NBSIR 85-3234

Dynamic Green's Functions of an Infinite Plate - A Computer Program

Nelson N. Hsu

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Gaithersburg, MD 20899

August 1985

Issued November 1985

Prepared for:
U.S. DEPARTMENT OF COMMERCE
Bureau of Standards
burg, MD 20899

NBSIR 85-3234

DYNAMIC GREEN'S FUNCTIONS OF AN INFINITE PLATE - A COMPUTER PROGRAM

Nelson N. Hsu

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Gaithersburg, MD 20899

August 1985

Issued November 1985

Prepared for:
U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Gaithersburg, MD 20899

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

Dynamic Green's Functions of an Infinite Plate - a Computer Program

Nelson N. Hsu
National Bureau of Standards
Gaithersburg, MD 20899 USA

Introduction

This report is a FORTRAN program to compute the Green's functions of an infinite plate. The Green's function, $G_{ij}(\underline{\xi},\underline{x},t)$, is defined as the ith component of the displacement at $\underline{x}$ due to a point force of step-function time dependency acting at $\underline{\xi}$ in the jth direction initiated at t=0. The Green's function is the fundamental solution of the transient elastic wave propagation problem. In general, the displacement field $\underline{u}(\underline{\xi},\underline{x},t)$ at $\underline{x}$ due to a point force of arbitrary time dependence acting at $\underline{\xi}$ can be computed by a convolution integration; i.e.,

$$u_i(\underline{\xi},\underline{x},t) = \int_0^\infty \dot{G}_{ij}(\underline{\xi},\underline{x},t)f_j(\tau-t)d\tau. \qquad (1)$$

Here, $\dot{G}_{ij}$ is the time derivative of G_{ij} and $f_j(t)$ is the point force component of arbitrary time dependence acting in the jth direction (summation over repeated indices is used). Displacement due to point dipoles or couple forces can be represented by the spatial derivatives of G_{ij}. Displacement produced by a dynamic force distributed over a finite area can also be computed by numerical integration using the Green's function as the kernel of the integral over the finite area.

The basic formulation of the problem and derivation of the solution formulas were reported in (1). The method used is called "ray theory" in the seismological literature. Our derivation was based upon John Willis' new Fourier Inversion method (2). Similar computation results can be found in Reference (3), (4), and (5).

The program was originally written for the analysis of acoustic emission signals. Acoustic emission is a nondestructive testing and monitoring technique in which the detection of the transient stress wave produced by localized deformation or cracks are used to locate, and to assess the criticality of the defects. The theoretical computation is an important link to predict how the acoustic emission waveform evolves through the structure. How the theoretical computations are used in the study of acoustic emission can be found in References (1), (6), (7), and (8).

This computer program is made available mainly for its application to calibrate acoustic emission systems and sensors. By making the present program available, duplication of efforts can be avoided, errors in theory can be checked, and experimental results can be reproduced and verified.

How to Use the Program

I. The easiest way to run this program is by setting up nondimensional parameters and calling the subroutine GREENFCT. First, the x-y-z coordinates are selected by choosing an x-y plane parallel to the plate and the origin at the center of the plate directly underneath the source, i.e., the source is always located at $\underline{\xi}$=(0,0,0.5); the x axis is aligned in the direction from the source pointing to the detector. The required nondimensionalized input parameters for the subroutine GREENFCT are:

1. ALPHA = shear wave speed/longitudinal wave speed;

2. XD = x-coordinate of the detector; actual length/thickness of the plate;

3. ZD = z-coordinate of the detector; actual length/thickness of the plate.
   ZD = 0.5 if the source and the detector are on the same side of the plate;
   ZD = -0.5 if they are on opposite sides of the plate.

(Note: the y-coordinate of the detector is always zero because of the way the coordinate system is chosen.)

4. INDEX = ij; subscript of Green's function, integer number 11, 12, 13, 21, 22, 23, 31, 32, or 33. (111, 112, 113, etc. for force dipoles.)

5. TDELTA = sampling time interval in terms of nondimensionalized time unit
   = actual sampling time interval * shear wave speed/thickness of the plate; and

6. NPOINT = Total number of sampling points to be computed; must be an integer.

The subroutine will return a double precision array DISPL of dimension (NPOINT) which corresponds to the desired $G_{ij}(\underline{\xi},\underline{x},t)$ or $G_{ij,k}(\underline{\xi},\underline{x},t)$ sampled at equal time intervals. However, the displacements corresponding to dipoles or couples (i.e., $G_{ij,k}$) are for linear ramp time dependency input. Differentitating the returned DISPL with respect to time once will give the proper Green's function due to step time dependency input.

A simple program calls GREENFCT for given nondimensionalized input paramters together with the output results is included in Appendix A; a program that prompts the user for input parameters of arbitrary physical units and returns displacement in physical units is included in Appendix B. The complete listing of the subroutines is in Appexdix C.

II. Another way to use this program is by calling GREENSUB which is basically a simple function: for given one time T, it returns a displacement DISPL. However, there are two preparatory steps that must be taken in the main program:
(1) RAYTIME (NRAY, 3), the time of arrival table, TA(NRAY) and CN(NRAY, 3), two working arrays, should be dimensioned.
(2) The subroutines INIT and TIMEARRI must be called first.

The subroutine GREENFCT may serve as an example of how to use the subroutine GREENSUB.

III. In addition to the two subroutines mentioned above, there are three subroutines which may be called independently and the user may find them useful in checking experimental or theoretical results.

(1) EPIDIS computes the vertical displacement at the epicenter for given ALPHA and T.

(2) SFWAVE computes the surface wave, which corresponds to the Green's function of a semi-infinite space.

(3) TIMEARRI computes the time of arrival for various ray paths.

Comments in these three subroutines serve as instructions about how to use these subroutines.

Some Remarks

I. There are two machine dependent constants which are used in the two numerical integration routines DGLQI and GLI5T. EPMACH is the largest relative spacing. UFLOW is the smallest positive magnitude. See Reference (9) for details about how to set these constants for different computers.

II. The program uses double precision complex arithmetic. In order to run on those FORTRAN compilers lacking double precision complex function libraries, the user may have to modify the program to single precision arithmetic.

III. The ray method is an exact solution for the Green's function in the time domain produced by summing the contribution from successive arrivals of reflected rays. The numerical computation is accurate if the source and detector are near each other (say less than ten plate thicknesses) and if the maximum time is less than ten dimensionless units. If the sensor and detector are far apart or the maximum time is large, the number of rays arriving at approximately the same time may be too large to compute in a reasonable time and the accumulated error may grow.

IV. The current program is limited to the test configuration where the sensor and detector are on the surface of the plate. The displacement computed for force dipole (i.e., INDEX = 111, 112, etc.) are for point dipole of linear ramp time dependency. Differentiation with respect to time once of the displacement will produce proper Green's functions due to step time dependency (See Reference 10).

Acknowledgement

The author would like to thank Dr. David Kahaner of the Scientific Computing Division, NBS, for supplying the integration subroutines DGLQ and GLIST.

References:

1. D.G. Eitzen, et. al., "Fundamental Developments for Quantitative Acoustic Emission Measurements," EPRI NP-2089, Project 608-1, Palo Alto, CA: Electric Power Research Institute, Oct. 1981.

2. J.R. Willis, "Self-Similar Problems in Elastodynamics," Phil. Trans. of Roy. Soc. London, 274, pp. 435-491, May 1973.

3. Y. Pao, Ralph R. Gajewski and Ahmet N. Ceranouglu, "Acoustic Emission and Transient Waves in an Elastic Plate," J. Acoust. Soc. Am., 65(1), Jan. 1979.

4. A.N. Cernoglu and Y.H. Pao, "Propagation of Eleastic Pulses and Acoustic Emission in a Plate," ASME Journal of Applied Mechanics, 48, pp. 125-147, Mar. 1981.

5. T.M. Proctor, F.R. Breckenridge and Y.H. Pao, "Transient Waves in an Elastic Plate: Theory and Experiment Compared," J. Acoust. Soc. Am., (6) Dec. 1983.

6. N.N. Hsu, J.A. Simmons and S.C. Hardy, "An Approach to Acoustic Emission Signal Analysis -- Theory and Experiment," Materials Evaluation, 35, pp. 100-106, Oct. 1977.

7. N.N. Hsu and S.C. Hardy, "Experiments in Acoustic Emission Waveform Analysis for Characterization of AE Sources, Sensors and Structures; In Elastic Waves and Non-Destructive Testing of Materials," AMD-Vol. 29, Y.H. Pao, Editor, The American Society of Mechanical Engineers, New York, pp. 85-106, Dec. 1978.

8. C. Chang and W. Sachse, "Analysis of Elastic Wave Signals from An Extended Source in A Plate," J. Acoust. Soc. Am., 77(4), April 1985.

9. P.A. Fox, A.D. Hall and N.L. Schryer, Bell Laboratories Computer Science Technical Report No. 47. Or: The PORT Mathematical Subroutine Library, ACM Transactions on Mathematical Software, Vol. 4, No. 2, pp 104-106.

10. J.E. Sinclair, "Epicentre Solutions for Point Multipole Sources in An Elastic Half-Space," J. Phys. D: Appl. Phys., Vol. 12, 1979, pp 1309-1315.

Appendix A

Sample Program and Results

```
C*****************************************************************
C
      PROGRAM TSGREEN
C
C     TEST PROGRAM TO COMPUTE GREEN'S FUNCTION OF A PLATE
C
      DOUBLE PRECISION DISPLC(256)
C
C     WRITE(5,*)'TYPE INDEX,ALPHA,XD, AND ZD'
C     READ(5,*) INDEX,ALPHA,XD,ZD
      INDEX=33
      ALPHA=0.59902
      XD=1.0
      ZD=-0.5
C     WRITE(5,*) 'TYPE DELTAT AND NUMBER OF POINTS'
C     READ(5,*) TDELTA,NPOINT
      TDELTA=0.01
      NPOINT=256
C
      CALL GREENFCT(ALPHA,XD,ZD,INDEX,TDELTA,NPOINT,DISPLC)
C
      DO 1000 I=1,256
         TIME = 0.01*REAL(I)
         WRITE(6,1001)  TIME,DISPLC(I)
 1000 CONTINUE
 1001 FORMAT(2X,'TIME = ',F8.3,',   DISPLACEMENT = ',D21.14)
      STOP
      END
```

```
TIME =    0.010,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.020,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.030,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.040,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.050,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.060,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.070,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.080,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.090,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.100,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.110,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.120,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.130,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.140,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.150,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.160,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.170,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.180,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.190,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.200,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.210,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.220,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.230,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.240,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.250,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.260,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.270,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.280,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.290,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.300,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.310,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.320,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.330,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.340,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.350,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.360,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.370,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.380,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.390,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.400,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.410,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.420,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.430,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.440,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.450,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.460,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.470,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.480,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.490,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.500,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.510,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.520,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.530,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.540,   DISPLACEMENT =  0.00000000000000D+00
```

```
TIME =    0.550,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.560,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.570,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.580,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.590,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.600,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.610,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.620,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.630,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.640,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.650,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.660,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.670,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.680,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.690,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.700,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.710,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.720,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.730,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.740,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.750,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.760,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.770,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.780,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.790,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.800,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.810,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.820,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.830,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.840,   DISPLACEMENT =  0.00000000000000D+00
TIME =    0.850,   DISPLACEMENT =  0.11075067698712D+00
TIME =    0.860,   DISPLACEMENT =  0.11098606156782D+00
TIME =    0.870,   DISPLACEMENT =  0.11118024981467D+00
TIME =    0.880,   DISPLACEMENT =  0.11133277774482D+00
TIME =    0.890,   DISPLACEMENT =  0.11144318482884D+00
TIME =    0.900,   DISPLACEMENT =  0.11151102652121D+00
TIME =    0.910,   DISPLACEMENT =  0.11153587694415D+00
TIME =    0.920,   DISPLACEMENT =  0.11151733605821D+00
TIME =    0.930,   DISPLACEMENT =  0.11145503541696D+00
TIME =    0.940,   DISPLACEMENT =  0.11134864281672D+00
TIME =    0.950,   DISPLACEMENT =  0.11119787020707D+00
TIME =    0.960,   DISPLACEMENT =  0.11100247472178D+00
TIME =    0.970,   DISPLACEMENT =  0.11076226552545D+00
TIME =    0.980,   DISPLACEMENT =  0.11047710764219D+00
TIME =    0.990,   DISPLACEMENT =  0.11014692394960D+00
TIME =    1.000,   DISPLACEMENT =  0.10977171429770D+00
TIME =    1.010,   DISPLACEMENT =  0.10935150914128D+00
TIME =    1.020,   DISPLACEMENT =  0.10888645033515D+00
TIME =    1.030,   DISPLACEMENT =  0.10837678979646D+00
TIME =    1.040,   DISPLACEMENT =  0.10782270845288D+00
TIME =    1.050,   DISPLACEMENT =  0.10722461062043D+00
TIME =    1.060,   DISPLACEMENT =  0.10658293623297D+00
TIME =    1.070,   DISPLACEMENT =  0.10589820949205D+00
TIME =    1.080,   DISPLACEMENT =  0.10517111210831D+00
TIME =    1.090,   DISPLACEMENT =  0.10440220346099D+00
```

```
TIME =    1.100,   DISPLACEMENT =  0.10359233809357D+00
TIME =    1.110,   DISPLACEMENT =  0.10274239332029D+00
TIME =    1.120,   DISPLACEMENT =  0.10185342388500D+00
TIME =    1.130,   DISPLACEMENT =  0.10092631946377D+00
TIME =    1.140,   DISPLACEMENT =  0.99962313600975D-01
TIME =    1.150,   DISPLACEMENT =  0.98962647939981D-01
TIME =    1.160,   DISPLACEMENT =  0.97928754785951D-01
TIME =    1.170,   DISPLACEMENT =  0.96861858478749D-01
TIME =    1.180,   DISPLACEMENT =  0.95763566016467D-01
TIME =    1.190,   DISPLACEMENT =  0.94635474207730D-01
TIME =    1.200,   DISPLACEMENT =  0.93479376354866D-01
TIME =    1.210,   DISPLACEMENT =  0.92296816327325D-01
TIME =    1.220,   DISPLACEMENT =  0.91089747924766D-01
TIME =    1.230,   DISPLACEMENT =  0.89860093569871D-01
TIME =    1.240,   DISPLACEMENT =  0.88609970379555D-01
TIME =    1.250,   DISPLACEMENT =  0.87341207961754D-01
TIME =    1.260,   DISPLACEMENT =  0.86056060561906D-01
TIME =    1.270,   DISPLACEMENT =  0.84756728127925D-01
TIME =    1.280,   DISPLACEMENT =  0.83445596172436D-01
TIME =    1.290,   DISPLACEMENT =  0.82124730252424D-01
TIME =    1.300,   DISPLACEMENT =  0.80796622274542D-01
TIME =    1.310,   DISPLACEMENT =  0.79463686723194D-01
TIME =    1.320,   DISPLACEMENT =  0.78128380592089D-01
TIME =    1.330,   DISPLACEMENT =  0.76793325943265D-01
TIME =    1.340,   DISPLACEMENT =  0.75460795794622D-01
TIME =    1.350,   DISPLACEMENT =  0.74133473644589D-01
TIME =    1.360,   DISPLACEMENT =  0.72813939012166D-01
TIME =    1.370,   DISPLACEMENT =  0.71504914223321D-01
TIME =    1.380,   DISPLACEMENT =  0.70208761123235D-01
TIME =    1.390,   DISPLACEMENT =  0.68928225721454D-01
TIME =    1.400,   DISPLACEMENT =  0.63777644085523D-01
TIME =    1.410,   DISPLACEMENT = -0.31205499826477D-01
TIME =    1.420,   DISPLACEMENT =  0.36764444097372D+00
TIME =    1.430,   DISPLACEMENT =  0.39105863151275D+00
TIME =    1.440,   DISPLACEMENT =  0.40949206785671D+00
TIME =    1.450,   DISPLACEMENT =  0.42584966367431D+00
TIME =    1.460,   DISPLACEMENT =  0.44102467848492D+00
TIME =    1.470,   DISPLACEMENT =  0.45540362930489D+00
TIME =    1.480,   DISPLACEMENT =  0.46918701309117D+00
TIME =    1.490,   DISPLACEMENT =  0.48248810354954D+00
TIME =    1.500,   DISPLACEMENT =  0.49537929192411D+00
TIME =    1.510,   DISPLACEMENT =  0.50790390548020D+00
TIME =    1.520,   DISPLACEMENT =  0.52009143324477D+00
TIME =    1.530,   DISPLACEMENT =  0.53196191904305D+00
TIME =    1.540,   DISPLACEMENT =  0.54352825516419D+00
TIME =    1.550,   DISPLACEMENT =  0.55480267134178D+00
TIME =    1.560,   DISPLACEMENT =  0.56579147374873D+00
TIME =    1.570,   DISPLACEMENT =  0.57650038247907D+00
TIME =    1.580,   DISPLACEMENT =  0.58693303751993D+00
TIME =    1.590,   DISPLACEMENT =  0.59709536636170D+00
TIME =    1.600,   DISPLACEMENT =  0.60698989951039D+00
TIME =    1.610,   DISPLACEMENT =  0.61661997278485D+00
TIME =    1.620,   DISPLACEMENT =  0.62598797275653D+00
TIME =    1.630,   DISPLACEMENT =  0.63509897646406D+00
TIME =    1.640,   DISPLACEMENT =  0.64395545056472D+00
```

TIME =	1.650,	DISPLACEMENT =	0.65256088669347D+00
TIME =	1.660,	DISPLACEMENT =	0.66091813088975D+00
TIME =	1.670,	DISPLACEMENT =	0.66903255717167D+00
TIME =	1.680,	DISPLACEMENT =	0.67690728626371D+00
TIME =	1.690,	DISPLACEMENT =	0.68454639170679D+00
TIME =	1.700,	DISPLACEMENT =	0.69195338607382D+00
TIME =	1.710,	DISPLACEMENT =	0.69913401239547D+00
TIME =	1.720,	DISPLACEMENT =	0.70609199225996D+00
TIME =	1.730,	DISPLACEMENT =	0.71283185830127D+00
TIME =	1.740,	DISPLACEMENT =	0.71935822154785D+00
TIME =	1.750,	DISPLACEMENT =	0.72567516353530D+00
TIME =	1.760,	DISPLACEMENT =	0.73178861132189D+00
TIME =	1.770,	DISPLACEMENT =	0.73770270898197D+00
TIME =	1.780,	DISPLACEMENT =	0.74342223326031D+00
TIME =	1.790,	DISPLACEMENT =	0.74895144993679D+00
TIME =	1.800,	DISPLACEMENT =	0.75429619306931D+00
TIME =	1.810,	DISPLACEMENT =	0.75946069053969D+00
TIME =	1.820,	DISPLACEMENT =	0.76444968084343D+00
TIME =	1.830,	DISPLACEMENT =	0.76926741601862D+00
TIME =	1.840,	DISPLACEMENT =	0.77391947485854D+00
TIME =	1.850,	DISPLACEMENT =	0.77840999911627D+00
TIME =	1.860,	DISPLACEMENT =	0.78274354050484D+00
TIME =	1.870,	DISPLACEMENT =	0.78692419559247D+00
TIME =	1.880,	DISPLACEMENT =	0.79095718088098D+00
TIME =	1.890,	DISPLACEMENT =	0.79484643653490D+00
TIME =	1.900,	DISPLACEMENT =	0.86806930229072D+00
TIME =	1.910,	DISPLACEMENT =	0.87369508356078D+00
TIME =	1.920,	DISPLACEMENT =	0.87923314459302D+00
TIME =	1.930,	DISPLACEMENT =	0.88468749838846D+00
TIME =	1.940,	DISPLACEMENT =	0.89006261168853D+00
TIME =	1.950,	DISPLACEMENT =	0.89536286267026D+00
TIME =	1.960,	DISPLACEMENT =	0.90059204444514D+00
TIME =	1.970,	DISPLACEMENT =	0.90575535254114D+00
TIME =	1.980,	DISPLACEMENT =	0.91085639280247D+00
TIME =	1.990,	DISPLACEMENT =	0.91589918253638D+00
TIME =	2.000,	DISPLACEMENT =	0.92088717546076D+00
TIME =	2.010,	DISPLACEMENT =	0.92582515984505D+00
TIME =	2.020,	DISPLACEMENT =	0.93071640538221D+00
TIME =	2.030,	DISPLACEMENT =	0.93556457171401D+00
TIME =	2.040,	DISPLACEMENT =	0.94037277440757D+00
TIME =	2.050,	DISPLACEMENT =	0.94514541664089D+00
TIME =	2.060,	DISPLACEMENT =	0.94988543954263D+00
TIME =	2.070,	DISPLACEMENT =	0.95459616131288D+00
TIME =	2.080,	DISPLACEMENT =	0.95928037273261D+00
TIME =	2.090,	DISPLACEMENT =	0.96394212326014D+00
TIME =	2.100,	DISPLACEMENT =	0.96858404264596D+00
TIME =	2.110,	DISPLACEMENT =	0.97320913182266D+00
TIME =	2.120,	DISPLACEMENT =	0.97781987758092D+00
TIME =	2.130,	DISPLACEMENT =	0.98242001202845D+00
TIME =	2.140,	DISPLACEMENT =	0.98701187638821D+00
TIME =	2.150,	DISPLACEMENT =	0.99159818288714D+00
TIME =	2.160,	DISPLACEMENT =	0.99618157630830D+00
TIME =	2.170,	DISPLACEMENT =	0.10007641987912D+01
TIME =	2.180,	DISPLACEMENT =	0.10053494399595D+01
TIME =	2.190,	DISPLACEMENT =	0.10099393168333D+01

```
TIME =    2.200,   DISPLACEMENT =  0.10145362237750D+01
TIME =    2.210,   DISPLACEMENT =  0.10191420575858D+01
TIME =    2.220,   DISPLACEMENT =  0.10237599774067D+01
TIME =    2.230,   DISPLACEMENT =  0.10283917712906D+01
TIME =    2.240,   DISPLACEMENT =  0.10330396140693D+01
TIME =    2.250,   DISPLACEMENT =  0.10377051845379D+01
TIME =    2.260,   DISPLACEMENT =  0.10423914489334D+01
TIME =    2.270,   DISPLACEMENT =  0.10470999932314D+01
TIME =    2.280,   DISPLACEMENT =  0.10518328029171D+01
TIME =    2.290,   DISPLACEMENT =  0.10565913655695D+01
TIME =    2.300,   DISPLACEMENT =  0.10613784905069D+01
TIME =    2.310,   DISPLACEMENT =  0.11476631005725D+01
TIME =    2.320,   DISPLACEMENT =  0.11533185695767D+01
TIME =    2.330,   DISPLACEMENT =  0.11590500263745D+01
TIME =    2.340,   DISPLACEMENT =  0.11648631801910D+01
TIME =    2.350,   DISPLACEMENT =  0.11707621604698D+01
TIME =    2.360,   DISPLACEMENT =  0.11767516974758D+01
TIME =    2.370,   DISPLACEMENT =  0.11828365811738D+01
TIME =    2.380,   DISPLACEMENT =  0.11890210681560D+01
TIME =    2.390,   DISPLACEMENT =  0.11953112520959D+01
TIME =    2.400,   DISPLACEMENT =  0.12017115269290D+01
TIME =    2.410,   DISPLACEMENT =  0.12082269375580D+01
TIME =    2.420,   DISPLACEMENT =  0.12148619575064D+01
TIME =    2.430,   DISPLACEMENT =  0.12216230328019D+01
TIME =    2.440,   DISPLACEMENT =  0.12285147870224D+01
TIME =    2.450,   DISPLACEMENT =  0.12355425415795D+01
TIME =    2.460,   DISPLACEMENT =  0.12427109991126D+01
TIME =    2.470,   DISPLACEMENT =  0.12500269925297D+01
TIME =    2.480,   DISPLACEMENT =  0.12574953844797D+01
TIME =    2.490,   DISPLACEMENT =  0.12651217887352D+01
TIME =    2.500,   DISPLACEMENT =  0.12729111430751D+01
TIME =    2.510,   DISPLACEMENT =  0.12808706969625D+01
TIME =    2.520,   DISPLACEMENT =  0.12890055553986D+01
TIME =    2.530,   DISPLACEMENT =  0.12973216346622D+01
TIME =    2.540,   DISPLACEMENT =  0.13058241075455D+01
TIME =    2.550,   DISPLACEMENT =  0.13145206649727D+01
TIME =    2.560,   DISPLACEMENT =  0.13234166514682D+01
TIME =    2.080,   DISPLACEMENT =  0.95928037273261D+00
TIME =    2.090,   DISPLACEMENT =  0.96394212326014D+00
TIME =    2.100,   DISPLACEMENT =  0.96858404264596D+00
TIME =    2.110,   DISPLACEMENT =  0.97320913182266D+00
TIME =    2.120,   DISPLACEMENT =  0.97781987758092D+00
TIME =    2.130,   DISPLACEMENT =  0.98242001202845D+00
TIME =    2.140,   DISPLACEMENT =  0.98701187638821D+00
TIME =    2.150,   DISPLACEMENT =  0.99159818288714D+00
TIME =    2.160,   DISPLACEMENT =  0.99618157630830D+00
TIME =    2.170,   DISPLACEMENT =  0.10007641987912D+01
TIME =    2.180,   DISPLACEMENT =  0.10053494399595D+01
TIME =    2.190,   DISPLACEMENT =  0.10099393168333D+01
TIME =    2.200,   DISPLACEMENT =  0.10145362237750D+01
TIME =    2.210,   DISPLACEMENT =  0.10191420575858D+01
TIME =    2.220,   DISPLACEMENT =  0.10237599774067D+01
TIME =    2.230,   DISPLACEMENT =  0.10283917712906D+01
TIME =    2.240,   DISPLACEMENT =  0.10330396140693D+01
TIME =    2.250,   DISPLACEMENT =  0.10377051845379D+01
```

```
TIME =    2.260,   DISPLACEMENT =  0.10423914489334D+01
TIME =    2.270,   DISPLACEMENT =  0.10470999932314D+01
TIME =    2.280,   DISPLACEMENT =  0.10518328029171D+01
TIME =    2.290,   DISPLACEMENT =  0.10565913655695D+01
TIME =    2.300,   DISPLACEMENT =  0.10613784905069D+01
TIME =    2.310,   DISPLACEMENT =  0.11476631005725D+01
TIME =    2.320,   DISPLACEMENT =  0.11533185695767D+01
TIME =    2.330,   DISPLACEMENT =  0.11590500263745D+01
TIME =    2.340,   DISPLACEMENT =  0.11648631801910D+01
TIME =    2.350,   DISPLACEMENT =  0.11707621604698D+01
TIME =    2.360,   DISPLACEMENT =  0.11767516974758D+01
TIME =    2.370,   DISPLACEMENT =  0.11828365811738D+01
TIME =    2.380,   DISPLACEMENT =  0.11890210681560D+01
TIME =    2.390,   DISPLACEMENT =  0.11953112520959D+01
TIME =    2.400,   DISPLACEMENT =  0.12017115269290D+01
TIME =    2.410,   DISPLACEMENT =  0.12082269375580D+01
TIME =    2.420,   DISPLACEMENT =  0.12148619575064D+01
TIME =    2.430,   DISPLACEMENT =  0.12216230328019D+01
TIME =    2.440,   DISPLACEMENT =  0.12285147870224D+01
TIME =    2.450,   DISPLACEMENT =  0.12355425415795D+01
TIME =    2.460,   DISPLACEMENT =  0.12427109991126D+01
TIME =    2.470,   DISPLACEMENT =  0.12500269925297D+01
TIME =    2.480,   DISPLACEMENT =  0.12574953844797D+01
TIME =    2.490,   DISPLACEMENT =  0.12651217887352D+01
TIME =    2.500,   DISPLACEMENT =  0.12729111430751D+01
TIME =    2.510,   DISPLACEMENT =  0.12808706969625D+01
TIME =    2.520,   DISPLACEMENT =  0.12890055553986D+01
TIME =    2.530,   DISPLACEMENT =  0.12973216346622D+01
TIME =    2.540,   DISPLACEMENT =  0.13058241075455D+01
TIME =    2.550,   DISPLACEMENT =  0.13145206649727D+01
TIME =    2.560,   DISPLACEMENT =  0.13234166514682D+01
```

Appendix B

Another sample program - use physical units rather nondimensionalized numbers.

```
      PROGRAM GPLATE
      DOUBLE PRECISION DISPV(1024)
      DIMENSION TIME(1024)
      INTEGER YORNO
C
      WRITE(1,*)'THIS PROGRAM WILL COMPUTE THE DISPLACEMENT U SUB I AT'
      WRITE(1,*)' A POINT ON THE SURFACE OF AN INFINITE PLATE DUE TO A'
      WRITE(1,*)' POINT STEP IMPULSIVE FORCE F SUB J OR A POINT LINEAR'
      WRITE(1,*)' RAMP FORCE DIPOLE F SUB J,K ACTING AT ANOTHER POINT'
      WRITE(1,*)' ON THE SURFACE OF THE PLATE.'
      WRITE(1,*)' INPUTS REQUIRED ARE PROMPTED AND TYPED IN ON THE '
      WRITE(1,*)' KEYBOARD.'
      WRITE(1,*)' '
   10 WRITE(1,*)' SUBSCRIPTS OF G = ? (i. e. 11,13,131,322, etc.)'
      READ(1,*) INDEX
      WRITE(1,*)' SHEAR WAVE SPEED = ?'
      READ(1,*) SHSPD
      WRITE(1,*)' LONGITUDINAL WAVE SPEED = ?'
         READ(1,*) PSPD
  100 WRITE(1,*)'ARE SOURCE AND DETECTER ON THE SAME SIDE OF THE PLATE'
      WRITE(1,*)' PLEASE TYPE 1 FOR YES'
      READ (1,*) YORNO
      IF (YORNO.EQ.1) THEN
         ZD = 0.5
      ELSE
         ZD = -0.5
      END IF
      WRITE(1,*)' WHAT IS THE THICKNESS OF THE PLATE ?'
      READ(1,*) HTHICK
      WRITE(1,*)' WHAT IS THE DISTANCE BETWEEN DETECTER AND SOURCE'
      WRITE(1,*)' OR EPICENTER ?'
      READ(1,*) X
      WRITE(1,*)' SAMPLING TIME INTERVAL = ?'
      READ(1,*) DT
      WRITE(1,*)' NUMBER OF SAMPLING POINTS = ? (MAX = 1024)'
      READ(1,*) NPT
      WRITE(1,*)' SHEAR MODULUS OF THE PLATE = ?'
      READ(1,*) SHMDUL
      WRITE(1,*)' THE OUTPUT WILL BE TIME VERSUS DISPLACEMENT.'
      WRITE(1,*)' '
C
      ALPHA = SHSPD/PSPD
      XD = X/HTHICK
      TSCALE = HTHICK/SHSPD
      DELTAT = DT/TSCALE
      DSCALE = 1./(2.*ASIN(1.0)*SHMDUL*HTHICK)
C
      WRITE(2,*)'  '
      WRITE(2,*)' THE FOLLOWINGS ARE INPUT PARAMETERS:'
      WRITE(2,*)'                      INDEX = ',INDEX
      WRITE(2,*)'           SHEAR WAVE SPEED = ',SHSPD
      WRITE(2,*)'LONGITUDINAL WAVE SPEED = ',PSPD
      WRITE(2,*)'THICKNESS OF THE PLATE  = ',HTHICK
      WRITE(2,*)'SAMPLING TIME INTERVAL  = ',DT
```

```
      WRITE(2,*)'          SHEAR MODULUS = ',SHMDUL
      WRITE(2,*)'NUMBER OF POINTS TO BE COMPUTED = ',NPT
      IF (YORNO.EQ.1) THEN
         WRITE(2,*) ' THE DETECTER AND THE SOURCE ARE ON THE SAME SIDE'
      ELSE
         WRITE(2,*)'THE DETECTER AND THE SOURCE ARE ON OPPOSITE SIDES'
      END IF
C
      WRITE(2,*) ' '
      CALL GREENFCT(ALPHA,XD,ZD,INDEX,DELTAT,NPT,DISPV)
      DO 200 I=1,NPT
         TIME(I)=DELTAT*REAL(I)*TSCALE
         DISPV(I)=DISPV(I)*DSCALE
         WRITE(2,*)' TIME = ',TIME(I),'     DISPLACEMENT = ',DISPV(I)
  200 CONTINUE
      STOP
      END
```

Appendix C

Listing of Subroutines for Computing Green's function of an infinite plate.

```
C*********************************************************************
C
      SUBROUTINE GREENFCT(ALPHA,XD,ZD,INDEX,TDELTA,NPOINT,DISPL)
C
C*********************************************************************
C
C     TO COMPUT GREEN'S FUNCTION - DISPLACEMENT AS A FUNCTION OF TIME
C     - FOR GIVEN TEST CONFIGURATION
C
C     INPUTS:
C        ALPHA = SHEAR WAVE SPEED / LONGITUDINAL WAVE SPEED
C           XD = X COORDINATE OF THE DETECTOR
C           ZD = Z COORDINATE OF DETECTOR
C                ZD = 0.5 => DETECTOR & SOURCE ON THE SAME SIDE
C                ZD =-0.5 => DETECTOR & SOURCE ON OPPOSIT SIDES
C        INDEX = SUBSCRIPT OF GREEN'S FUNCTION, i.e. 33, 322
C       TDELTA = TIME INCREMENT
C       NPOINT = TOTAL NUMBER OF POINTS TO BE COMPUTED
C
C     OUTPUT:
C        DISPL = DISPLACEMENT ** DOUBLE PRECISION, MUST BE
C                DIMENSIONED (NPOINT)
C---------------------------------------------------------------------
C
C     NOTE 1:
C        ALL INPUTS AND OUTPUTS ARE NONDIMENSIONALIZED PARAMETERS
C      DISPL = NORMALIZED DISPLACEMENT = ACTUAL DISPLACEMENT * PI
C                  * SHEAR MODULUS * PLATE THICKNESS / FORCE
C      T = NORMALIZED TIME = ACTUAL TIME *SHEAR WAVE SPEED / PLATE
C                 THICKNESS
C        DISPL MUST BE DIMENSIONED IN THE CALLING PROGRAM (NPOINT)
C
C---------------------------------------------------------------------
C
C     NOTE 2:
C     THE ARRIVAL TIMES ARE STORED IN RAYTIME(N,I)
C        N = THE Nth RAY
C        RAYTIME(N,3) = THE ARRIVAL TIME
C        RAYTIME(N,1) = N1, NUMBER OF P TRIPS
C        RAYTIME(N,2) = N2, NUMBER OF S TRIPS
C           IF N2<0, IT IS A HEAD WAVE
C           IF N1=-1 , IT IS SURFACE WAVE
C           IF N1=0 AND N2=0, IT IS A RAYLEIGH WAVE
C
C     THE TIME OF ARRIVAL INFORMATION IS ALSO PRINTED ON LU 6
C
C---------------------------------------------------------------------
C
      DOUBLE PRECISION DISPL(NPOINT)
      DIMENSION TA(101),CN(101,3),RAYTIME(101,3)
C     NRAY = NUMBER OF MAXIMUM RAYS EXPECTED
      DATA NRAY/101/
C
      CALL INIT(ALPHA,XD,ZD,INDEX)
```

```
      CALL TIMEARRI(ALPHA,XD,ZD,NRAY,RAYTIME,TA,CN)
      WRITE(6,*) '    I      N1        N2     TIME ARRIVAL'
      DO 500 I=1,NRAY
         WRITE (6,499) I,RAYTIME(I,1),RAYTIME(I,2),RAYTIME(I,3)
  499    FORMAT(2X,I5,3X,3F10.5)
  500 CONTINUE
      DO 1000 I=1,NPOINT
         TIMEI = FLOAT(I)*TDELTA
       CALL GREENSUB(ALPHA,INDEX,XD,ZD,NRAY,TIMEI,RAYTIME,DISPL(I))
          WRITE(6,*)'TIME = ',TIMEI,'        DISPL = ',DISPL(I)
 1000 CONTINUE
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE INIT(ALPHA,XD,ZD,INDEX)
C
C*********************************************************************
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
      COMMON /BLK5/YR,Y1,Y2
      DXD = DBLE(XD)
      DZ  = DBLE(ZD)
      DA  = DBLE(ALPHA)
      IF (ZD.EQ.0.5) KASE = 1
      IF (ZD.EQ.-0.5) KASE = 2
      IF (INDEX.LT.100) THEN
         K = 0
         M = INDEX/10
         L = INDEX - M*10
      ELSE IF ((INDEX.GT.100).AND.(INDEX.LT.334)) THEN
         M = INDEX/100
         L = (INDEX - M*100)/10
         K = (INDEX - M*100 - L*10)
      ELSE
         WRITE(6,*) '  WRONG INDEX '
         PAUSE
      END IF
      YI = DCMPLX(0.D0,1.D0)
      YONE = DCMPLX(1.D0,0.D0)
      YTWO = DCMPLX(2.D0,0.D0)
      YTHREE = DCMPLX(3.D0,0.D0)
      YFOUR = DCMPLX(4.D0,0.D0)
      YEIGHT = DCMPLX(8.D0,0.D0)
      YZERO = DCMPLX(0.D0,0.D0)
      CALL RAYRT(YR,Y1,Y2,ALPHA)
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE GREENSUB(ALPHA,INDEX,XD,ZD,NRAY,T,RAYTIME,DISPL)
C
```

```
C************************************************************************
C     SUBROUTINE TO COMPUTE THE GREENS FUNCTION OF A PLATE AT T
C
C   INPUTS:
C     CONFIGURATION PARAMETERS:
C        DETECTOR LOCATION:
C           X = XD
C           Y = 0
C           Z = ZD, ZD=0.5 => TOP; ZD=-0.5 => BOTTOM
C        SOURCE LOCATION:
C           X0 = 0
C           Y0 = 0
C           Z0 = +0.5 (ON TOP OF THE PLATE)
C
C     MATERIALS PARAMETERS:
C        ALPHA = RATIO OF SHEAR VELOCITY TO LONGITUDINAL VELOCITY
C
C     OTHER PARAMETERS:
C        INDEX: SUBSCRIPTS OF GREENS FUNCTION  i.e. 33, or 113
C        NRAY: MAXIMUM NUMBER OF RAYS EXPECTED. ~100
C        RAYTIME: TIME OF ARRIVAL TABLE OF DIMENSION (NRAY,3)
C
C      T = NORMALIZED TIME = ACTUAL TIME *SHEAR WAVE SPEED / PLATE
C                 THICKNESS
C
C     DISPLACEMENT WILL BE CORRECT ONLY IF T < RAYTIME(NRAY,3)
C
C   OUTPUT:
C     DISPL = NORMALIZED DISPLACEMENT = ACTUAL DISPLACEMENT * PI
C                 * SHEAR MODULUS * PLATE THICKNESS / FORCE
C
C     NOTE: SUBROUTINE INIT(ALPHA,XD,ZD,INDEX) AND
C           SUBROUTINE TIMEARRI(.......) SHOULD BE CALLED FIRST IN
C             THE MAIN PROGRAM.
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DOUBLE PRECISION DINTG
      DIMENSION RAYTIME(NRAY,3)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
      COMMON /BLK5/YR,Y1,Y2
C
        EXTERNAL DINTG
C
C
      IF ( (INDEX.EQ.12).OR.(INDEX.EQ.21).OR.
     +     (INDEX.EQ.23).OR.(INDEX.EQ.32).OR.
     +     (INDEX.EQ.121).OR.(INDEX.EQ.211).OR.
     +     (INDEX.EQ.231).OR.(INDEX.EQ.321).OR.
     +     (INDEX.EQ.123).OR.(INDEX.EQ.213).OR.
```

```
     +    (INDEX.EQ.233).OR.(INDEX.EQ.323).OR.
     +    (INDEX.EQ.132).OR.(INDEX.EQ.312).OR.
     +    (INDEX.EQ.112).OR.(INDEX.EQ.222).OR.(INDEX.EQ.332) ) THEN
         DISPL=0.0D0
         RETURN
      ELSE
C
               DZERO=0.D0
C     NMAX = NRAY; Maximum number of rays. The same as DIM RAYTIME
      NMAX = NRAY
      IF(T.GT.(RAYTIME(NRAY,3))) THEN
         WRITE(6,*)'**************************************'
         WRITE(6,*)'ERROR IN SUBROUTINE GREEN'
            WRITE(6,*)'T > TARRIVAL(NRAY), INCREASE NRAY.'
         WRITE(6,*)'PROGRAM IS PAUSED'
         WRITE(6,*)'**************************************'
         PAUSE 'GREENSUB'
      ENDIF
C
      DT=DBLE(T)
C
C
         DSURF = 0.D0
      IF (KASE.EQ.1) THEN
         CALL SFWAVE(ALPHA,XD,T,INDEX,DSURF)
      ELSE IF (KASE.EQ.2)THEN
         DSURF = 0.D0
      END IF
         DISPL=0.D0
C
         DO 8000 N=1,NMAX
            IF (T.LE.RAYTIME(N,3)) GO TO 8001
            N1 = RAYTIME(N,1)
            N2 = RAYTIME(N,2)
C
            IF((N1.EQ.0).AND.(N2.EQ.0)) GOTO 8000
            IF((N1.LT.0).AND.(N2.EQ.0)) GOTO 8000
C***        IT IS RAYLEIGH WAVE ARRIVAL WHICH HAS BEEN TAKEN
C           CARE OF BY SUBROUTINE SFWAVE
            IF(N2.LT.0) THEN
C***           IT IS A HEAD WAVE
               N2=-N2
               DK2=DBLE(REAL(N2))
               DTSARR = DSQRT(DXD*DXD + DK2*DK2)
               IF (DT.GE.DTSARR) GO TO 8000
               DHEAD = (DT/DA)-DK2*DSQRT(1.0D0-DA*DA)/DA
               DRAYI=DINTG(-DXD,-DHEAD)
            ELSE
C***           IT IS A REGULAR RAY
               DRAYI=DINTG(DZERO,-DXD)
            END IF
C
            DISPL = DISPL + DRAYI
C8000    Y=YZERO   ;set try root eq. 0
 8000    Y=YZERO
```

```
 8001   DISPL=DISPL/(6.28318530717959D0)
          DISPL=DISPL+DSURF
      RETURN
      END IF
      END
C*******************************************************************
C
      DOUBLE PRECISION FUNCTION DINTG(DINIT,DEND)
C
C*******************************************************************
C
      IMPLICIT DOUBLE PRECISION (D)
C
C
      DOUBLE PRECISION EPS,R,E,W(50,6),FMIN,FMAX,F
      INTEGER NINT,NMAX,KF,IFLAG
      LOGICAL RST
C
C
      EXTERNAL F
C
      EPS = 1.D-6
      RST = .FALSE.
      NMAX = 50
         NINT = 1
         W(1,1) = DINIT
         W(2,1) = DEND
C
      CALL DGLQ1(F,DINIT,DEND,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,KF
     1    ,IFLAG)
C
C
      DINTG=R
      RETURN
      END
C*******************************************************************
C
      DOUBLE PRECISION FUNCTION F(DS)
C
C*******************************************************************
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DOUBLE PRECISION ETA1,ETA2SQ,DD,AA
      DIMENSION YQD(3,3),YRSTAR(3,3),YQS(3,3)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      DK1=DBLE(REAL(N1))
      DK2=DBLE(REAL(N2))
       D=DS
```

```
C
      CALL PHIEQ0(DK1,DK2,DA,DT,DXD,DS,Y,DERR,NROOT)
C      WRITE(6,*)DK1,DK2,DS,Y,DERR,NROOT
       IF (NROOT.EQ.0) THEN
         F=0.D0
C
       ELSE
      CALL COMMON(DS)
      CALL RSTAR(YRSTAR,IDUMMY)
      CALL QD(YQD,IDUMMY)
      CALL QSU(YQS,IDUMMY)
C
      YINTG = YZERO
      DO 200 I=1,2
         DO 100 J=1,2
            YINTG=YINTG+YQD(M,I)*YRSTAR(I,J)*YQS(L,J)
  100    CONTINUE
  200    CONTINUE
C
       IF ((N1.EQ.0).AND.(N2.GT.0))
     1   YINTG=YINTG+YQD(M,3)*YQS(L,3)
C
       YINTG=YINTG*YDPHI/CDSQRT(DCMPLX(DXD*DXD-DS*DS))
      F=DBLE(YINTG)
C
       END IF
      RETURN
      END
C**********************************************************************
C
      SUBROUTINE PHIEQ0(DK1,DK2,DA,DT,DXD,D,YROOT,DERR,NROOT)
C
C**********************************************************************
C
C     GIVEN DK1,DK2,DA,DT,D TO SOLVE PHI(YROOT)=0
      IMPLICIT DOUBLE PRECISION (D)
      IMPLICIT DOUBLE COMPLEX (Y)
      DOUBLE PRECISION T,R,A,C1,C2
      COMMON /PARM1/T,R,A,C1,C2
C
      YOLD=YROOT
         IER=0
         DYREAL=0.D0
         YPHI=DCMPLX(0.D0,0.D0)
      T = DT
      R = D
      A = DA
      C1 = DK1
      C2 = DK2
      EPS=1.E-12
      IEND=100
C
       NROOT=1
      IF (DK2.EQ.0.D0) THEN
C
```

```
      DSTSQ = (DT/DA)*(DT/DA)-DK1*DK1
      DSTAR = DSQRT(DSTSQ)
       IF (D.LT.0.D0)DSTAR=-DSTAR
C
      IF (D.EQ.0.D0) THEN
         DYIMAG=-DK1/(DSTAR*DA)
         YROOT = DCMPLX(0.D0,DYIMAG)
      ELSE IF (D.EQ.DSTAR) THEN
         DYREAL=DT/(DA*DA*DSTAR)
         YROOT = DCMPLX(DYREAL,0.0D0)
      ELSE
         DYREAL = D*DT/(DA*DA*DSTSQ)
        DSMD=DSTSQ-D*D
         IF(DSMD.LT.0.D0) WRITE(6,*) DT,D,DSMD,DSTSQ
         DYIMAG = -(DK1/(DSTSQ*DA))*DSQRT(DSMD)
         YROOT = DCMPLX(DYREAL,DYIMAG)
      END IF
      YOLD = YROOT
     CALL DCRTNI(YROOT,YPHI,YDPHI,YOLD,EPS,IEND,IER)
C
   ELSE IF (DK1.EQ.0.D0) THEN
      DTHARR = DA*DABS(DXD)+DK2*DSQRT(1.0D0-DA*DA)
      DTSARR = DSQRT(DXD*DXD+DK2*DK2)
      DSTSQ = (DT)*(DT)-DK2*DK2
      DSTAR = DSQRT(DSTSQ)
C
     IF (DT.GT.DTSARR) THEN
       IF (D.LT.0.D0) DSTAR=-DSTAR
C
      IF (D.EQ.0.D0) THEN
         DYIMAG=-DK2/DABS(DSTAR)
         YROOT = DCMPLX(0.D0,DYIMAG)
      ELSE IF (D.EQ.DSTAR) THEN
         DYREAL=DT/(DSTAR)
         YROOT = DCMPLX(DYREAL,0.0D0)
      ELSE
         DYREAL = D*DT/(DSTSQ)
         DYIMAG = -(DK2/DSTSQ)*DSQRT(DSTSQ-D*D)
         YROOT = DCMPLX(DYREAL,DYIMAG)
      END IF
     ELSE IF ((DT.LT.DTSARR).AND.(DT.GT.DTHARR)) THEN
      DHEAD = (DT/DA)-DK2*DSQRT(1.0D0-DA*DA)/DA
      IF (D.LT.0.D0)DHEAD=-DHEAD
      IF (D.LT.0.D0)DSTAR=-DSTAR
C
      IF (D.EQ.0.D0) THEN
         DYIMAG=-DK2/DSTAR
         YROOT = DCMPLX(0.D0,DYIMAG)
      ELSE IF (D.EQ.DSTAR) THEN
         DYREAL=DT/(DSTAR)
         YROOT = DCMPLX(DYREAL,0.0D0)
      ELSE IF (DABS(D).GT.DABS(DSTAR)) THEN
        IF (D.LT.0.D0) THEN
         DYREAL = (D*DT-DK2*DSQRT(D*D-DSTSQ))/(DSTSQ)
        ELSE IF (D.GT.0.D0) THEN
```

```
            DYREAL = (D*DT+DK2*DSQRT(D*D-DSTSQ))/(DSTSQ)
           ENDIF
             DYIMAG = 0.D0
            YROOT = DCMPLX(DYREAL,DYIMAG)
         END IF
        END IF
         YOLD = YROOT
       CALL DCRTNI(YROOT,YPHI,YDPHI,YOLD,EPS,IEND,IER)
C
      ELSE IF ((DK1.NE.0.D0).AND.(DK2.NE.0.D0)) THEN
         IF((D.EQ.0.D0).OR.(YOLD.EQ.DCMPLX(0.D0,0.D0))) THEN
            CALL DEQO(DT,DA,DK1,DK2,YROOT,NOROOT)
         ENDIF
         YOLD = YROOT
         CALL DCRTNI(YROOT,YPHI,YDPHI,YOLD,EPS,IEND,IER)
      END IF
       IF(IER.NE.0) NROOT=0
C
C     TEST ROOT TO FIND ERROR
C
      DERR=CDABS(YPHI)
C
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE DEQO(DT,DA,DK1,DK2,Y,NOROOT)
C
C*********************************************************************
C
C     COMPUTE THE ROOT BY SOLVING QUARTIC EQ.
C
      IMPLICIT DOUBLE PRECISION (D)
      IMPLICIT DOUBLE COMPLEX (Y)
      DOUBLE PRECISION B
      DIMENSION DB(5),B(3),DROOT1(2),DROOT2(2)
C
      YF(Y)=-DCMPLX(DT,0.D0)*Y-DCMPLX(DK1,0.D0)*
     1      CDSQRT(Y*Y*DCMPLX(DA*DA,0.D0)-DCMPLX(1.D0,0.D0))
     2      -DCMPLX(DK2,0.D0)*CDSQRT(Y*Y-DCMPLX(1.D0,0.D0))
      DT2 = DT*DT
      DK12 =DK1*DK1
      DK22 = DK2*DK2
      DAA=DA*DA
C
      DB(1)=((DT2-DK12*DAA-DK22)**2 - 4.D0*DK12*DK22*DAA)
      DB(3)=2.D0*((DT2-DK12*DAA-DK22)*(DK12+DK22)
     1      +2.D0*DK12*DK22*(DAA+1.D0))
      DB(5)=(DK12+DK22)**2-4.D0*DK12*DK22
C
      B(1) = DB(1)
      B(2) = DB(3)
      B(3) = DB(5)
         TOL=1.E-20
C     CALL QUADRTIC EQ. TO FIND Y**2
```

```
        CALL QUADRT(B,DROOT1,DROOT2,TOL,NOROOT)
C
      Y1=CDSQRT(DCMPLX(DROOT1(1),DROOT1(2)))
      Y2=CDSQRT(DCMPLX(DROOT2(1),DROOT2(2)))
        IF (DIMAG(Y1).GT.0.D0) Y1=-Y1
        IF (DIMAG(Y2).GT.0.D0) Y2=-Y2
       YF1=YF(Y1)
       YF2=YF(Y2)
        DERR1=CDABS(YF(Y1))
        DERR2=CDABS(YF(Y2))
        IF (DERR1.LT.DERR2) Y=Y1
        IF (DERR1.GE.DERR2) Y=Y2
C
        IF (DIMAG(Y).GE.0.D0) Y=DCONJG(Y)
C
      RETURN
      END
C*******************************************************************
C
      DOUBLE COMPLEX FUNCTION PHIO(Z)
C
C*******************************************************************
C
C    FUNCTION PHIO
C
      DOUBLE PRECISION T,R,A,C1,C2
       DOUBLE COMPLEX Z,YAA,YONE
      COMMON /PARM1/T,R,A,C1,C2
C
      YONE=DCMPLX(1.D0,0.D0)
      YAA=DCMPLX(A*A,0.D0)
C
      PHIO=DCMPLX(R,0.D0)-DCMPLX(T,0.D0)*Z
     1   -DCMPLX(C1,0.D0)*CDSQRT(YAA*Z*Z-YONE)
     2   -DCMPLX(C2,0.D0)*CDSQRT(Z*Z-YONE)
C
      RETURN
      END
C*******************************************************************
C
      DOUBLE COMPLEX FUNCTION DPHIO(Z)
C
C*******************************************************************
C
C    FUNCTION DPHIO
C
C
      DOUBLE PRECISION T,R,A,C1,C2
       DOUBLE COMPLEX Z,YAA,YONE
      COMMON /PARM1/T,R,A,C1,C2
C
      YONE=DCMPLX(1.D0,0.D0)
      YAA=DCMPLX(A*A,0.D0)
C
      DPHIO=-DCMPLX(T,0.D0)
```

```
     1   -DCMPLX(C1,0.D0)*YAA*Z/CDSQRT(YAA*Z*Z-YONE)
     2   -DCMPLX(C2,0.D0)*Z/CDSQRT(Z*Z-YONE)
C
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE COMMON(DS)
C
C*********************************************************************
C
C     TO COMPUTE PARAMETERS DEPENDING ON Y AND TO PUT IN
C     A COMMON BLK2
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DOUBLE PRECISION ETA1,ETA2SQ,DD,AA
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      AA=DA*DA
      DD=DS
       D=DS
      DN1=DBLE(REAL(N1))
      DN2=DBLE(REAL(N2))
      YZERO=DCMPLX(0.D0,0.D0)
      YI=DCMPLX(0.D0,1.0D0)
      YAA=DCMPLX(AA,0.D0)
      YA=DCMPLX(DA,0.D0)
      YONE=DCMPLX(1.0D0,0.0D0)
      YTWO=DCMPLX(2.D0,0.D0)
      YFOUR=DCMPLX(4.0D0,0.D0)
C
      ETA1=DS/DXD
      ETA2SQ=1.D0-ETA1*ETA1
      YETA1 = DCMPLX(ETA1,0.D0)
      YETA2S= DCMPLX(ETA2SQ,0.D0)
      YETA2=CDSQRT(YETA2S)
C
      YD = DCMPLX(DD,0.D0)
      YY=Y*Y
      YP=CDSQRT(YAA*YY-YONE)
      YQ=CDSQRT(YY-YONE)
      YPSQP1=YAA*YY
      YQSQP1=YY
      YQSQM1=YQ*YQ-YONE
      YPSQM1=YP*YP-YONE
C
      YSQRP1=Y*YA
      YSQRQ1=Y
      YDELTA=YQSQM1*YQSQM1+YFOUR*YP*YQ
```

```
      YM=(YQSQM1*YQSQM1-YFOUR*YP*YQ)
      YRO=YM/YDELTA
      YRP=-(YFOUR*YP*YQSQM1)/(YA*YDELTA)
      YRM=(YRO*YRO-YONE)/YRP
C
      YDPHI=YONE/(YD+(DCMPLX(DN1,0.D0)/YP)+(DCMPLX(DN2,0.D0)/YQ))
      RETURN
      END
C*******************************************************************
C
      SUBROUTINE RSTAR(YRSTAR,IDUMMY)
C
C*******************************************************************
C
C      TO COMPUTE RSTAR (JS,J2)   EQ.6.39
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DIMENSION YRSTAR(3,3),KSUM(20)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YRO,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      EXTERNAL KDELTA
C
C
         IDUMMY=0
      YRSTAR(3,3) = YONE
      YRSTAR(1,3) = YZERO
      YRSTAR(2,3) = YZERO
      YRSTAR(3,1) = YZERO
      YRSTAR(3,2) = YZERO
C
      DO 2000 JS=1,2
      DO 1000 J2=1,2
            IF (J2.EQ.1) N1=N1-1
            IF (J2.EQ.2) N2=N2-1
          ISIGN = (-1)**(J2*(N1+N2)+N2)
          CALL COEFF(N1,N2,JS,J2,KD,KU,KL,KSUM)
C       WRITE(7,*) N1,N2,JS,J2,KD,KU,KL,KSUM
            YSUM=YZERO
            DO 100 NM=KL,KU
               YSUM=YSUM+DCMPLX(DBLE(REAL(KSUM(NM+1))),0.D0)*
     1            ((YRO*YRO-YONE)/(YRO*YRO))**NM
  100       CONTINUE
            KD2=KDELTA(1,JS)-KDELTA(1,J2)
C
            YSUM = ( (YRO/YRP)**(KD2) )*YSUM
        RSIGN=REAL(ISIGN)
       DRSIGN=RSIGN
       DKD=DBLE(REAL(KD))
            YRSTAR(JS,J2)=DCMPLX(DRSIGN,0.D0)*(YRO**(N1+N2))
```

```
     1                  *(DCMPLX(DKD,0.D0)+YSUM)
               IF (J2.EQ.1) N1=N1+1
               IF (J2.EQ.2) N2=N2+1
C
1000  CONTINUE
2000  CONTINUE
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE QD(YQD,IDUMMY)
C
C*********************************************************************
C
C     TO COMPUTE QD(I,J) BASED ON EQ. 6.21
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      DIMENSION YQD(3,3)
C
      IDUMMY=0
      DO 100 I=1,3
         DO 90 J=1,3
            YQD(I,J) = YZERO
   90    CONTINUE
  100 CONTINUE
C
      IF(M.EQ.1) THEN
      YQD(1,1)=YETA1*YFOUR*YP*YQ*YQSQP1
     1      /(YSQRP1*YDELTA)
      YQD(1,2)=-YTWO*YETA1*YQ*YQSQM1*YSQRQ1
     1      /(YDELTA)
      YQD(1,3) =- YTWO*YETA2
C
      ELSE IF (M.EQ.2)THEN
      YQD(2,1)=YETA2*YFOUR*YP*YQ*YQSQP1
     1      /(YSQRP1*YDELTA)
      YQD(2,2)=-YTWO*YETA2*YQ*YQSQM1*YSQRQ1
     1      /(YDELTA)
      YQD(2,3)=YTWO*YETA1
C
      ELSE IF (M.EQ.3) THEN
      YQD(3,1)=-YTWO*YP*YQSQM1*YQSQP1
     1      /(YSQRP1*YDELTA)
      YQD(3,2)=-YFOUR*YP*YQ*YSQRQ1
     1      /(YDELTA)
      YQD(3,3)=YZERO
C
```

```
      END IF
      RETURN
      END
C***********************************************************************
C
      INTEGER FUNCTION KDELTA(I,J)
C
C***********************************************************************
C
         IF (I.EQ.J) THEN
            KDELTA=1
         ELSE
            KDELTA=0
         END IF
      RETURN
      END
C***********************************************************************
C
      INTEGER FUNCTION IBINO(M,N)
C
C***********************************************************************
C
      IF (M.EQ.N)GO TO 11
      IF ((M.LT.0).OR.(N.LT.0)) GO TO 10
      K=M-N
      IF (K) 10,11,12
10    IBINO = 0
      RETURN
11    IBINO = 1
      RETURN
12    IF (N.EQ.0)GO TO 11
      IN = 1
      IP = M
      IQ = 1
      DO 20 I=1,K
        IN = IN*IP/IQ
        IP = IP-1
        IQ = IQ +1
20    CONTINUE
       IBINO = IN
      RETURN
      END
C***********************************************************************
C
      INTEGER FUNCTION KD1(N1,N2,JS,J2)
C
C***********************************************************************
C
         EXTERNAL KDELTA
      KD1=KDELTA(0,N1)*KDELTA(2,J2)+KDELTA(0,N2)*KDELTA(1,J2)
      KD1=KD1*KDELTA(JS,J2)
      RETURN
      END
C***********************************************************************
C
```

```
      INTEGER FUNCTION KUPPER(N1,N2,JS,J2)
C
C*********************************************************************
C
         EXTERNAL KDELTA
      K=N2+KDELTA(2,J2)*KDELTA(1,JS)-KDELTA(1,J2)*KDELTA(2,JS)
         IF (N1.LE.K) THEN
            KUPPER=N1
         ELSE
            KUPPER=K
         END IF
      RETURN
      END
C*********************************************************************
C
      INTEGER FUNCTION KLOWER(N1,N2,JS,J2)
C
C*********************************************************************
C
         EXTERNAL KDELTA
      KLOWER=KDELTA(2,J2)+KDELTA(1,J2)*KDELTA(1,JS)
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE COEFF(N1,N2,JS,J2,KD,KU,KL,KSUM)
C
C*********************************************************************
C
C     TO COMPUTE THE COEFFICIENTS FOR THE RSTAR TURM
C
C     INPUTS:
C        N1 = NUMBER OF P TRIPS
C        N2 = NUMBER OF S TRIPS
C        JS = FINAL TRIP TYPE, 1 OR 2
C        J2 = INITIAL TRIP TYPE, 1 OR 2
C
C     OUTPUT:
C        KD = FIRST COEFF
C        KU = UPPER LIMIT IN SUMMATION
C        KL = LOWER LIMIT IN SUMMATION
C        KSUM = COEFF. IN SUMMATION, MUST BE DIMENSIONED IN CALLING
C               PROGRAM TO BE MAX(N1,N2)
C
C--------------------------------------------------------------------
C
C     PROGRAM TESTCOEF
C     DIMENSION KSUM(20)
CC***
C   1  WRITE(5,10)
C  10 FORMAT (2X,'TYPE K1,K2,JS,J2')
C     READ(5,*)I,J,JS,J2
C     WRITE(6,30)I,J,JS,J2
C  30 FORMAT(2X,' I =',I5,' J=',I5,' JS=',I2,' J2=',I2)
CC***
```

```
      END IF
      RETURN
      END
C*********************************************************************
C
      INTEGER FUNCTION KDELTA(I,J)
C
C*********************************************************************
C
         IF (I.EQ.J) THEN
            KDELTA=1
         ELSE
            KDELTA=0
         END IF
      RETURN
      END
C*********************************************************************
C
      INTEGER FUNCTION IBINO(M,N)
C
C*********************************************************************
C
      IF (M.EQ.N)GO TO 11
      IF ((M.LT.0).OR.(N.LT.0)) GO TO 10
      K=M-N
      IF (K) 10,11,12
10    IBINO = 0
      RETURN
11    IBINO = 1
      RETURN
12    IF (N.EQ.0)GO TO 11
      IN = 1
      IP = M
      IQ = 1
      DO 20 I=1,K
        IN = IN*IP/IQ
        IP = IP-1
        IQ = IQ +1
20    CONTINUE
       IBINO = IN
      RETURN
      END
C*********************************************************************
C
      INTEGER FUNCTION KD1(N1,N2,JS,J2)
C
C*********************************************************************
C
         EXTERNAL KDELTA
      KD1=KDELTA(0,N1)*KDELTA(2,J2)+KDELTA(0,N2)*KDELTA(1,J2)
      KD1=KD1*KDELTA(JS,J2)
      RETURN
      END
C*********************************************************************
C
```

```
      INTEGER FUNCTION KUPPER(N1,N2,JS,J2)
C
C***********************************************************************
C
         EXTERNAL KDELTA
      K=N2+KDELTA(2,J2)*KDELTA(1,JS)-KDELTA(1,J2)*KDELTA(2,JS)
         IF (N1.LE.K) THEN
            KUPPER=N1
         ELSE
            KUPPER=K
         END IF
      RETURN
      END
C***********************************************************************
C
      INTEGER FUNCTION KLOWER(N1,N2,JS,J2)
C
C***********************************************************************
C
         EXTERNAL KDELTA
      KLOWER=KDELTA(2,J2)+KDELTA(1,J2)*KDELTA(1,JS)
      RETURN
      END
C***********************************************************************
C
      SUBROUTINE COEFF(N1,N2,JS,J2,KD,KU,KL,KSUM)
C
C***********************************************************************
C
C     TO COMPUTE THE COEFFICIENTS FOR THE RSTAR TURM
C
C     INPUTS:
C        N1 = NUMBER OF P TRIPS
C        N2 = NUMBER OF S TRIPS
C        JS = FINAL TRIP TYPE, 1 OR 2
C        J2 = INITIAL TRIP TYPE, 1 OR 2
C
C     OUTPUT:
C        KD = FIRST COEFF
C        KU = UPPER LIMIT IN SUMMATION
C        KL = LOWER LIMIT IN SUMMATION
C        KSUM = COEFF. IN SUMMATION, MUST BE DIMENSIONED IN CALLING
C               PROGRAM TO BE MAX(N1,N2)
C
C-----------------------------------------------------------------------
C
C     PROGRAM TESTCOEF
C     DIMENSION KSUM(20)
CC***
C   1  WRITE(5,10)
C  10 FORMAT (2X,'TYPE K1,K2,JS,J2')
C     READ(5,*)I,J,JS,J2
C     WRITE(6,30)I,J,JS,J2
C  30 FORMAT(2X,' I =',I5,' J=',I5,' JS=',I2,' J2=',I2)
CC***
```

```
      IF (KASE.EQ.1) THEN
C        SOURCE AT TOP SURFACE
C
      DO 200 I=1,3
         YQS(I,1)=YRRP(1,1)*YUP(I,1)*YP1+YRRP(1,2)*YUP(I,2)*YP2
     1         - YUM(I,1)*YP1
         YQS(I,2)=YRRP(2,1)*YUP(I,1)*YP1
     1         +YRRP(2,2)*YUP(I,2)*YP2 - YUM(I,2)*YP2
         YQS(I,3)=(YRRP(3,3)*YUP(I,3) - YUM(I,3))*YP2
  200    CONTINUE
C
C
      ELSE IF (KASE.EQ.2) THEN
C        SOURCE AT BOTTOM SURFACE
C
      DO 300 I=1,3
         YQS(I,1)=YRRM(1,1)*YUM(I,1)*YP1+YRRM(1,2)*YUM(I,2)*YP2
     1         -YUP(I,1)*YP1
         YQS(I,2)=YRRM(2,1)*YUM(I,1)*YP1
     1         +YRRM(2,2)*YUM(I,2)*YP2 - YUP(I,2)*YP2
         YQS(I,3)=(YRRM(3,3)*YUM(I,3) - YUP(I,3) )*YP2
  300    CONTINUE
C
      ENDIF
      RETURN
      END
C  *****************************************************************
C
      SUBROUTINE EPIDIS(A,T,U)
C
C  *****************************************************************
C
C
C       THIS PROGRAM CALCULATES THE VERTICAL DISPLACEMENT
C       AT THE EPICENTER OF A LARGE PLATE DUE TO AN UNIT
C       STEP FUNCTION VERTICAL FORCE.
C       DOUBLE PRECISION IS USED.
C
C  INPUT PARAMETERS :
C
C  A = SHEAR WAVE SPEED / LONGITUDINAL WAVE SPEED
C       e.g. ALUMINUM :  A=0.495
C  T = NORMALIZED TIME = ACTUAL TIME *SHEAR WAVE SPEED / PLATE
C                 THICKNESS
C      e.g. SHEAR WAVE SPEED ~ 3 mm /micro second
C
C  OUTPUT PARAMETERS :
C
C  U = NORMALIZED DISPLACEMENT = ACTUAL DISPLACEMENT * 3.14159
C     * SHEAR MODULUS * PLATE THICKNESS / FORCE
C
C  SUBROUTINES REQUIRED:
C
C  QPARA(T,A), AND FUNCTION BINO
C
```

```
C-------------------------------------------------------------------
C
C   THE PROGRAM MAY BE SET UP AS THE FOLLOWINGS:
C
C-     PROGRAM EPICT
C-     DOUBLE PRECISION A,DT,DELTAT,DU(1024)
C-     A = 0.584D0
C-     NPT=1024
C-     DELTAT=0.00277D0
C-     DO 100 I=1,NPT
C-        DT=DELTAT*DBLE(REAL(I))
C-        CALL EPIDIS(A,DT,DU(I))
C-100  CONTINUE
C-     WRITE(7) DU
C-     STOP
C-     END
C
C-------------------------------------------------------------------
C
      IMPLICIT DOUBLE PRECISION (Q)
      DOUBLE PRECISION U,T,A,TA,DXM
      INTEGER BINO
      COMMON /BLOCK/ QN1,QN2,QO,QP,QM,QPP,QPS,QSP,QSS,QPHI
C
C     NRAY = MAX. POSSIBLE NUMBER OF P RAYS
C     NRAY2 = MAX. POSSIBLE NUMBER OF S RAYS
      NRAY = 20
      NRAY2 = 12
         U = 0.0
       DO 300 NA = 1,NRAY
        DO 200 NB = 1,NRAY2
          N1 = NA-1
          N2 = NB-1
C          N1+N2 MUST BE AN ODD NUMBER
          IF ((FLOAT(N1+N2)/2.).EQ.FLOAT((N1+N2)/2))GO TO 200
          TA = N2 + N1*A
          IF (T.LE.TA) GO TO 200
           QN1 = N1
           QN2 = N2
C
          CALL QPARA(T,A)
C
          NTERM = N2
          IF (N1.LT.N2) NTERM=N1
          NTERM=NTERM+1
          QSUM = 0.0
          DO 100 MT = 1,NTERM
            MM = MT-1
            QFPP = BINO(N1+1,MM+2)*BINO(N2+1,MM+1)
            QFPS = BINO(N2+1,MM+1)*BINO(N1+1,MM+1)
            QFSS = BINO(N2+1,MM+2)*BINO(N1+1,MM+1)
            MO = N1 + N2 - 1 - 2*MM
C
             QSUM = ((-1.)**N2)*
     1      ((QFPP*QPP-QFSS*QSS)*(DXM(QO,MO)*DXM((QM*QP*(-1.)),MM))
```

```
     2     +QFPS*(QPS*DXM(QO,(MO+1))*DXM(((-1.)*QP),MM)*DXM(QM,MM)
     3     -QSP*DXM(QO,(MO+1))*DXM(((-1.)*QM),MM)*DXM(QP,MM)))
     4     +QSUM
C
  100    CONTINUE
          U = U + QSUM/QPHI
  200   CONTINUE
  300  CONTINUE
      RETURN
      END
C
C  *****************************************************************
C
      SUBROUTINE QPARA(T,A)
C
C  *****************************************************************
C
      IMPLICIT DOUBLE PRECISION (Q)
      DOUBLE PRECISION T,A,B,AC,BAC,XX,X,SX,SAX
      COMMON /BLOCK/ QN1,QN2,QO,QP,QM,QPP,QPS,QSP,QSS,QPHI
C
C CALCULATE X
C
      Q1=QN1*QN1
      Q2=QN2*QN2
      IF(QN1.EQ.0.D0) THEN
         XX=(T*T-Q2)/Q2
      ELSE IF (QN2.EQ.0.D0) THEN
         XX=(T*T-Q1*A*A)/Q1
      ELSE
      B = ((Q1+Q2)*T*T -(Q1*A*A-Q2)*(Q1-Q2))/((Q1-Q2)**2)
      AC= ((Q1*A*A-Q2)**2+T*T*(T*T-2.*Q1*A*A-2.*Q2))
     1    /((Q1-Q2)**2)
      BAC =B*B -AC
      IF (BAC.LE.0.0) BAC = 0.0
      XX = B-DSQRT(BAC)
      END IF
      X = DSQRT(XX)
      SX = DSQRT(1.+XX)
      SAX = DSQRT(A*A+XX)
C
C  CALCULATE R'S, Q'S, AND QPHI
C
      QA= (1.+2.*XX)**2
      QB= 4.*SAX*SX*XX
      QO= (QA+QB)/(QA-QB)
      QP= -4.*X*A*SX*(1.+2.*XX)/(QA-QB)
      QM=  4.*X*SAX*(1.+2.*XX)/((QA-QB)*A)
      QPP= 2.*SAX*QA/((QA-QB)**2)
      QPS= SAX/(QA-QB)
      QSP=-SAX/(QA-QB)
      QSS= -2.*SAX*QB/((QA-QB)**2)
      QPHI= 2.*(QN1/SAX + QN2/SX)
      RETURN
      END
```

```
C
C   ******************************************************************
C
      INTEGER FUNCTION BINO(J1,J2)
C
C   ******************************************************************
C
      M = J1-2
      N = J2-2
      IF ((M.EQ.N).OR.(N.EQ.0)) THEN
         BINO=1
         RETURN
      ELSE IF ((M.LT.0).OR.(N.LT.0).OR.(M.LT.N)) THEN
         BINO = 0
         RETURN
      ELSE
         K=M-N
         IN = 1
         IP = M
         IQ = 1
         DO 20 I=1,K
            IN = IN*IP/IQ
            IP = IP-1
            IQ = IQ +1
20       CONTINUE
          BINO = IN
         RETURN
      ENDIF
      END
C*********************************************************************
C
      FUNCTION DXM(D,M)
C
C*********************************************************************
C
C     THE FUNCTION DXM(D,M) IS
C THE SAME AS D**M. BOTH D AND DXM MUST BE DECLARED IN
C THE MAIN PROGRAM AS DOUBLE PRECISION VARIABLES.
C
      DOUBLE PRECISION DXM,D
      IF (M.EQ.0) DXM=1.0D0
      IF (M.GT.0) GO TO 11
      IF (M.LT.0) GO TO 22
      RETURN
C
11    DXM=1.0D0
      DO 10 I=1,M
      DXM=DXM*D
10    CONTINUE
      RETURN
C
22    DXM=1.0D0
      N=-M
      DO 20 I=1,N
      DXM=DXM/D
```

```
20    CONTINUE
      RETURN
C
      END
C*********************************************************************
C
      SUBROUTINE SFWAVE(ALPHA,XR,TTIME,INDEX,DISPL)
C
C*********************************************************************
C          SFWAVE
C     SUBROUTINE TO COMPUTE THE SURFACE DISPLACEMENT
C     INPUTS:
C        ALPHA=RATIO OF SHEAR WAVE SPEED TO LONG. WAVE SPEED
C        XR   =DISTANCE BETWEEN SOURCE AND DETECTOR
C              =ACTUAL DIST./H
C        TTIME =NONDIMENSIONALIZED TTIME
C            =ACTUAL TIME*SHEAR WAVE SPEED / H
C        INDEX=SUBSCRIPT OF THE GREEN'S FUNCTION
C             e.g.  11,22,33,12,13 or 131,113, etc.
C
C     OUTPUT:
C        DISPL=NONDIMENSIONALIZED DISPLACEMENT
C              =ACTUAL DISPL.* PI * SHEAR MODULUS * H / FORCE
C        NOTE:
C        THE PARAMETERS IN COMMON BLKS 0,1, & 5 SHOULD BE ENTERED
C        BEFORE THE FIRST CALL OF THIS SUBROUTINE,
C        I. E.   CALL INIT(ALPHA,XR,ZD,INDEX) FIRST.
C
C     SUBROUITNES REQUIRED:
C
C        DSFINT,DQR,QU,COMMON,RAYRT & INTEGRATION ROUTINES
C
C     REMARK: THE COMPUTATION IS DONE IN Y-PLANE, 4tn QUARD. ON A
C             THREE LEGGED U-PATH TO REPLACE THE INTEGRATION FROM
C             -V TO V.
C
C
C
      IMPLICIT DOUBLE PRECISION (D)
      IMPLICIT DOUBLE COMPLEX (Y)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK5/YR,Y1,Y2
C
      N1=0
      N2=0
      DYR=DBLE(YR)
      DT=DBLE(TTIME)
      DZERO=0.0D0
      DMONEH=-0.5D0
      DSQRYR=DSQRT(DYR)
         RWSPD=DSQRYR
C
      DPF=1.2D0/DBLE(ALPHA)
      DV=DXD/DT
```

```
         IF (TTIME.LE.(ALPHA*XR)) THEN
            DISPL=0.0D0
            RETURN
      ELSE
            DISPL=0.D0
            DISPL=DISPL+DSFING(DZERO,DMONEH,1)
     2                 +DSFING(DV,DPF,2)
     3                 +DSFING(DMONEH,DZERO,3)
      END IF
      END
C*********************************************************************
C
      DOUBLE PRECISION FUNCTION DSFING(DINIT,DEND,IPATH)
C
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (D)
C
      DOUBLE PRECISION EPS,R,E,W(50,6),FMIN,FMAX,F
      INTEGER NINT,NMAX,KF,IFLAG
      LOGICAL RST
C
C
      EXTERNAL DQR1,DQR2,DQR3
C
      EPS = 1.D-8
      RST = .FALSE.
      NMAX = 50
         NINT = 1
         W(1,1) = DINIT
         W(2,1) = DEND
      IF (IPATH.EQ.1) THEN
         CALL DGLQ1(DQR1,DINIT,DEND,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,
     +      KF,IFLAG)
      ELSE IF (IPATH.EQ.2)THEN
         CALL DGLQ1(DQR2,DINIT,DEND,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,
     +      KF,IFLAG)
      ELSE IF (IPATH.EQ.3) THEN
         CALL DGLQ1(DQR3,DINIT,DEND,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,
     +      KF,IFLAG)
      ELSE
      PAUSE 'DSFING: IPATH .NE. 1,2, OR 3'
      END IF
C
      D2PEI=8.D0*DASIN(1.D0)
      DSFING=-R/D2PEI
      RETURN
      END
C*********************************************************************
C
      DOUBLE PRECISION FUNCTION DQR1(DL)
C
C*********************************************************************
      IMPLICIT DOUBLE PRECISION (D)
      EXTERNAL DQR
```

```
      DQR1=DQR(DL,1)
      RETURN
      END
C*********************************************************************
C
      DOUBLE PRECISION FUNCTION DQR2(DL)
C
C*********************************************************************
      IMPLICIT DOUBLE PRECISION (D)
      EXTERNAL DQR
      DQR2=DQR(DL,2)
      RETURN
      END
C*********************************************************************
C
      DOUBLE PRECISION FUNCTION DQR3(DL)
C
C*********************************************************************
      IMPLICIT DOUBLE PRECISION (D)
      EXTERNAL DQR
      DQR3=DQR(DL,3)
      RETURN
      END
C*********************************************************************
C
      DOUBLE PRECISION FUNCTION DQR(DL,IPATH)
C
C*********************************************************************
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DIMENSION YQD(3,3),YUP(3,3),YQK(3),YFEE(3)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
C
      DV=DXD/DT
      IF (IPATH.EQ.1) THEN
         Y= DCMPLX( DV,DL)
      ELSE IF (IPATH.EQ.2) THEN
         Y= DCMPLX(DL,-0.5D0)
      ELSE IF (IPATH.EQ.3) THEN
         DPF=1.2D0/DA
         Y= DCMPLX(DPF,DL)
      ELSE
         PAUSE 'DRQ: IPATH ?'
      ENDIF
      CALL SFCOM(DL)
      CALL QD(YQD,IDUMMY)
      CALL QU(YUP,IDUMMY)
C
```

```
         YFEE(1)=YAA/(YP*YD)
         YFEE(2)=YONE/(YQ*YD)
         YFEE(3)=YONE/(YQ*YD)
C
      YQR = YZERO
C
      IF (K.EQ.0) THEN
         YQK(1)=YONE
         YQK(2)=YONE
         YQK(3)=YONE
      ELSE IF (K.EQ.1) THEN
         YQK(1)=YETA1/Y
         YQK(2)=YETA1/Y
         YQK(3)=YETA1/Y
      ELSE IF (K.EQ.2) THEN
         YQK(1)=YETA2/Y
         YQK(2)=YETA2/Y
         YQK(3)=YETA2/Y
      ELSE IF (K.EQ.3) THEN
         YQK(1)=-YP/Y
         YQK(2)=-YQ/Y
         YQK(3)=-YQ/Y
      ELSE
         PAUSE ' FUUCTION DQR: WRONG INDEX, K .NE. 0,1,2, OR 3'
      END IF
      DO 100 I=1,3
           YQR=YQR+YQD(M,I)*YUP(L,I)*YQK(I)*YFEE(I)
  100    CONTINUE
C
      IF ((IPATH.EQ.1).OR.(IPATH.EQ.3)) THEN
       YQR=YQR*YI/CDSQRT(DCMPLX(DV*DV,0.D0)-YY)
      ELSE
       YQR=YQR/CDSQRT(DCMPLX(DV*DV,0.D0)-YY)
      END IF
      DQR=DBLE(YQR)
C
      RETURN
      END
C***********************************************************************
C
      SUBROUTINE SFCOM(DL)
C
C***********************************************************************
C
C     TO COMPUTE PARAMETERS DEPENDING ON Y AND TO PUT IN
C     A COMMON BLK2
C
      IMPLICIT DOUBLE COMPLEX (Y)
      IMPLICIT DOUBLE PRECISION (D)
      DOUBLE PRECISION ETA1,ETA2SQ,DD,AA
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
```

```
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      AA=DA*DA
      DV=DXD/DT
      DN1=DBLE(REAL(N1))
      DN2=DBLE(REAL(N2))
      YZERO=DCMPLX(0.D0,0.D0)
      YI=DCMPLX(0.D0,1.0D0)
      YAA=DCMPLX(AA,0.D0)
      YA=DCMPLX(DA,0.D0)
      YONE=DCMPLX(1.0D0,0.0D0)
      YTWO=DCMPLX(2.D0,0.D0)
      YFOUR=DCMPLX(4.0D0,0.D0)
C
      YD=-Y*DCMPLX(DT,0.D0)
      YETA1 = -Y/DCMPLX(DV,0.D0)
      YETA2=CDSQRT(YONE-YETA1*YETA1)
       IF (DBLE(Y).LT.0.D0) YETA2=-YETA2
      YY=Y*Y
      YP=CDSQRT(YAA*YY-YONE)
      YQ=CDSQRT(YY-YONE)
      YPSQP1=YAA*YY
      YQSQP1=YY
      YQSQM1=YQ*YQ-YONE
      YPSQM1=YP*YP-YONE
C
      YSQRP1=Y*YA
      YSQRQ1=Y
      YDELTA=YQSQM1*YQSQM1+YFOUR*YP*YQ
      YM=(YQSQM1*YQSQM1-YFOUR*YP*YQ)
      YR0=YM/YDELTA
      YRP=-(YFOUR*YP*YQSQM1)/(YA*YDELTA)
      YRM=(YR0*YR0-YONE)/YRP
C
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE QU(YUP,IDUMMY)
C
C*********************************************************************
C
C     TO COMPUTE QU(I,J) BASED ON EQ. 6.4
C
      IMPLICIT DOUBLE COMPLEX (Y)
      COMMON /BLK0/DXD,DT,DZ,DA
      COMMON /BLK1/KASE,M,L,K,N1,N2,D,DTHETA
      COMMON /BLK2/YD,Y,YP,YQ,YR0,YRP,YRM,YDELTA,YPSQP1,YQSQP1,
     1    YPSQM1,YQSQM1,YSQRP1,YSQRQ1,YY,YA,YAA,YETA1,YETA2
     2    ,YDPHI
      COMMON /BLK3/YI,YONE,YTWO,YTHREE,YFOUR,YEIGHT,YZERO
C
      DIMENSION YUP(3,3)
C
         IDUMMY=0
```

```
      YUP(1,1)=YETA1/YSQRP1
      YUP(1,2)=-YETA1*YQ/YSQRQ1
      YUP(1,3)=-YETA2
      YUP(2,1)=YETA2/YSQRP1
      YUP(2,2)=-YETA2*YQ/YSQRQ1
      YUP(2,3)=YETA1
      YUP(3,1)=-YP/YSQRP1
      YUP(3,2)=-YONE/YSQRQ1
      YUP(3,3)=YZERO
C
      RETURN
      END
C**********************************************************************
C
      SUBROUTINE RAYRT(YR,YI1,YI2,ALPHA)
C
C**********************************************************************
C
C     TO SOLVE RAYLEIGH EQUITION
C
C     INPUT: ALPHA = SHEAR WAVE SPEED / LONG. WAVE SPEED
C
C OUTPUT: YR,YI1,YI2 ARE THE ROOTS TO
C
C  Y*Y*Y-8*Y*Y+8*(3-2*AA)*Y-16*(1-AA) = 0
C
      DOUBLE COMPLEX YR,YI1,YI2
      DOUBLE PRECISION A,AA,RT1,RT2,RT3
      DIMENSION A(4),RT1(2),RT2(2),RT3(2)
C
       IF((ALPHA.LE.0.).OR.(ALPHA.GE.1.0)) THEN
          WRITE(6,*)'RAYRT: WRONG VALUE OF ALPHA'
          WRITE(6,*) ALPHA
          WRITE(6,*)'RAYRT: COMPUTATION PAUSED'
          PAUSE
       ENDIF
C
      AA=ALPHA*ALPHA
       TOL= 1.E-37
         A(1) = 1.0
         A(2) = -8.0D0
         A(3) = 8.*(3.-2.*AA)
         A(4) = -16.*(1.-AA)
C
      CALL CUBIC(A,RT1,RT2,RT3,TOL,NOROOT)
C
C     IF   (ALPHA.LT.0.567) THEN
C
C       ONLY ONE REAL ROOT
C
C     ELSE
C
C       ALL THREE ROOTS ARE REAL
C
         IF ((RT1(1).LT.RT2(1)).AND.(RT1(1).LT.RT3(1)))THEN
```

```
          YR=DCMPLX(RT1(1),RT1(2))
          YI1=DCMPLX(RT2(1),RT2(2))
          YI2=DCMPLX(RT3(1),RT3(2))
        ELSEIF ((RT2(1).LT.RT1(1)).AND.(RT2(1).LT.RT3(1))) THEN
          YI1=DCMPLX(RT1(1),RT1(2))
          YR =DCMPLX(RT2(1),RT2(2))
          YI2=DCMPLX(RT3(1),RT3(2))
        ELSEIF ((RT3(1).LT.RT1(1)).AND.(RT3(1).LT.RT2(1)))THEN
          YI2=DCMPLX(RT1(1),RT1(2))
          YI1=DCMPLX(RT2(1),RT2(2))
          YR =DCMPLX(RT3(1),RT3(2))
        ENDIF
      RETURN
      END
C*********************************************************************
C
       SUBROUTINE TIMEARRI(A,R,ZD,NRAY,RCN,TA,CN)
C
C*********************************************************************
C
C      9/27/82
C
C
      DIMENSION Z1(4),Z2(4), TA(NRAY),CN(NRAY,3),RCN(NRAY,3)
      DOUBLE COMPLEX YR,YI1,YI2
      ITER=100
      NRAY=NRAY
      I=0
      LM=1
C     Z = ZD+0.5 ; ZD refers to origin at the center of the plate
      Z = ZD+0.5
      Z1(1)=0.0
      Z2(1)=Z
      Z1(2)=Z
      Z2(2)=0.0
      Z1(3)=-Z
      Z2(3)=0.0
      Z1(4)=0.0
      Z2(4)=-Z
C
      IF (Z.EQ.1.0) Z2(1)=0.0
      OREVEN=FLOAT(INT(Z))*0.5
      DO 100 J=1,21
        DO 200 K=1,12
         JJ=J-1
         KK=K-1
          IF ( (K.EQ.1).AND.(J.EQ.1) ) THEN
             IF(Z.EQ.1.0) THEN
C     THREE SURFACE WAVE ARRIVALS:
               I=I+1
               CN(I,1)=-1.
               CN(I,2)=0.
               CN(I,3)=R*A
               TA(I)=CN(I,3)
               I=I+1
```

```
                 CN(I,1)=-1.
                 CN(I,2)=-1.
                 CN(I,3)=R
                 TA(I)=CN(I,3)
                 CALL RAYRT(YR,YI1,YI2,A)
                 I=I+1
                 CN(I,1)=0.
                 CN(I,2)=0.
                 CN(I,3)=R/DSQRT(DBLE(YR))
                 TA(I)=CN(I,3)
              ENDIF
              GOTO 200
           ENDIF
           IF ( (Z.EQ.0.0).OR.(Z.EQ.1.0) ) LM=4
           IF ( ((FLOAT(JJ+KK))/2.0-OREVEN).EQ.FLOAT((JJ+KK)/2) )
     1                                    GO TO 200
          DO 300 LL=1,4,LM
               C1=JJ+Z1(LL)
               C2=KK+Z2(LL)
                IF ( (C1.LT.0.0) .OR. (C2.LT.0.0) ) GO TO 300
                I=I+1
            IF (I.GT.NRAY) GO TO 99
            CALL TARRV (A,C1,C2,R,ITER,ERROR,T)
                IF (ERROR.GT.1.E-6) WRITE (6,603) I,C1,C2,ERROR
                CN(I,1)=C1
                CN(I,2)=C2
                CN(I,3)=T
                   TA(I)=T
                IF (C1.GT.0.)GO TO 300
              APJ=A*C2/SQRT(1.-A*A)
              IF(R.LE.APJ) GO TO 300
            C2=-C2
            CALL TARRV (A,C1,C2,R,ITER,ERROR,T)
                I=I+1
            IF (I.GT.NRAY) GO TO 99
                CN(I,1)=C1
                CN(I,2)=C2
                CN(I,3)=T
                   TA(I)=T
300        CONTINUE
200      CONTINUE
100    CONTINUE
C
   99  CONTINUE
      CALL RSORT(CN,TA,RCN,NRAY,3,0)
C
603   FORMAT(2X,'ERROR IS LARGER THAN 1.E-6 ',I3,3F10.5)
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE TARRV(A,C1,C2,R,ITER,ERR,TARR)
C
C*********************************************************************
C
```

```
      IMPLICIT DOUBLE PRECISION (D)
C
      NUM = 0
      DT1=0.157079632679490D1
      DT2=0.0D0
      DA=DBLE(A)
      DAA=DA*DA
      DN1=DBLE(C1)
      DN2=DBLE(C2)
       DR=DBLE(R)
      IF (C1.EQ.0.0) GO TO 2
      IF (C2.EQ.0.0) GO TO 3
C
    1 CONTINUE
      NUM = NUM + 1
      DT=(DT1+DT2)/2.0D0
        DAASS=DAA*DSIN(DT)*DSIN(DT)
        DAASS=1.0D0-DAASS
       DCS2=DSQRT(DAASS)
      DRT = DN1*DTAN(DT) + DN2*DA*DSIN(DT)/DCS2
C     WRITE (6,20) NUM,DT,DRT,C1,C2
      IF (NUM.GE.ITER) GO TO 10
      IF (DABS(DRT-DR).LE.5.D-11) GO TO 10
      IF (DRT.LE.DR) DT2=DT
      IF (DRT.GT.DR) DT1=DT
      GO TO 1
C
C 20   FORMAT ('NUM= ',I3,'  T= ',D17.10,'  RT= ',D17.10,2(2X,F4.0))
10    ITER=NUM
      DTARR=DA*DN1/DCOS(DT) + DN2/DCS2
      DERR=DABS(DR-DRT)
      ERR=DERR
      TARR=DTARR
      RETURN
C
    2 IF (C2.LT.0.0) GO TO 4
      DTARR=DSQRT(DR*DR + DN2*DN2)
      TARR=DTARR
      ERR=0.0
      RETURN
C
    3 DTARR=DSQRT(DR*DR+DN1*DN1)*DA
      TARR=DTARR
      ERR=0.0
      RETURN
C
    4 DTARR=DR*DA - DN2*DSQRT(1.D0-DAA)
C            NOTE  HERE DN2 < 0
      TARR=DTARR
      ERR=0.0
      RETURN
C
      END
C*********************************************************************
C
```

```
      SUBROUTINE CUBIC(A,RT1,RT2,RT3,TOL,NOROOT)
C
C*********************************************************************
C
C     THIS SUBROUTINE FINDS THE ROOTS OF A CUBIC EQUITION
C
C     INPUTS:
C        A     COEFFICIENTS, A(4)
C
C     OUTPUTS:
C        RT1,RT2,RT3  THREE ROOTS
C           RT1(1)=REAL PART OF RT1, ETC.
C
C      THE ROUTINE CALLS LINCNG,AND QUADRT
C
      DOUBLE PRECISION A,B,C,RT1,RT2,RT3,ZERO,X,Y,ZZ,SQT3,RC27,ANG
      DIMENSION A(4),RT1(2),RT2(2),RT3(2),B(2),C(4)
C
      ZERO=TOL/10.
      SQT3=DSQRT(3.0D0)
      RC27=1.0/27.0
C
      IF (DABS(A(1))-ZERO) 7,7,12
    7    CALL QUADRT (A(2),RT1,RT2,TOL,NOROOT)
         RETURN
C
   12 NOROOT=3
      IT=0
C
      DO 1 I=2,4
         A(I)=A(I)/A(1)
    1 CONTINUE
      A(1)=1.0
C
      IF (DABS(A(2)).LE.ZERO) GOTO 2
         NDA=3
      B(1)=1.0
      B(2)=-(A(2)/3.0)
         CALL LINCNG(A,NDA,B,C)
      IT=1
      GO TO 4
    2 DO 3 I=1,4
         C(I)=A(I)
    3 CONTINUE
C
    4 X = C(4)*C(4)*0.25+(C(3)**3)*RC27
         IF (X.LT.0.0) GO TO 9
      X = DSQRT(X)
      Y = -(C(4)*0.5)
      I=1
      RT1(I) =Y+X
C
    5 N = 0
      IF(RT1(I).LT.0.0) N=1
      IF (DABS(RT1(I)).LE.ZERO) GO TO 6
```

```
         RT1(I) = DABS(RT1(I))**(1./3.)
      IF (N.EQ.1) RT1(I) = -RT1(I)
    6 IF (I.EQ.2) GOTO 8
      I = 2
      RT1(I) = Y-X
         GO TO 5
C
    8 RT2(2) = ((RT1(1) - RT1(2))*0.5)*SQT3
         RT1(1) = RT1(1) + RT1(2)
         RT1(2) = 0.0
         RT2(1) = -RT1(1)*0.5
         RT3(1) = RT2(1)
         RT3(2) = -RT2(2)
         GO TO 11
C
    9 ZZ = DABS(C(3))
         X = -(C(4)*0.5)/DSQRT((ZZ**3)*RC27  )
         ANG = DACOS(X)
         Y = 2.0*(DSQRT(ZZ/3.0))
         ANG=ANG/3.0
            RT1(1) = Y*DCOS(ANG)
            RT2(1) = Y*DCOS(ANG + 2.0943951024D0)
            RT3(1) = Y*DCOS(ANG + 4.1887902047D0)
            RT1(2) = 0.0
            RT2(2) = 0.0
            RT3(2) = 0.0
C
   11 IF (IT.EQ.1) THEN
         RT1(1) = RT1(1) + B(2)
         RT2(1) = RT2(1) + B(2)
         RT3(1) = RT3(1) + B(2)
      ELSE
      ENDIF
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE LINCNG(A,NDA,B,C)
C
C*********************************************************************
C
C     TO MAKE A LINEAR CHANGE OF VARIABLES
C     IN A GIVEN POLYNOMIAL
C
      DOUBLE PRECISION A,B,C,BIPWR
      DIMENSION A(1),B(2),C(1)
C
      BIPWR = 1.0D0
      NZ = NDA +1
C
      DO 4 I=1,NZ
         C(I) = A(I)
    4 CONTINUE
C
    6 DO 8 I=2,NZ
```

```
         C(I) = C(I) + C(I-1)*B(2)
    8 CONTINUE
C
      C(NZ) = C(NZ)*BIPWR
      NZ = NZ - 1
      BIPWR = BIPWR*B(1)
      IF (NZ.GT.1) GO TO 6
      C(1) = C(1)*BIPWR
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE DGLQ1(F,A,B,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,KF,IFLAG)
C
C*********************************************************************
C
C
C        ***  ALL REAL VARIABLES ARE TYPED DOUBLE PRECISION
C        ***          9/20/82    NH
C
C
C      DGLQ1 IS A SUBROUTINE FOR THE AUTOMATIC EVALUATION
C            OF DEFINITE INTEGRALS OF A USER DEFINED FUNCTION
C            OF ONE VARIABLE PROVIDING FLEXIBLE USAGE.
C
C            FOR AN EASY TO USE VERSION SEE SUBROUTINE DGLQ.
C
C            CAPABILITIES OF DGLQ1 (IN ADDITION TO THOSE OF DGLQ)
C            INCLUDE:
C               ABILITY TO RESTART A CALCULATION TO GREATER
C                 ACCURACY WITHOUT PENALTY...
C               ABILITY TO SPECIFY AN INITIAL PARTITION OF
C                 THE INTEGRATION INTERVAL...
C               ABILITY TO INCREASE THE WORK SPACE TO HANDLE
C                 MORE DIFFICULT PROBLEMS...
C               OUTPUT OF LARGEST/SMALLEST INTEGRAND VALUE FOR
C                 APPLICATIONS SUCH AS GRAPHING...
C
C      A R G U M E N T S   I N   T H E   C A L L   S E Q U E N C E
C
C      F      (INPUT) THE NAME OF YOUR INTEGRAND FUNCTION.
C                     THIS NAME MUST APPEAR IN AN EXTERNAL STATEMENT
C                     IN ANY PROGRAM WHICH CALLS DGLQ1.
C                     YOU MUST WRITE F IN THE FORM
C                          FUNCTION F(X)
C                             F=(EVALUATE INTEGRAND AT THE POINT X)
C                             RETURN
C                          END
C      A
C      B       (INPUT) ENDPOINTS OF INTEGRATION INTERVAL
C      EPS     (INPUT)  ACCURACY TO WHICH THE INTEGRAL IS TO BE CALCULA
C                       DGLQ1 WILL TRY TO ACHIEVE RELATIVE ACCURACY,
C                       E.G. SET EPS=.01 FOR 2 DIGITS, .001 FOR 3, ET
C      R       (OUTPUT) THE ESTIMATE OF THE INTEGRAL
C      E       (OUTPUT) THE ESTIMATE OF THE ABSOLUTE ERROR IN R.
```

```
C       NINT    (INPUT                                                  GR017600
C                OUTPUT)                                                GR017610
C                        AS AN OUTPUT QUANTITY, NINT GIVES THE          GR017620
C                          NUMBER OF SUBINTERVALS IN THE FINAL          GR017630
C                          PARTITION OF [A,B].                          GR017640
C                        AS AN INPUT QUANTITY, NINT MUST BE SET TO      GR017650
C                          THE NUMBER OF SUBINTERVALS IN THE INITIAL    GR017660
C                          PARTITION OF [A,B].  FOR MOST PROBLEMS       GR017670
C                          THIS IS JUST 1, THE INTERVAL [A,B] ITSELF.   GR017680
C                          NINT IS USEFUL IF YOU WOULD LIKE TO HELP     GR017690
C                          DGLQ1 LOCATE A DIFFICULT SPOT ON [A,B].      GR017700
C                          IN THIS REGARD NINT IS USED ALONG            GR017710
C                          WITH THE ARRAY W (SEE BELOW).  IF YOU SET    GR017720
C                          NINT=1 IT IS NOT NECESSARY TO BE CONCERNED   GR017730
C                          WITH W, EXCEPT THAT IT MUST BE DIMENSIONED...GR017740
C                          AS AN EXAMPLE OF MORE GENERAL APPLICATIONS,  GR017750
C                          IF [A,B]=[0,1] BUT THE INTEGRAND JUMPS AT 0.3GR017760
C                          IT WOULD BE WISE TO SET NINT=2 AND THEN SET  GR017770
C                                  W(1,1)=0.0  (LEFT ENDPOINT)          GR017780
C                                  W(2,1)=0.3  (SINGULAR POINT)         GR017790
C                                  W(3,1)=1.0  (RIGHT ENDPOINT)         GR017800
C                          IF YOU SET NINT GREATER THAN 1, BE SURE TO   GR017810
C                          CHECK THAT YOU HAVE ALSO SET                 GR017820
C                                W(1,1)=A  AND  W(NINT+1,1)=B           GR017830
C       RST     (INPUT) A LOGICAL VARIABLE (E.G. TRUE OR FALSE)         GR017840
C                       SET RST=.FALSE. FOR INITIAL CALL TO DGLQ1       GR017850
C                       SET RST=.TRUE. FOR A SUBSEQUENT CALL,           GR017860
C                           E.G. ONE FOR WHICH MORE ACCURACY IS         GR017870
C                           DESIRED (SMALLER EPS).  A RESTART ONLY      GR017880
C                           MAKES SENSE IF THE PRECEDING CALL RETURNED  GR017890
C                           WITH A VALUE OF IFLAG (SEE BELOW) LESS THAN GR017900
C                           ON A RESTART YOU MAY NOT CHANGE THE VALUES OGR017910
C                           ANY ARGUMENTS IN THE CALL SEQUENCE, EXCEPT EGR017920
C       W(NMAX,6)       W IS AN ARRAY USED FOR SCRATCH STORAGE BY DGLQ1.GR017930
C                        YOU  M U S T  INCLUDE A DIMENSION STATEMENT IN GR017940
C                        YOUR CALLING PROGRAM TO ALLOCATE THIS STORAGE. GR017950
C                        THIS SHOULD BE OF THE FORM                     GR017960
C                                   DIMENSION W(NMAX,6)                 GR017970
C                        WHERE NMAX IS AN INTEGER. AN ADEQUATE VALUE OF GR017980
C                        NMAX IS 50.                                    GR017990
C       NMAX    (INPUT) AN INTEGER EQUAL TO THE FIRST SUBSCRIPT IN THE  GR018000
C                        DIMENSION STATEMENT FOR THE ARRAY W.  THIS IS  GR018010
C                        ALSO EQUAL TO THE MAXIMUM NUMBER OF SUBINTERVALGR018020
C                        PERMITTED IN THE INTERNAL PARTITION OF [A,B].  GR018030
C                        A VALUE OF 50 IS AMPLE FOR MOST PROBLEMS.      GR018040
C       FMIN                                                            GR018050
C       FMAX    (OUTPUT) THE SMALLEST AND LARGEST VALUES OF THE INTEGRANGR018060
C                          WHICH OCCURRED DURING THE CALCULATION.  THE  GR018070
C                          ACTUAL INTEGRAND RANGE ON [A,B] MAY, OF COURSGR018080
C                          BE GREATER BUT PROBABLY NOT BY MORE THAN 10%.GR018090
C       KF      (OUTPUT) THE ACTUAL NUMBER OF INTEGRAND EVALUATIONS USEDGR018100
C                          BY DGLQ1 TO APPROXIMATE THIS INTEGRAL.  KF   GR018110
C                          WILL ALWAYS BE AT LEAST 30.                  GR018120
C       IFLAG   (OUTPUT) TERMINATION FLAG...POSSIBLE VALUES ARE         GR018130
C                  0   NORMAL COMPLETION, E SATISFIES                   GR018140
```

```
C                        E<EPS  AND  E<EPS*DABS(R)                     GR018150
C                 1   NORMAL COMPLETION, E SATISFIES                    GR018160
C                        E<EPS, BUT NOT RELATIVE ERROR REQUEST         GR018170
C                 2   NORMAL COMPLETION, E SATISFIES                    GR018180
C                        E<EPS*DABS(R)                                 GR018190
C                 3   NORMAL COMPLETION BUT EPS WAS TOO SMALL TO        GR018200
C                       SATISFY ABSOLUTE OR RELATIVE ERROR REQUEST.    GR018210
C                 4   ABORTED CALCULATION BECAUSE OF SERIOUS ROUNDING  GR018220
C                       ERROR.  PROBABLY E AND R ARE CONSISTENT.       GR018230
C                 5   ABORTED CALCULATION BECAUSE OF INSUFFICIENT STORAGR018240
C                       R AND E ARE CONSISTENT.  PERHAPS INCREASING NMAGR018250
C                       WILL PRODUCE BETTER RESULTS.                   GR018260
C                 6   ABORTED CALCULATION BECAUSE OF SERIOUS DIFFICULTIGR018270
C                       MEETING YOUR ERROR REQUEST.                    GR018280
C                 7   ABORTED CALCULATION BECAUSE EITHER EPS, NINT OR NGR018290
C                       HAS BEEN SET TO AN ILLEGAL VALUE.              GR018300
C                 8   ABORTED CALCULATION BECAUSE YOU SET NINT>1 BUT FOGR018310
C                       TO SET W(1,1)=A  AND  W(NINT+1,1)=B            GR018320
C                                                                      GR018330
C    T Y P I C A L   P R O B L E M   S E T   U P                       GR018340
C                                                                      GR018350
C     DIMENSION W(50,6)                                                GR018360
C     LOGICAL RST                                                      GR018370
C     EXTERNAL F                                                       GR018380
C     A=0.0                                                            GR018390
C     B=1.0                                                            GR018400
C     W(1,1)=A                                                         GR018410
C     W(2,1)=.3       [SET INTERNAL PARTITION POINT AT .3]             GR018420
C     W(3,1)=B                                                         GR018430
C     NINT=2          [INITIAL PARTITION HAS 2 INTERVALS]              GR018440
C     RST=.FALSE.                                                      GR018450
C     EPS=.001                                                         GR018460
C     NMAX=50                                                          GR018470
C                                                                      GR018480
C     CALL DGLQ1(F,A,B,EPS,R,E,NINT,RST,W,NMAX,FMIN,FMAX,KF,IFLAG)     GR018490
C                                                                      GR018500
C    1        A,B,EPS,R,E,NINT,FMIN,FMAX,KF,IFLAG                      GR018510
C     IF(EPS.EQ. .0001 .OR. IFLAG.GE.3)STOP                            GR018520
C     RST=.TRUE                                                        GR018530
C     EPS=.0001       [ASK FOR ANOTHER DIGIT]                          GR018540
C     GO TO 1                                                          GR018550
C     END                                                              GR018560
C     FUNCTION F(X)                                                    GR018570
C     IF(X.LT. .3)                                                     GR018580
C    1  THEN                                                           GR018590
C         F=X**(0.2D0)*ALOG(X)                                         GR018600
C       ELSE                                                           GR018610
C         F=DSIN(X)                                                    GR018620
C     ENDIF                                                            GR018630
C     RETURN                                                           GR018640
C     END                                                              GR018650
C                                                                      GR018660
C   E N D   O F   D O C U M E N T A T I O N                            GR018670
C                                                                      GR018680
      INTEGER C                                                        GR018690
```

```
      DOUBLE PRECISION A,B,E,EB,EPMACH,EPS,FMAX,FMAXL,FMAXR,FMIN,FMINL
     1             ,FMINR,FMN,FMX,R,RAB,RABS,RAV,SIGN,T,TE
     2            ,TE1,TE2,TR,TR1,TR2,UFLOW,W,XM,F
      DIMENSION W(NMAX,6)
      EXTERNAL F
      LOGICAL RST,DEBUG
      DATA SIGN /-1.D0/
C     THE FOLLOWING DATA ARE FOR PERKIN-ELMER COMPUTERS
      DATA EPMACH,UFLOW/Z3410000000000000,Z0010000000000000/
C     THE FOLLOWING DATA ARE FOR VAX COMPUTERS
C     DATA EPMACH,UFLOW/Z0000250000000000,Z0000008000000000/
C
C     EPMACH = Y'3410000000000000'
C     UFLOW = Y'00100000'
C     EPMACH = DBLE(16**(-13))
C     UFLOW = DBLE(16**(-65))
      IF(A.EQ.B) THEN
          R=0.
          E=0.
          NINT=0
          IFLAG=0
          KF=1
          FMIN=F(A)
          FMAX=FMIN
          GO TO 20
      ENDIF
      IF(RST) THEN
            IF(IFLAG.LT.3) GO TO 15
            GO TO 20
      ENDIF
      KF=0
      IF(EPS .LE. 0. .OR. NMAX .LE. 1 .OR. NINT .LE. 0) THEN
          IFLAG=7
          GO TO 20
      ENDIF
      IF(NINT.EQ.1)  THEN
C    1  THEN
          W(1,1)=A
          W(2,2)=B
          W(1,5)=A
          W(1,6)=B
          W(2,5)=A
          W(2,6)=B
          W(1,2)=A+(B-A)/2.D0
          W(2,1)=A+(B-A)/2.D0
          NINT=2
        ELSE
          IF(W(1,1).NE.A .AND. W(NINT+1,1).NE.B) THEN
               IFLAG=8
               GO TO 20
          ENDIF
          W(1,5)=A
          DO 89 I=1,NINT
             W(I,2)=W(I+1,1)
             W(I,5)=W(I,1)
```

```
           W(I,6)=W(I,2)
 89     CONTINUE
      ENDIF
C
      DEBUG=.FALSE.
      IFLAG = 0
      IROFF=0
      RABS=0.0
      DO 3 I=1,NINT
          CALL GL15T(F,W(I,1),W(I,2),W(I,5),W(I,6),
     1          W(I,4),W(I,3),RAB,RAV,FMN,FMX)
          IF(DEBUG) WRITE(7 ,*)'INITIALIZE',(W(I,J),J=1,6)
          KF=KF+15
          IF(I.EQ.1)
     1       THEN
               R=W(I,4)
               E=W(I,3)
               RABS=RABS+RAB
               FMIN=FMN
               FMAX=FMX
             ELSE
               R=R+W(I,4)
               E=E+W(I,3)
               RABS=RABS+RAB
               FMAX=DMAX1(FMAX,FMX)
               FMIN=DMIN1(FMIN,FMN)
          ENDIF
    3 CONTINUE
      DO 10 I=NINT+1,NMAX
          W(I,3) = 0.
   10 CONTINUE
   15 CONTINUE
C
C   MAIN SUBPROGRAM LOOP
C
      IF(100.D0*EPMACH*RABS.GE.DABS(R) .AND. E.LT.EPS)GO TO 20
      EB=DMAX1(100.D0*UFLOW,DMAX1(EPS,50.D0*EPMACH)*DABS(R))
      IF(E.LE.EB) GO TO 20
      IF (NINT.LT.NMAX)
     1 THEN
        NINT = NINT+1
        C = NINT
       ELSE
        C=0
 16     IF(C.EQ.NMAX) THEN
            IFLAG=5
            GO TO 20
        ENDIF
        C=C+1
        IF(W(C,3).GT.0.D0) GO TO 16
      END IF
C     LOC=ISAMAX(NINT,W(1,3),1)
      CALL LSE(W,NMAX,NINT,LOC)
      IF(DEBUG)WRITE(7,200)LOC,(W(LOC,I),I=1,6),R,E
      XM = W(LOC,1)+(W(LOC,2)-W(LOC,1))/2.
```

```
      IF ((DMAX1(DABS(W(LOC,1)),DABS(W(LOC,2)))).GT.
     1   ((1.D0+100.D0*EPMACH)*(DABS(XM)+0.1D+04*UFLOW)))
     2    THEN
         CALL GL15T(F,W(LOC,1),XM,W(LOC,5),W(LOC,6),
     1              TR1,TE1,RAB,RAV,FMINL,FMAXL)
         KF=KF+15
         IF (TE1.LT.(EB*(XM-W(LOC,1))/(B-A)))
     A      TE1 = TE1*SIGN
         CALL GL15T(F,XM,W(LOC,2),W(LOC,5),W(LOC,6),
     1              TR2,TE2,RAB,RAV,FMINR,FMAXR)
         KF=KF+15
         FMIN=DMIN1(FMIN,FMINL,FMINR)
         FMAX=DMAX1(FMAX,FMAXL,FMAXR)
         IF (TE2.LT.(EB*(W(LOC,2)-XM)/(B-A))) TE2=TE2*SIGN
         TE = DABS(W(LOC,3))
         TR = W(LOC,4)
         W(C,3) = TE2
         W(C,4) = TR2
         W(C,1) = XM
         W(C,2) = W(LOC,2)
         W(C,5) = W(LOC,5)
         W(C,6) = W(LOC,6)
         W(LOC,3) = TE1
         W(LOC,4) = TR1
         W(LOC,2) = XM
         IF(DEBUG)WRITE(7,200)C,(W(C,K),K=1,6)
         IF(DEBUG)WRITE(7,200)LOC,(W(LOC,K),K=1,6)
         E = E-TE+(DABS(TE1)+DABS(TE2))
         R = R-TR+(TR1+TR2)
         IF(DEBUG)WRITE(7 ,*)NINT,R,E
         IF(DABS(DABS(TE1)+DABS(TE2)-TE).LT..001D0*TE) THEN
             IROFF=IROFF+1
             IF(IROFF.GE.10) THEN
                  IFLAG=4
                  GO TO 20
             ENDIF
         ENDIF
       ELSE
         IF (EB.GT.W(LOC,3))
     1      THEN
              W(LOC,3) = 0.
            ELSE
              IFLAG=6
              GO TO 20
         END IF
      END IF
      GO TO 15
C
C        ALL EXITS FROM HERE
C
   20 CONTINUE
      IF(IFLAG.GE.4)RETURN
      IFLAG=3
      T=EPS*DABS(R)
      IF(E.GT.EPS .AND. E.GT.T)RETURN
```

```
      IFLAG=2
      IF(E.GT.EPS .AND. E.LT.T)RETURN
      IFLAG=1
      IF(E.LT.EPS .AND. E.GT.T)RETURN
      IFLAG=0
      RETURN
  200 FORMAT (I4,8(E11.3))
      END
C**********************************************************************
C
      SUBROUTINE LSE (WORK,NMAX,NI,LOC)
C
C**********************************************************************
C
C   THIS SUBPROGRAM FINDS THE CELL IN THE WORK AREA
C     THAT HAS THE LARGEST ABSOLUTE ERROR AND
C     RETURNS THE LOCATION OF THAT CELL.
C
      DOUBLE PRECISION ERROR,WORK
      DIMENSION WORK(NMAX,6)
C
C   INITIALIZE VARIABLES
C
      ERROR = DABS(WORK(1,3))
      LOC = 1
C
C   MAIN SUBPROGRAM LOOP
C
      DO 20 I=1,NI
          IF (DABS(WORK(I,3)).GT.ERROR) THEN
              ERROR = DABS(WORK(I,3))
              LOC = I
          END IF
   20 CONTINUE
      RETURN
      END
C**********************************************************************
C
      SUBROUTINE GL15T(F,A,B,XL,XR,R,AE,RA,
     1                 RASC,FMIN,FMAX)
C
C**********************************************************************
C
C***AUTHORS          ROBERT PIESSENS AND ELISE DE DONCKER
C                    APPL. MATH. AND PROGR. DIV.- K.U.LEUVEN
C                    DAVID KAHANER, NBS WASHINGTON
C     ..................................................................
C           PURPOSE
C              TO COMPUTE I = INTEGRAL OF G(X) OVER (A,B),
C                             WITH ERROR ESTIMATE
C                         J = INTEGRAL OF DABS(G) OVER (A,B)
C
C           PARAMETERS
C            ON ENTRY
C              F      - DOUBLE PRECISION
```

```
C                        FUNCTION SUBPROGRAM DEFINING THE INTEGRAND
C                        FUNCTICN F(X). THE ACTUAL NAME FOR F NEEDS
C                        TO BE DECLARED E X T E R N A L IN THE
C                        CALLING PROGRAM.
C                        THE FUNCTION G(X) IS DEFINED TO BE
C                        G(X)=F(PHI(X))*PHIP(X)
C                        WHERE PHI(X) IS THE CUBIC GIVEN BY
C                        THE ARITHMETIC STATEMENT FUNCTION BELOW.
C                        PHIP(X) IS ITS DERIVATIVE.  THE VARIABLES
C                        XL AND XR ARE THE LEFT AND RIGHT ENDPOINTS
C                        OF A PARENT INTERVAL OF WHICH (A,B) IS A PART.
C
C               A      - DOUBLE PRECISION
C                        LOWER LIMIT OF INTEGRATION
C
C               B      - DOUBLE PRECISION
C                        UPINTG LIMIT OF INTEGRATION
C
C               XL     - DOUBLE PRECISION
C                        LEFT ENDPOINT OF PARENT INTERVAL
C
C               XR     - DOUBLE PRECISION
C                        RIGHT ENDPOINT OF PARENT INTERVAL
C
C             ON RETURN
C               R - DOUBLE PRECISION
C                        APPROXIMATION TO THE INTEGRAL I
C                        R IS COMPUTED BY APPLYING THE 15-POINT
C                        KRONROD RULE (RESK) CBTAINED BY OPTIMAL
C                        ADDITION OF ABSCISSAE TO THE 7-POINT GAUSS
C                        RULE (RESG).
C
C               AE - DOUBLE PRECISION
C                        ESTIMATE OF THE MODULUS OF THE ABSOLUTE ERROR,
C                        WHICH SHOULD NOT EXCEED DABS(I-R)
C
C               RA - DOUBLE PRECISION
C                        APPROXIMATION TO THE INTEGRAL J
C
C               RASC - DOUBLE PRECISION
C                        APPROXIMATION TO THE INTEGRAL OF DABS(G-I/(B-A))
C                        OVER (A,B)
C
C               FMAX, FMIN - DOUBLE PRECISION
C                        MAX AND MIN VALUES OF THE FUNCTION F ON (A,B)
C            SUBROUTINES OR FUNCTIONS NEEDED
C                  - F (USER-PROVIDED FUNCTION)
C                  - R1MACH
C                  - FORTRAN DABS, DMAX1, DMIN1
C
C     .................................................................
C***END PROLOGUE
C
      DOUBLE PRECISION A,AE,B,DHLGTH,EPMACH,F,FC,FSUM,FVAL1,FVAL2,
     *  FV1,FV2,HLGTH,RA,RASC,RESG,RESK,RESKH,R,R1MACH,UFLCW,
```

```
     *  WG,WGK,XGK
      DOUBLE PRECISION XL,XR,CENTR,ABSC,U,FMAX,FMIN,PHI,PHIP
      INTEGER J,JTW,JTWM1
C
      DIMENSION FV1(7),FV2(7),WG(4),WGK(8),XGK(8)
C
C           THE ABSCISSAE AND WEIGHTS ARE GIVEN FOR THE INTERVAL (-1,1)
C           BECAUSE OF SYMMETRY ONLY THE POSITIVE ABSCISSAE AND THEIR
C           CORRESPONDING WEIGHTS ARE GIVEN.
C
C           XGK    - ABSCISSAE OF THE 15-POINT KRONROD RULE
C                    XGK(2), XGK(4), ...  ABSCISSAE OF THE 7-POINT
C                    GAUSS RULE
C                    XGK(1), XGK(3), ...  ABSCISSAE WHICH ARE OPTIMALLY
C                    ADDED TO THE 7-POINT GAUSS RULE
C
C           WGK    - WEIGHTS OF THE 15-POINT KRONROD RULE
C
C           WG     - WEIGHTS OF THE 7-POINT GAUSS RULE
C
      DATA XGK(1),XGK(2),XGK(3),XGK(4),XGK(5),XGK(6),XGK(7),XGK(8)/
     *     0.9914553711208126D+00,   0.9491079123427585D+00,
     *     0.8648644233597691D+00,   0.7415311855993944D+00,
     *     0.5860872354676911D+00,   0.4058451513773972D+00,
     *     0.2077849550078985D+00,   0.0D+00               /
      DATA WGK(1),WGK(2),WGK(3),WGK(4),WGK(5),WGK(6),WGK(7),WGK(8)/
     *     0.2293532201052922D-01,   0.6309209262997855D-01,
     *     0.1047900103222502D+00,   0.1406532597155259D+00,
     *     0.1690047266392679D+00,   0.1903505780647854D+00,
     *     0.2044329400752989D+00,   0.2094821410847278D+00/
      DATA WG(1),WG(2),WG(3),WG(4)/
     *     0.1294849661688697D+00,   0.2797053914892767D+00,
     *     0.3818300505051189D+00,   0.4179591836734694D+00/
      DATA EPMACH,UFLOW/Z3410000000000000,Z0010000000000000/
C
      PHI(U)=XR-(XR-XL)*U*U*(2.D0*U+3.D0)
      PHIP(U)=-6.D0*U*(U+1.D0)
C
C           LIST OF MAJOR VARIABLES
C           -----------------------
C
C           CENTR  - MID POINT OF THE INTERVAL
C           HLGTH  - HALF-LENGTH OF THE INTERVAL
C           ABSC   - ABSCISSA
C           FVAL*  - FUNCTION VALUE
C           RESG   - R OF THE 7-POINT GAUSS FORMULA
C           RESK   - R OF THE 15-POINT KRONROD FORMULA
C           RESKH  - APPROXIMATION TO THE MEAN VALUE OF F OVER (A,B),
C                    I.E. TO I/(B-A)
C
C           MACHINE DEPENDENT CONSTANTS
C           ---------------------------
C
C           EPMACH IS THE LARGEST RELATIVE SPACING.
C           UFLOW IS THE SMALLEST POSITIVE MAGNITUDE.
```

```
C
C***FIRST EXECUTABLE STATEMENT
C     EPMACH = DBLE(16**(-13))
C     UFLOW = DBLE(16**(-65))
C
      HLGTH = 0.5D+00*(B-A)
      CENTR = A+HLGTH
      DHLGTH = DABS(HLGTH)
C
C           COMPUTE THE 15-POINT KRONROD APPROXIMATION TO
C           THE INTEGRAL, AND ESTIMATE THE ABSOLUTE ERROR.
C
      U=(CENTR-XR)/(XR-XL)
      FMIN=F(PHI(U))
      FMAX=FMIN
      FC=FMIN*PHIP(U)
      RESG = FC*WG(4)
      RESK = FC*WGK(8)
      RA = DABS(RESK)
      DO 10 J=1,3
        JTW = J*2
        ABSC = HLGTH*XGK(JTW)
        U=(CENTR-ABSC-XR)/(XR-XL)
        FVAL1 = F(PHI(U))
        FMAX=DMAX1(FMAX,FVAL1)
        FMIN=DMIN1(FMIN,FVAL1)
        FVAL1=FVAL1*PHIP(U)
        U=(CENTR+ABSC-XR)/(XR-XL)
        FVAL2=F(PHI(U))
        FMAX=DMAX1(FMAX,FVAL2)
        FMIN=DMIN1(FMIN,FVAL2)
        FVAL2=FVAL2*PHIP(U)
        FV1(JTW) = FVAL1
        FV2(JTW) = FVAL2
        FSUM = FVAL1+FVAL2
        RESG = RESG+WG(J)*FSUM
        RESK = RESK+WGK(JTW)*FSUM
        RA = RA+WGK(JTW)*(DABS(FVAL1)+DABS(FVAL2))
   10 CONTINUE
      DO 15 J = 1,4
        JTWM1 = J*2-1
        ABSC = HLGTH*XGK(JTWM1)
        U=(CENTR-ABSC-XR)/(XR-XL)
        FVAL1=F(PHI(U))
        FMAX=DMAX1(FMAX,FVAL1)
        FMIN=DMIN1(FMIN,FVAL1)
        FVAL1=FVAL1*PHIP(U)
        U=(CENTR+ABSC-XR)/(XR-XL)
        FVAL2=F(PHI(U))
        FMAX=DMAX1(FMAX,FVAL2)
        FMIN=DMIN1(FMIN,FVAL2)
        FVAL2=FVAL2*PHIP(U)
        FV1(JTWM1) = FVAL1
        FV2(JTWM1) = FVAL2
        FSUM = FVAL1+FVAL2
```

```
        RESK = RESK+WGK(JTWM1)*FSUM
        RA = RA+WGK(JTWM1)*(DABS(FVAL1)+DABS(FVAL2))
   15 CONTINUE
      RESKH = RESK*0.5D+00
      RASC = WGK(8)*DABS(FC-RESKH)
      DO 20 J=1,7
        RASC = RASC+WGK(J)*(DABS(FV1(J)-RESKH)+DABS(FV2(J)-RESKH))
   20 CONTINUE
      R = RESK*HLGTH
      RA = RA*DHLGTH
      RASC = RASC*DHLGTH
      AE = DABS((RESK-RESG)*HLGTH)
      IF(RASC.NE.0.0D+00.AND.AE.NE.0.0D+00)
     *  AE = RASC*DMIN1(0.1D+01,
     *  (0.2D+03*AE/RASC)**1.5D+00)
      IF(RA.GT.UFLOW/(0.5D+02*EPMACH)) AE = DMAX1
     *  ((EPMACH*0.5D+02)*RA,AE)
      RETURN
      END
C**********************************************************************
C
      SUBROUTINE QUADRT(A,ROOT1,ROOT2,TOL,NOROOT)
C
C**********************************************************************
C
C     TO SOLVE ANY QUADRTIC EQUATION OF THE FORM
C
C        A(1)*X**2+A(2)*X+A(3) = 0
C
      DOUBLE PRECISION A,ROOT1,ROOT2,X,Y,Z,W,ZERO
      DIMENSION A(3),ROOT1(2),ROOT2(2)
C
      ZERO = TOL/10.0
C
      IF (DABS(A(1))-ZERO)2,2,1
    2    IF (DABS(A(2))-ZERO) 3,3,4
    3 NOROOT =0
      RETURN
C
    1 NOROOT= 2
         X = A(2)*A(2) - 4.0*A(1)*A(3)
         Y = A(1) + A(1)
         Z = DSQRT(DABS(X))/Y
         W = -A(2)/Y
C
      IF (X.LT.0.0) GO TO 7
         ROOT1(1)=W+Z
         ROOT1(2)=0.0
         ROOT2(1)=W-Z
         ROOT2(2)=0.0
         RETURN
C
    7 ROOT1(1)=W
      ROOT1(2)=Z
      ROOT2(1)=W
```

```
      ROOT2(2)=-Z
      RETURN
C
    4 NOROOT=1
      ROOT1(1) = -A(3)/A(2)
      ROOT1(2) = 0.0
      RETURN
      END
C*******************************************************************
C
      SUBROUTINE DCRTNI(X,F,DERF,XST,EPS,IEND,IER)
C
C*******************************************************************
C
C        PURPOSE
C           TO SOLVE PHI = 0
C           BY MEANS OF NEWTON-S ITERATION METHOD.
C
C        USAGE
C           CALL DCRTNI (X,F,DERF,XST,EPS,IEND,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - DOUBLE COMPLEX RESULTANT ROOT OF EQUATION F(X)=0.
C           F      - DOUBLE COMPLEX RESULTANT FUNCTION VALUE AT
C                    ROOT X.
C           DERF   - DOUBLE COMPLEX RESULTANT VALUE OF DERIVATIVE
C                    AT ROOT X.
C           XST    - DOUBLE COMPLEX INPUT VALUE WHICH SPECIFIES THE
C                    INITIAL GUESS OF THE ROOT X.
C           EPS    - SINGLE PRECISION INPUT VALUE WHICH SPECIFIES THE
C                    UPPER BOUND OF THE ERROR OF RESULT X.
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS
C                     IER=0 - NO ERROR,
C                     IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS,
C                     IER=2 - AT ANY ITERATION STEP DERIVATIVE DERF WAS
C                             EQUAL TO ZERO.
C
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C            PHIO,AND DPHIO
C
C        METHOD
C           SOLUTION OF EQUATION F(X)=0 IS DONE BY MEANS OF NEWTON-S
C           ITERATION METHOD, WHICH STARTS AT THE INITIAL GUESS XST OF
C           A ROOT X. CONVERGENCE IS QUADRATIC IF THE DERIVATIVE OF
C           F(X) AT ROOT X IS NOT EQUAL TO ZERO. ONE ITERATION STEP
C           REQUIRES ONE EVALUATION OF F(X) AND ONE EVALUATION OF THE
C           DERIVATIVE OF F(X).
C           FOR REFERENCE, SEE R. ZURMUEHL, PRAKTISCHE MATHEMATIK FUER
C           INGENIEURE UND PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/
C           HEIDELBERG, 1963, PP.12-17.
C
C
      DOUBLE COMPLEX X,F,DERF,XST,TOL,DX,PHIO,DPHIO
```

```
      DOUBLE PRECISION A,DTOL,DTOLF
      EXTERNAL PHIO,DPHIO
C     PREPARE ITERATION
      IER=0
      X=XST
      TOL=X
      F = PHIO(TOL)
      DERF = DPHIO(TOL)
      DTOLF=100.D0*EPS
C
C
C     START ITERATION LOOP
      DO 6 I=1,IEND
      IF(CDABS(F))1,7,1
C
C     EQUATION IS NOT SATISFIED BY X
    1 IF(CDABS(DERF))2,8,2
C
C     ITERATION IS POSSIBLE
    2 DX=F/DERF
      X=X-DX
      TOL=X
      F = PHIO(TOL)
      DERF = DPHIO(TOL)
C
C     TEST ON SATISFACTORY ACCURACY
      DTOL=DBLE(EPS)
      A=CDABS(X)
      IF(A-1.D0)4,4,3
    3 DTOL=DTOL*A
    4 IF(CDABS(DX)-DTOL)5,5,6
    5 IF(CDABS(F)-DTOLF)7,7,6
    6 CONTINUE
C     END OF ITERATION LOOP
C
C
C     NO CONVERGENCE AFTER IEND ITERATION STEPS. ERROR RETURN.
      IER=1
    7 RETURN
C
C     ERROR RETURN IN CASE OF ZERO DIVISOR
    8 IER=2
      RETURN
      END
C***********************************************************************
C
      SUBROUTINE RSORT(A,B,R,N,M,MS)
C
C***********************************************************************
      DIMENSION A(1),B(1),R(1)
C
C        MOVE SORTING KEY VECTOR TO FIRST COLUMN OF OUTPUT MATRIX
C        AND BUILD ORIGINAL SEQUENCE LIST IN SECOND COLUMN
C
      DO 10 I=1,N
```

```
      R(I)=B(I)
      I2=I+N
   10 R(I2)=I
C
C        SORT ELEMENTS IN SORTING KEY VECTOR (ORIGINAL SEQUENCE LIST
C        IS RESEQUENCED ACCORDINGLY)
C
      L=N+1
   20 ISORT=0
      L=L-1
      IF(L .LT. 2) GO TO 50
      DO 40 I=2,L
      IF(R(I)-R(I-1)) 30,40,40
   30 ISORT=1
      RSAVE=R(I)
      R(I)=R(I-1)
      R(I-1)=RSAVE
      I2=I+N
      SAVER=R(I2)
      R(I2)=R(I2-1)
      R(I2-1)=SAVER
   40 CONTINUE
      IF(ISORT) 20,50,20
C
C        MOVE ROWS FROM MATRIX A TO MATRIX R (NUMBER IN SECOND COLUMN
C        OF R REPRESENTS ROW NUMBER OF MATRIX A TO BE MOVED)
C
   50 DO 80 I=1,N
C
C        GET ROW NUMBER IN MATRIX A
C
      I2=I+N
      IN=R(I2)
C
      IR=I-N
      DO 80 J=1,M
C
C        LOCATE ELEMENT IN OUTPUT MATRIX
C
      IR=IR+N
C
C        LOCATE ELEMENT IN INPUT MATRIX
C
      CALL LOC(IN,J,IA,N,M,MS)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(IA) 60,70,60
C
C        MOVE ELEMENT TO OUTPUT MATRIX
C
   60 R(IR)=A(IA)
      GO TO 80
   70 R(IR)=0
   80 CONTINUE
```

```
      RETURN
      END
C*********************************************************************
C
      SUBROUTINE LOC(I,J,IR,N,M,MS)
C
C*********************************************************************
C
C        PURPOSE
C           COMPUTE A VECTOR SUBSCRIPT FOR AN ELEMENT IN A MATRIX OF
C           SPECIFIED STORAGE MODE
C
C        USAGE
C           CALL LOC (I,J,IR,N,M,MS)
C
C        DESCRIPTION OF PARAMETERS
C           I   - ROW NUMBER OF ELEMENT
C           J   - COLUMN NUMBER  OF ELEMENT
C           IR  - RESULTANT VECTOR SUBSCRIPT
C           N   - NUMBER OF ROWS IN MATRIX
C           M   - NUMBER OF COLUMNS IN MATRIX
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C
C        REMARKS
C           NONE
C           M IS UNUSED BUT IS REQUIRED TO BE PRESENT
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           MS=0   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*M ELEMENTS
C                  IN STORAGE (GENERAL MATRIX)
C           MS=1   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*(N+1)/2 IN
C                  STORAGE (UPPER TRIANGLE OF SYMMETRIC MATRIX). IF
C                  ELEMENT IS IN LOWER TRIANGULAR PORTION, SUBSCRIPT IS
C                  CORRESPONDING ELEMENT IN UPPER TRIANGLE.
C           MS=2   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N ELEMENTS
C                  IN STORAGE (DIAGONAL ELEMENTS OF DIAGONAL MATRIX).
C                  IF ELEMENT IS NOT ON DIAGONAL (AND THEREFORE NOT IN
C                  STORAGE), IR IS SET TO ZERO.
C
C     ..................................................................
C
      IX=I
      JX=J
      IF(MS-1) 10,20,30
   10 IRX=N*(JX-1)+IX
      GO TO 36
   20 IF(IX-JX) 22,24,24
   22 IRX=IX+(JX*JX-JX)/2
      GO TO 36
```

```
24 IRX=JX+(IX*IX-IX)/2                                                  GR025300
   GO TO 36                                                             GR025310
30 IRX=0                                                                GR025320
   IF(IX-JX) 36,32,36                                                   GR025330
32 IRX=IX                                                               GR025340
36 IR=IRX                                                               GR025350
   RETURN                                                               GR025360
   END                                                                  GR025370
```

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR/85-3234	2. Performing Organ. Report No.	3. Publication Date NOVEMBER 1985

4. TITLE AND SUBTITLE

Dynamic Green's Functions of An Infinite Plate - A Computer Program

5. AUTHOR(S)

Nelson N. Hsu

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

This report is a FORTRAN program to compute the Green's functions of an infinite plate. The Green's function, $G_{ij}(\underline{\xi},\underline{x},t)$, is defined as the ith component of the displacement at $\underline{x}$ due to a point force of step-function time dependency acting at $\underline{\xi}$ in the jth direction initiated at t=0. The Green's function is the fundamental solution of the transient elastic wave propagation problem. In general, the displacement field $\underline{u}(\underline{\xi},\underline{x},t)$ at $\underline{x}$ due to a point force of arbitrary time dependence acting at $\underline{\xi}$ can be computed by a convolution integration; i.e.,

$$u_i(\underline{\xi},\underline{x},t) = \int_0^\infty G_{ij}(\underline{\xi},\underline{x},t)f_j(\tau-t)d\tau. \quad (1)$$

Here, G_{ij} is the time derivative of G_{ij} and $f_j(t)$ is the point force component of arbitrary time dependence acting in the jth direction (summation over repeated indices is used). Displacement due to point dipoles or couple forces can be represented by the spatial derivatives of G_{ij}. Displacement produced by a dynamic force distributed over a finite area can also be computed by numerical integration using the Green's function as the kernel of the integral over the finite area.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

Acoustic emission; dynamic displacement due to point impact; Green's function; ray theory; transient wave propagation; waveform analysis

13. AVAILABILITY

[X] Unlimited
[ ] For Official Distribution. Do Not Release to NTIS
[ ] Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
[X] Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

66

15. Price

$11.95

USCOMM-DC 6043-P80