∘∘                                          PTP40800
   TAN THETA FUNCTION (Z[1]∘ TAN THETA[1]),                             PTP40900
   ZR FLAG ( 0≠                                                         PTP41000
   COMPUTE TRAVEL TIME.∘                                                PTP41100
   →$TAN THETA SUB0 * (RSUBI + (2.0/ALPHA) *                            PTP41200
   (BETA[1]/TAN THETA [1] - BETA/TAN THETA)) $ | D SUBI∘                PTP41300
SF * 0≠ EXIT COMPUTE RANGE.∘                                            PTP41400
(≠ TRAVEL TIME COMPUTED SINCE THIS IS A SOLUTION RAY)                   PTP41500
COMPUTE TRAVEL TIME≠                                                    PTP41600
0.25/(A1 * VSUBX) | TEMP, ALPHA * 8.0*A1 | TEMP [1],                    PTP41700
2.0 * BETATS * TAN THETA | TEMP [2],                                    PTP41800
VF = 1≠                                                                 PTP41900
   UP OR DOWN = 1≠                                                      PTP42000
      - TEMP [2] | TEMP [2]∘∘                                           PTP42100
   TEMP * (TEMP [2] + 0.5 * TEMP [1] * RSUBI) | T SUBI∘                 PTP42200
   TEMP * (BETA[1] * TAN THETA [1]- BETA * TAN THETA +                  PTP42300
   0.5 * TEMP [1] * RSUBI) | TSUBI∘                                     PTP42400
(≠PAGE F 5494-13)                                                       PTP42500
FIND PATH LENGTH≠                                                       PTP42600
ZR FLAG ( 0≠                                                            PTP42700
   EXIT COMPUTE RANGE.∘                                                 PTP42800
VF = 1≠∘                                                                PTP42900
   Z [1] - Z | PLSUBI * PLSUBI * RSUBI * RSUBI | PLSUBI,                PTP43000
   SQRT (PLSUBI∘ PLSUBI), EXIT COMPUTE RANGE.                           PTP43100
FIND VERTEXING DEPTH≠ FVD≠                                              PTP43200
SOLVE QUAD IN Z (V SUBX, A1, A2, A3∘ ROOT1, ROOT2),                     PTP43300
Z ≤ ROOT1 ≤ Z [1]≠          (≠COMPARISON ∨1)                            PTP43400
   Z ≤ ROOT 2 ≤ Z [1]≠                                                  PTP43500
      UP OR DOWN ( 1≠                                                   PTP43600
         ROOT 1 > ROOT 2≠                                               PTP43700
            ROOT 2 | Z∘                                                 PTP43800
            ROOT 1 | Z∘∘                                                PTP43900
         ROOT 1 > ROOT 2≠                                               PTP44000
            ROOT 1 | Z [1]∘                                             PTP44100
            ROOT 2 | Z [1]∘∘∘                                           PTP44200
      UP OR DOWN ( 1≠                                                   PTP44300
            ROOT 1 | Z∘                                                 PTP44400
            ROOT 1 | Z [1]∘∘∘          (≠END OF TRUE ∨1)                PTP44500
(≠PAGE F 5494-14)                                                       PTP44600
Z ≤ ROOT 2 ≤ Z [1]≠          (≠BEGIN FALSE ∨1)                          PTP44700
   UP OR DOWN ( 1≠                                                      PTP44800
      ROOT 2 | Z∘                                                       PTP44900
      ROOT 2 | Z [1]∘∘                                                  PTP45000
   DHLE CK ROOT.∘                                                       PTP45100
FPL≠ Z [1] - Z | PLSUBI * PLSUBI + (RSUBI * RSUBI) / 4.0 | PLSUBI,      PTP45200
SQRT (PLSUBI∘ PLSUBI), 2.0 * PLSUBI | PLSUBI,                           PTP45300
EXIT COMPUTE RANGE.                                                     PTP45400
ROOT ERROR≠                                                             PTP45500
, →<< ERROR $ IN $ PATH $ LENGTH $ COMPUTATION >>                       PTP45600
```

```
<< BOTH $ ROOTS 1 OF $ QUADRATIC $ ARE $ OUTSIDE $ LAYER >>
<< ROOT 1$ = > ROOT 1>
<<ROOT2$=>ROOT2>*,NEXT MODE.
DBLE CK ROOT≠
$Z-ROOT1$|RCK,
$Z-ROOT2$|RCK[1],
$Z[1]-ROOT1$|RCK[2],
$Z[1]-ROOT2$|RCK[3],
RCK<RCK[1]≠
   RCK<RCK[2]≠
      RCK<RCK[3]≠
         0|RKEY^
         3|RKEY^^
      RCK[2]<RCK[3]≠
         2|RKEY^
         3|RKEY^^^
   RCK[1]<RCK[2]≠
      RCK[1]<RCK[3]≠
         1|RKEY^
         3|RKEY^^
      RCK[2]<RCK[3]≠
         2|RKEY^
         3|RKEY^^^
RCK[RKEY]<0.1≠^
  ROOT ERROR.
SETVD[RKEY],
SETVD≠
SETRCK,
SETRCK1,
SETRCK2,
SETRCK3,
SETRCK≠ ROOT1|Z,FPL.
SETRCK1≠ROOT2|Z,FPL.
SETRCK2≠ROOT1|Z[1],FPL,
SETRCK3≠ROOT2|Z[1],FPL,
EXIT COMPUTE RANGE≠
ZR1STINC FLAG (0≠
   0 | VF, -UP OR DOWN | UP OR DOWN,
   RSUBI / 2.0 | RSUBI, TSUBI / 2.0 | TSUBI^^$
(≠PAGE + 5494-15)
SOLVE QUAD IN Z (VQ = 0*0, A1Q = 0*0, A2Q = 0*0, A3Q = 0*0, R1, R2, TS
(4),)≠
-VQ * VQ | TS * TS | TS [1],
4.0 * TS * A1Q * (TS * A3Q - 1.0) | TS [2],
TS [1] * A2Q * A2Q - TS [2] | TS [3] < 0≠
   DISC ERROR,^
SQRT (TS [3]^ TS [3]),
-(TS * A2Q - TS [3]) / (2.0 * TS * A1Q) | R1,
-(TS * A2Q + TS [3]) / (2.0 * TS * A1Q) | R2, SQEX.
DISC ERROR≠, -<>^ << ERROR $ IN $ PATH $ LENGTH $ COMPUTATION >>
<< V $ = > VQ >><< A1 $ = > A1Q> << A2 $ = >A2Q> << A3$ => A3Q>$
,NEXT MODE,
SQEX≠$
TAN THETA FUNCTION (DZ, TTTS,)≠
-VSQ *(A1 * DZ * DZ + A2 * DZ + A3) - 1.0 | TTTS,
SQRT (TTTS^ TTTS)$..
7
```

```
(≠PAGE F5494-16)                                                        PTP51400
VB(2),                                                                  PTP51500
VB SQ(2),                                                               PTP51600
RE TEMP(8),                                                             PTP51700
PI @VER TW0 = 1.5707963268⍝                                             PTP51800
RANGE FUNCTIONS≠                                                        PTP51900
RANGE FUNCTION 1(VEE, B(2), ALF, R,)≠                                   PTP52000
→VEE * B | VB * VB | VB SQ + ALF | RETEMP,                              PTP52100
VEE * B[1] | VB[1] * VB[1] | VB SQ[1] + ALF | RETEMP [1],                PTP52200
SQRT (RE TEMP⍝ RE TEMP), SQRT (RE TEMP[1]⍝ RE TEMP[1]),                  PTP52300
(VB[1] + RE TEMP(1))/(VB + RE TEMP) | RE TEMP [2],                       PTP52400
LN (RETEMP[2]⍝ RETEMP[3]),                                              PTP52500
VEE * VEE * A1 | RE TEMP[4], SQRT (RETEMP[4]⍝ RETEMP[4]),                PTP52600
1.0 / RE TEMP[4] * RE TEMP[3] | R⍎                                       PTP52700
RANGE FUNCTION 2(VEE, B(2), ALF, R,)≠                                   PTP52800
→SQRT (-ALF⍝ RETEMP),                                                   PTP52900
VEE * B/RETEMP | RETEMP[1], VEE * B[1]/RETEMP | RETEMP[3],               PTP53000
* VEE * VEE * A1 | RETEMP[5], SQRT (RETEMP[5]⍝ RETEMP[6]),               PTP53100
RETEMP[1]<1.0≠⍝                                                         PTP53200
    VERTEX IN LAYER.                                                    PTP53300
RETEMP[3]<1.0≠⍝                                                         PTP53400
    VERTEX IN LAYER.                                                    PTP53500
ARC SIN (RETEMP[1]⍝ RETEMP[2]), ARCSIN (RETEMP[3]⍝ RETEMP[4]),           PTP53600
(RETEMP[2] - RETEMP[4])/RETEMP[6] | R⍎                                   PTP53700
(≠PAGE F5494-17)                                                        PTP53800
RANGE FUNCTION 3(VEE, B, ALF, R,)≠                                      PTP53900
→VEE * B | VB * VB | VB SQ + ALF | RE TEMP, %VB$ | VB, $ ALF$ | RETEMP PTP54000
[6],                                                                    PTP54100
SQRT (RETEMP⍝ RETEMP[1]), SQRT (RETEMP [6]⍝ RETEMP [2]),                 PTP54200
(VB + RETEMP[1])/RETEMP[2] | RETEMP[3],                                 PTP54300
LN (RETEMP[3]⍝ RETEMP[4]), VEE * VEE * A1 | RETEMP[5],                   PTP54400
SQRT (RETEMP[5]⍝ RETEMP[6]),                                            PTP54500
2.0/RETEMP[6] * RETEMP[4] | R⍎                                           PTP54600
RANGE FUNCTION 4(VEE, B, ALF, R,)≠                                      PTP54700
→VEE * VEE * A1 | RETEMP, SQRT (-RETEMP⍝ RETEMP[1]),                    PTP54800
SQRT (-ALF⍝ RETEMP[2]), VEE * B/RETEMP[2] | RETEMP[3],                  PTP54900
ARC SIN (RETEMP[3]⍝ RETEMP[4]),                                         PTP55000
(PI @VER TW0 - RETEMP[4]) * 2.0/RETEMP[1] | R⍎                          PTP55100
(≠PAGE F5494-18)                                                        PTP55200
RANGE FUNCTION 5(VEE, B, ALF, R,)≠                                      PTP55300
→VEE * VEE * A1 | RETEMP, SQRT (-RETEMP⍝ RETEMP[1]),                    PTP55400
SQRT (-ALF⍝ RETEMP[2]), VEE * B/RETEMP[2] | RETEMP[3],                  PTP55500
ARCSIN (RETEMP[3]⍝ RETEMP[4]),                                          PTP55600
(RETEMP[4] + PI @VER TW0) * 2.0/ RETEMP[1] | R⍎                          PTP55700
ERR IN ARC2≠                                                            PTP55800
,→<<ERR0R$IN$RANGE$EQUATION2,$$ARCSIN$ARG$GT.$1.0>>⍎.,                  PTP55900
7⍝                                                                      PTP56000
(≠PAGE F5494-19)                                                        PTP56100
RAY TRACE MATRIX (RSIGN (2), D SIGN(2)⍝ CASE NUMBER, ROW, COLUMN, EN,   PTP56200
CASE TABLE (16)= 2, 2, 3, 1, 0, 2, 1, 1, 1, 1, 2, 0, 1, 3, 2, 2)≠       PTP56300
→0 | ROW | COLUMN,                                                      PTP56400
RSIGN = 1≠⍝                                                             PTP56500
   2 | ROW⍝                                                             PTP56600
D SIGN = 1≠⍝                                                            PTP56700
   ROW + 1 | ROW⍝                                                       PTP56800
R SIGN [1] = 1≠⍝                                                        PTP56900
   2 | COLUMN⍝                                                          PTP57000
```

```
D SIGN [1] = 1≠⍝                                                        PTP57100
   COLUMN +1 | COLUMN⍝                                                  PTP57200
4 * COLUMN + ROW | EN, CASE TABLE [EN] | CASE NUMBER⋄,.                 PTP57300
7                                                                       PTP57400
(≠RAY ACOUSTIC - CASE FUNCTIONS)                                        PTP57500
(≠PAGE F5276-1)                                                         PTP57600
(≠THIS FLOWCHART CONTAINS THE FUNCTIONS USED TO DETERMINE THE           PTP57700
VERTEX VELOCITIES BETWEEN V AND V[1] FOR WHICH A RAY WILL STRIKE THE REPTP57800
CEIVER)                                                                 PTP57900
RTOLERANCE = 100,0,                                                     PTP58000
V TOLERANCE = 1,0 * ¯2⍝                                                 PTP58100
CASE FUNCTIONS≠                                                         PTP58200
CASE 1 FUNCTION (V (2) = 0*0, R (2) = 0*0, RSIGN (2), VP = 0*0, V2P = 0PTP58300
*0,                                                                     PTP58400
RP = 0*0, R2P = 0*0, RP SIGN, R2 PSIGN)≠ →                              PTP58500
FIND SOLUTION≠                                                          PTP58600
(≠A LINEAR INTERPOLATION METHOD IS USED TO FIND SOLUTION)               PTP58700
INTERSECT PT (V, R, V [1], R [1]⍝ VP), TRACE RAY (VP, 0⍝ RP,, RPSIGN),  PTP58800
RP SIGN = R SIGN≠                                                       PTP58900
   INTERSECT PT (V, R, VP, RP⍝ V2P), VP | V, RP | R⍝                    PTP59000
   INTERSECT PT (VP, RP, V [1], R [1]⍝ V2P), VP | V [1], RP | R [1]⍝    PTP59100
V < V2P < V [1]≠⍝                                                       PTP59200
   CIR≠ (V [1] + V) / 2,0 | V2P⍝                                        PTP59300
TRACERAY (V2P, 0⍝ R2P,, R2PSIGN),                                       PTP59400
R2PSIGN = RSIGN≠                                                        PTP59500
         V2P | V, R2P | R⍝                                              PTP59600
         V2P | V [1], R2P | R [1]⍝                                      PTP59700
(≠PAGE F 5276-2)                                                        PTP59800
TEST≠                                                                   PTP59900
V [1] - V ≤ V TOLERANCE≠ NOW TEST RANGES.⍝                              PTP60000
$ 1,0 - RECEIVER RANGE / R $ <V TOLERANCE ?                             PTP60100
$ 1,0 - RECEIVER RANGE / R [1] $ < V TOLERANCE≠                         PTP60200
   CIR,                                                                 PTP60300
   FIND SOLUTION,                                                       PTP60400
NOW TEST RANGES≠                                                        PTP60500
$ RECEIVER RANGE-R$ < R TOLERANCE ;                                     PTP60600
$ RECEIVER RANGE - R [1] $ < R TOLERANCE≠⍝                              PTP60700
V [1] - V < 1,0 * ¯5≠                                                   PTP60800
   CASE1 EXIT,                                                          PTP60900
   CIR,⍝                                                                PTP61000
CASE 1 EXIT≠                                                            PTP61100
V | VP, V [1] | V2P⋄                                                    PTP61200
INTERSECT PT (V, R, V1, R1, V INTERSECT)≠                               PTP61300
→(RECEIVER RANGE - R) * (V1 - V) / (R1 - R) + V | V INTERSECT,⋄         PTP61400
(≠PAGE F 5276-3)                                                        PTP61500
CASE 20 FUNCTION (V (2), R (2), RSIGN (2), DSIGN (2), VRR (4), RR (4),  PTP61600
   RP, RPSIGN, DP, DPSIGN, VP, PP20, ORIGVP,)≠                          PTP61700
→RECYCLE 20≠                                                            PTP61800
2,0 | PP20,                                                             PTP61900
V [1] - V < 1,0*¯2≠                                                     PTP62000
   0 | VRR | VRR [1] | VRR [2] | VRR [3], CASE 20 EXIT,⍝               PTP62100
V + (V [1] - V) / 2,0 | ORIG VP,                                        PTP62200
RESUB 20≠                                                               PTP62300
V + ( V [1] - V) / PP 20 | VP - V < 1,0*¯3≠                              PTP62400
   ORIG VP | V, TRACE RAY (V, 0⍝ R,, RSIGN, DSIGN), RECYCLE 20,⍝        PTP62500
TRACE RAY (VP, 0⍝ RP, DP, RPSIGN, DPSIGN),                              PTP62600
P201≠                                                                   PTP62700
```

```
RP SIGN = RSIGN≠⍝                                                    PTP62800
   CASE 1FUNCTION (VP, V [1], RP, R [1], RPSIGN, RSIGN [1]⍝          PTP62900
   VRR [2],VRR[3],RR[2],RR[3]),                                      PTP63000
   CASE 1 FUNCTION (V,VP, R, RP, RSIGN,RPSIGN⍝                       PTP63100
   VRR, VRR [1],RR, RR[1]), CASE 20 EXIT.                            PTP63200
(≠PAGE F 5276-4)                                                     PTP63300
DSIGN = DSIGN [1]≠                                                   PTP63400
   DSIGN = DPSIGN≠                                                   PTP63500
      PP20 = 2.0 | PP20, RESUB20.                                    PTP63600
      DPSIGN = 1≠                                                    PTP63700
         R SIGN = 1≠                                                 PTP63800
            PT0IPLUS20.                                              PTP63900
            PT0I20.⍝                                                 PTP64000
         RSIGN = 1≠                                                  PTP64100
            PT0I20.                                                  PTP64200
            PT0IPLUS20.⍝⍝⍝                                           PTP64300
   DPSIGN = DSIGN≠                                                   PTP64400
      P T0I20,                                                       PTP64500
      PT0I PLUS 20.⍝                                                 PTP64600
P T0 I 20≠                                                           PTP64700
VP | V, RP | R, DPSIGN | DSIGN, RPSIGN | RSIGN, RECYCLE 20.          PTP64800
P T0 IPLUS 20≠                                                       PTP64900
VP | V [1], RP | R [1], DPSIGN | DSIGN [1],                          PTP65000
RPSIGN | RSIGN [1], RECYCLE 20,                                      PTP65100
CASE 20 EXIT≠.                                                       PTP65200
(≠PAGE  F 5276-5)                                                    PTP65300
CASE 31 FUNCTION (V(2), R(2), RSIGN(2), DSIGN(2), VRR(6), RR (6),    PTP65400
VP, RP, RPSIGN, DPSIGN, PP31,)≠                                      PTP65500
-0 | VRR | VRR[1] | VRR[2] | VRR [3] | VRR [4] | VRR [5], 2.0 | PP31, PTP65600
   RECYCLE 31≠                                                       PTP65700
V[1] - V < 2.0*-2≠                                                   PTP65800
   CASE 1 FUNCTION (V,V[1], R, R[1], RSIGN, RSIGN[1]⍝                PTP65900
VRR, VRR [1], RR, RR [1]), CASE 31 EXIT.⍝                            PTP66000
V + (V[1] - V) / PP31 | VP,                                          PTP66100
FIRST APPROX 31≠                                                     PTP66200
TRACE RAY (VP, 0⍝ RP, , RPSIGN, DP SIGN),                            PTP66300
$RECEIVER RANGE - RP $ < 1.0 * -2≠                                   PTP66400
PP31 + 1.0 | PP31, RECYCLE31.⍝                                       PTP66500
RP SIGN = R SIGN≠                                                    PTP66600
   P310,                                                             PTP66700
   P311,                                                             PTP66800
P310≠                                                                PTP66900
CASE 20 FUNCTION (V, VP, R, RP, RSIGN, RPSIGN, DSIGN, DPSIGN⍝        PTP67000
VRR,VRR[1],VRR[2],VRR[3],RR,RR[1],RR[2],RR[3]),                      PTP67100
VRR=0≠                                                               PTP67200
   P T0 I 31.                                                        PTP67300
   CASE 1 FUNCTION (VP, V[1], RP, R[1], RPSIGN, RSIGN[1]⍝            PTP67400
VRR[4],VRR[5],RR[4],RR[5]),CASE31EXIT.                               PTP67500
(≠PAGE F5276-6)                                                      PTP67600
P311≠                                                                PTP67700
CASE 20 FUNCTION (VP, V[1], RP, R[1], RPSIGN, RSIGN[1], DPSIGN, DSIGN[1PTP67800
]⍝                                                                   PTP67900
VRR [2], VRR [3], VRR [4], VRR [5], RR[2], RR[3], RR[4], RR[5]),     PTP68000
VRR[2] = 0 ≠                                                         PTP68100
   P T0 I PLUS 131.                                                  PTP68200
   CASE 1 FUNCTION (V, VP, R, RP, RSIGN, RPSIGN⍝ VRR, VRR [1], RR, RR [PTP68300
1]),                                                                 PTP68400
```

```
CASE 31 EXIT.                                                          PTP68500
P T0 I 31≠                                                             PTP68600
VP | V, RP | R, DPSIGN | D SIGN, RECYCLE 31,                           PTP68700
P T0 I PLUS 131≠                                                       PTP68800
VP | V[1], RP | R[1], DPSIGN | DSIGN[1], RECYCLE 31.                   PTP68900
CASE 31 EXIT≠#..                                                       PTP69000
7                                                                      PTP69100
(≠PAGE F 5529-1)                                                       PTP69200
(≠THE FIRST SIX NAMES IN THIS DIMENSIONING STATEMENT                   PTP69300
ARE USED BY THE PROGRAM AS THREE ORDERED PAIRS.)                       PTP69400
TXATBNDRY =00,                                                         PTP69500
RCATBNDRY = 00,                                                        PTP69600
TBI≠ INDEXTX = 00,                                                     PTP69700
RBI≠ INDEXRC = 00,                                                     PTP69800
TX SND SPD,                                                            PTP69900
RC SND SPD,                                                            PTP70000
B1:BNDRY 1 = 0000 ,                                                    PTP70100
B2:BNDRY 2 = 0000 ,                                                    PTP70200
INT=00,    NICHE = 00,    BOTTOM = 00,    TOP = 00,                    PTP70300
,SHIFTED TABLE=00000,0000,                                             PTP70400
DEPTH = 000000,00000,                                                  PTP70500
LEVEL,                                                                 PTP70600
SUDOR,                                                                 PTP70700
DEVICE 1 =00      ,DEVICE 2 = 00 ,                                     PTP70800
ONE,                                                                   PTP70900
ATWO,                                                                  PTP71000
NEG2 = -2.0,                                                           PTP71100
WLOC,                                                                  PTP71200
ZEE=0*0,                                                               PTP71300
CEE = 0*0,                                                             PTP71400
STR,                                                                   PTP71500
STR1,                                                                  PTP71600
CNDX,                                                                  PTP71700
LIMIT,                                                                 PTP71800
IIL LIMIT = 00,                                                        PTP71900
            D TAB= 000000.00000 ,                                      PTP72000
TITLE FLAG^                                                            PTP72100
(≠PAGE F 5529-2)                                                       PTP72200
SEARCH≠ -NMBR OF A1S | IIL LIMIT,                                      PTP72300
INDEX INSTRUMENT LEVEL (ZSUBTX, IIL LIMIT^ INDEXTX, TXATBNDRY),        PTP72400
INDEX INSTRUMENT LEVEL (ZSUBRC, IIL LIMIT^ INDEXRC, RCATBNDRY),        PTP72500
COMPUTE INST C (ZSUBTX, INDEXTX^ TX SND SPD),                          PTP72600
TX SND SPD | TX SOUND SPEED,                                           PTP72700
COMPUTE INST C (ZSUBRC, INDEXRC^ RC SND SPD),                          PTP72800
0 | LEVEL,                                                             PTP72900
ZSUBTX = ZSUBRC≠                                                       PTP73000
   TX AT BNDRY > 0≠                                                    PTP73100
      INDEXTX | B1 | B2, IIL LIMIT - 1 | LIMIT, CONTROL.               PTP73200
      SHIFT DOWN (INDEXTX, IIL LIMIT, 1^), INDEXTX + 1 | INDEXTX | INDEPTP73300
XRC | B1 | B2,                                                         PTP73400
      ZSUBTX | DEPTH TABLE [INDEXTX], TX SND SPD | C [INDEXTX],        PTP73500
      IIL LIMIT | LIMIT + 1 | IIL LIMIT | NMBR OF A1S, CONTROL.^^      PTP73600
(≠PAGE F 5529-3)                                                       PTP73700
ZSUBTX < ZSUBRC≠                                                       PTP73800
   INDEXTX | BNDRY 1, INDEXRC | BNDRY 2, 0 | DEVICE 1, 1 | DEVICE 2^   PTP73900
   INDEXRC | BNDRY 1, INDEX TX | BNDRY 2, 1 | DEVICE 1, 0 | DEVICE 2^   PTP74000
TX AT BNDRY [DEVICE 1] > 0≠                                            PTP74100
```

```
   TX AT BNDRY [DEVICE 2] > 0≠⍝                                          PTP74200
      SHIFT DOWN (B2, IIL LIMIT, 1⍝), INDEXTX [DEVICE 2] +               PTP74300
      1 | B2 | INDEXTX [DEVICE 2], ZSUBTX [DEVICE 2] |                   PTP74400
      DEPTH TABLE [B2], TX SND SPD [DEVICE 2] | C [B2],                  PTP74500
      IIL LIMIT | LIMIT + 1 | IIL LIMIT | NMBR OF A1S⍝                   PTP74600
   IIL LIMIT - 1 | LIMIT, CONTROL.⍝                                      PTP74700
TX AT BNDRY [DEVICE 2] > 0≠                                              PTP74800
   SHIFT DOWN (B1, IIL LIMIT, 1⍝), B1 + 1 | B1 | INDEXTX [DEVICE 1],    PTP74900
   INDEXTX [DEVICE 2] + 1 | B2 | INDEXTX [DEVICE 2],                     PTP75000
   ZSUBTX [DEVICE 1] | DEPTH TABLE [B1],                                 PTP75100
   TX SND SPD [DEVICE 1] | C [B1], IIL LIMIT | LIMIT +                   PTP75200
   1 | IIL LIMIT | NMBR OF A1S⍝                                         PTP75300
   SHIFT DOWN (B2, IIL LIMIT, 2⍝), B2 | SUDOR + 2 | B2 | INDEXTX [DEVICPTP75400
E 2],                                                                    PTP75500
   ZSUBTX [DEVICE 2] | DEPTH TABLE [B2],                                 PTP75600
   TX SND SPD [DEVICE 2] | C [B2], IIL LIMIT + 1 | LIMIT + 1 | IIL LIMIPTP75700
T                                                                        PTP75800
   | NMBR OF A1S,                                                        PTP75900
   SHIFT DOWN (B1, SUDOB, 1⍝), B1 + 1 | B1 | INDEXTX [DEVICE1],          PTP76000
   ZSUBTX [DEVICE 1] | DEPTH TABLE [B1], TX SND SPD [DEVICE 1] | C [B1]PTP76100
⍝                                                                        PTP76200
(≠PAGE F 5529-4)                                                         PTP76300
CONTROL≠ C | STR, CNDX = 0 (1) NMBR OF A1S - 1 →C [CNDX + 1] | STR1 > SPTP76400
TR≠                                                                      PTP76500
   STR1 | STR | CMAX TABLE[CNDX], DEPTH TABLE [CNDX+1] | CMXZTAB [CNDX]PTP76600
⍝                                                                        PTP76700
   STR | CMAX TABLE[CNDX], STR1 | STR, DEPTH TABLE [CNDX] | CMXZTAB [CNPTP76800
DX]⍝.                                                                    PTP76900
BNDRY 1 = 0≠                                                             PTP77000
   0 | MAX V ON PROFILE, ST LOOP2.⍝                                      PTP77100
ST LOOP1≠ C | MAX V ON PROFILE, 0 | TITLE FLAG,                          PTP77200
LOOP 1≠  LEVEL < BNDRY 1≠                                                PTP77300
   SEARCH MAX V ON PROFILE, LOOP1.⍝                                      PTP77400
ST LOOP2≠ C [LEVEL] | INITIAL VELOCITY,                                  PTP77500
LOOP2≠ LEVEL = LIMIT≠                                                    PTP77600
   INITIAL VELOCITY > MAX V ON PROFILE≠                                  PTP77700
      INIVZ | MAX PROFILE DEPTH,                                         PTP77800
      INITIAL VELOCITY | MAX V ON PROFILE⍝⍝ SEARCH MAX V ON PROFILE,     PTP77900
      EXIT SEARCH.⍝                                                      PTP78000
LEVEL = BNDRY 2≠⍝                                                        PTP78100
   SEARCH INITIAL VELOCITY , LOOP2.                                      PTP78200
INITIAL VELOCITY > MAX V ON PROFILE≠                                     PTP78300
   INIVZ | MAX PROFILE DEPTH,                                            PTP78400
   INITIAL VELOCITY | MAX V ON PROFILE⍝⍝                                 PTP78500
LOOP 3≠ LEVEL > LIMIT≠⍝                                                  PTP78600
   SEARCH MAX V ON PROFILE, LOOP3. EXIT SEARCH≠⍝                         PTP78700
(≠PAGE F 5529-5)                                                         PTP78800
INDEX INSTRUMENT LEVEL (DEPTH OF INSTR. LIMIT, INDEX. AT BNDRY, LMFLT.)PTP78900
≠ →FX TO FL (LIMIT⍝ LMFLT), LMFLT * DEPTH OF INSTR / (DEPTH TABLE [LIMIPTP79000
T] - DEPTH TABLE) | INDEX,                                               PTP79100
FL TO FX (INDEX⍝ INDEX),                                                 PTP79200
INDEX > LIMIT≠                                                           PTP79300
   LIMIT | INDEX⍝⍝                                                       PTP79400
INDEX = 0≠                                                               PTP79500
   LOOP.⍝                                                                PTP79600
LP≠ DEPTH TABLE [INDEX] = 0≠                                             PTP79700
   INDEX - 10 | INDEX, INDEX > 0≠                                        PTP79800
```

```
      LP.
      0 | INDEX, LOOP.^^
L$OOP≠  INDEX > LIMIT≠
   ,→PRINTER <> << THE $INSTRUMENT $ LEVEL $ IS $ GREATER $ THAN$ THE $
 VALUES$IN$THE$DEPTH*TABLE.>>. NEXT TRC.^
L$OP≠  DEPTH TABLE [INDEX] = DEPTH OF INSTR≠
   1 | AT BNDRY, EXIT IIL.^
DEPTH TABLE [INDEX] < DEPTH OF INSTR≠
   DEPTH OF INSTR < DEPTH TABLE [INDEX + 1]≠
      -1 | AT BNDRY, EXIT IIL.
      INDEX + 2 | INDEX, LOOP.^
   INDEX - 1 | INDEX, LOP. E$XIT IIL≠ . S$TOP≠..
7^
(≠PAGE F 5529-6)
COMPUTE INSTC (ZEE, INDEX, INSTC, WKLC.)≠ →INDEX = NMBR OF A1S≠
(≠CHECK IF ON BOTTOM) INDEX - 1 | INDEX^^
ZEE * ZEE * A1TABLE [INDEX] + ZEE * A2 TABLE [INDEX] + A3 TABLE [INDEX]
| WKLC, 1.0 / WKLC | WKLC,
SQRT (WKLC^ INSTC).
SHIFT DOWN (TOP, BOTTOM, INT, NICHE,)≠
 *
           DEPTH TABLE[B2] | D TAB  ,
     RDEPTH | DEPTH ,
NICHE = BOTTOM (-1) TOP →
DEPTH TABLE[NICHE] | D TAB , DEPTH TABLE[NICHE+INT] | SHIFTED TABLE,
DEPTH TABLE [NICHE] |  DEPTH TABLE[NICHE + INT] ,
C [NICHE] | C [NICHE + INT],
A1 TABLE [NICHE] | A1 TABLE [NICHE + INT],
A2 TABLE [NICHE] | A2 TABLE [NICHE + INT],
A3 TABLE [NICHE] | A3 TABLE [NICHE + INT].
(≠PAGE F 5529-7)
SEARCH MAX V ON PROFILE≠ →A1 TABLE [LEVEL] | ACNE < 0≠
   SEARCH MV.^
COMPUTE MV≠ A2 TABLE [LEVEL] | ATWO / NEG2 / ACNE | ZEE,
DEPTH TABLE [LEVEL] < ZEE < DEPTH TABLE [LEVEL + 1]≠
   ZEE * ZEE * ACNE + ZEE * ATWO + A3 TABLE [LEVEL] | WLOC,
   1.0 / WLOC | WLOC, SQRT (WLOC^ CEE),
   CEE > CMAX TABLE [LEVEL]≠
      CEE | CMAXTABLE [LEVEL], ZEE | CMXZTAB[LEVEL],
      TITLE FLAG = 0≠
     1|TITLE FLAG,→PRINTER<>,<<CMAX?>CEE<?Z(CMAX)?>ZEE>.^
     ,→PRINTER<?$$$$CEE??$$$ZEE>.^^^^^
SEARCH MV≠
CMAXTABLE [LEVEL] > MAX VON PROFILE≠
   CMAXTABLE [LEVEL] | MAX VON PROFILE,
   CMXZTAB [LEVEL] | MAX PROFILE DEPTH^^
SET MV≠ LEVEL + 1 | LEVEL.
(≠PAGE F5529-8)
SEARCH INITIAL VELOCITY≠ →A1 TABLE [LEVEL] | ACNE < 0≠
   SEARCH IV.^
COMPUTE IV≠ A2 TABLE [LEVEL] | ATWO / NEG2 / ACNE | ZEE,
DEPTH TABLE [LEVEL] < ZEE < DEPTH TABLE [LEVEL +1]≠
   ZEE * ZEE * ACNE + ZEE * ATWO + A3 TABLE [LEVEL] | WLOC,
   1.0 / WLOC | WLOC, SQRT (WLOC^ CEE),
   CEE > CMAXTABLE [LEVEL]≠
      CEE | CMAX TABLE [LEVEL], ZEE | CMXZTAB [LEVEL],
      TITLE FLAG = 0≠
```

```
        1 | TITLE FLAG, →PRINTER <>,,<<CMAX?>CEE<?Z(CMZX)?>ZEE>.^      PTP85600
        ,→PRINTER <?$$$$CEE??$$$ZEE>.^^^^^                              PTP85700
SEARCH IV≠ CMAX TABLE [LEVEL] > INITIAL VEL@CITY≠                       PTP85800
   CMAXTABLE [LEVEL] | INITIAL VEL@CITY,                               PTP85900
   CMXZTAB [LEVEL] | INIVZ^^                                           PTP86000
SET IV≠ LEVEL + 1 | LEVEL.,.                                           PTP86100
```