Seismic Hydrodynamic Forces Acting on 2D Models of Gravity Dams

Jablonski, A. M.

Publisher's version / Version de l'éditeur:
https://doi.org/10.4224/20358989
Internal Report (National Research Council of Canada. Institute for Research in Construction), 1990-11-01

National Research Council Canada
Institute for Research in Construction

Conseil national de recherches Canada
Institut de recherche en construction

Seismic Hydrodynamic Forces Acting on 2D Models of Gravity Dams

Volume 1

by A.M. Jablonski

Internal Report No. 602

Date of Issue: November 1990

This is an internal report of the Institute for Research in Construction. It is not to be cited as a reference in other publications.

Canada

CONTENTS

1. INTRODUCTION

1.1 General Comments

This document describes a procedure to be followed to evaluate seismic hydrodynamic forces acting on 2D models of gravity dams. To meet objectives of this procedure a number of computer programs had to be developed together with a modification of the core program BEMC.2D based on the constant boundary element formulation. At the beginning of this chapter, the short description of the procedure is presented. The rest of this document covers user's guides to these programs together with a numerical example. Listings of them and references are also included.

1.2 Description of Procedure

The procedure is divided into four main steps. These steps should be followed according to the presented description.

Step 1 - Earthquake Input Data

First preparation of an input data file DATAS INPUT A1 must be completed based on an available earthquake record. Depending on the time step and the required number of points in the Fast Fourier Transformation subroutines the specific total time (representative duration time of an earthquake record) should be chosen. Creation of the file DATAS INPUT A1 is made through XEDIT (editor on VM) as e.g. a modification of a copied earthquake record file (in acceleration history).

In order to obtain a file INPS1 INPUT A1 run the program ACCELN FORTRAN A1. After screening INPS1 INPUT A1 w.r.t. number of points and chosen part of the original earthquake file, it can be used as an input file to the programs in Step 3 - Earthquake Response. Execution file EQN EXEC A1 is used to run the program ACCELN FORTRAN A1. Details are included in Section 2.2 - Earthquake Input Data.

Step 2 - Harmonic Response

The response of a dam-reservoir-foundation system subjected to a horizontal or vertical excitation is obtained with the use of the previously developed computer program BEMC2DN based on the constant boundary element formulation. The running of this program is fulfilled using a batch file BATCHS BATCH A1. Input data file DATAM INPUT A1 will be described in detail in subsection 2.3.2. The program BEMC2DN FORTRAN A1 can create three output files:

OUTM OUTPUT A1	a complete output file for each frequency step;
EIGENM OUTPUT A1	an empty file, which may be used in obtaining data from an eigenvalue subroutine;
INFTM INPUT A1	a vital output file, which is used as an input file in step 3 - Earthquake Response. Its name should be changed to INFT1 INPUT A1.

Step 3 - Earthquake Response

The Fast Fourier Transformation technique is used to obtain the response of the system to a horizontal or vertical component of the given earthquake acceleration record.

In order to obtain file OUTFFT OUTPUT A1 run program FOURTN FORTRAN A1 with two input files: INPS1 INPUT A1 from the ACCEL program and INFT1 INPUT A1 from the BEMC2DN program.

Execution file FOURTN2 EXEC A1 is used to run the program FOURTN FORTRAN A1 (NE = 4096 pts) or file FOURTN 1 EXEC A1 is used to run a modified program FOURTNE FORTRAN A1 (NE = 8192 pts).

Details are included in Section 2.4.

Step 4 - Plotting

The set of DISPPLA subroutines is used to obtain several output plots. They could be first reviewed online and later be dumped onto the laser printer associated with VM.

In order to obtain output plots run the program SPLOTN2 FORTRAN A1. Execution file DUM2 EXEC A1 is used to run this program. There are two input files: INPS1 INPUT A1 obtained from ACCELN FORTRAN A1 and OUTFFT OUTPUT A1 obtained from FOURTN FORTRAN A1 (or FOURTNE FORTRAN A1), for this program.

The program SPLOTN2 FORTRAN A1 using DISSPLA creates eight files:

1.	Acceleration record (ACCEL vs TIME)	by	PLOT1;
2.	FFT of Acceleration (Re)	by	PLOT2;
3.	FFT of Acceleration (Im)	by	PLOT3;
4.	FFT of Acceleration (Abs)	by	PLOT31;
5.	Harmonic response vs. frequency (Re)	by	PLOT4;
6.	Harmonic response vs. frequency (Im)	by	PLOT5;
7.	Harmonic response vs. frequency (Abs)	by	PLOT51;
8.	Earthquake response vs. frequency	by	PLOT 6.

Note: The above listed 4-step procedure should be performed for the horizontal and the vertical excitations. In order to share results of the earthquake response by PLOT6 a file named SUM1 OUTPUT A1 is produced. Rename this file to SUM1H OUTPUTA1 for the horizontal response and to SUM1V OUTPUT A1 for the vertical one. These files are later used by an additional program SUM FORTRAN A1. An execution file SUM1 EXEC A1 runs this program. The program SUM FORTRAN A1 produces the combined response from the superposition of two components (horizontal and vertical).

1.3 List of Computer Programs

ACCELN FORTRAN A1 - to prepare an earthquake input data
BEMC2DN FORTRAN A1 - to calculate a harmonic response
FOURTN FORTRAN A1 - to calculate earthquake response
SPLOTN2 FORTRAN A1 - to plot results
SUM FORTRAN A1 - to plot a combined response

2. *USER'S GUIDES TO PROGRAMS*

2.1 General Comments

In this chapter five computer programs are presented. The most important programs are: a core program BEMC2DN FORTRAN A1, a FFT program FOURTN FORTRAN A1 and a plotting program SPLOTN2 FORTRAN A1. Each step of this procedure was planned to run separately in allowing a prospective user to make appropriate changes in input and output statements.

First each program is described and later its input and output. A description of a procedure to run a specific program follows. A numerical example is presented in Appendix A. The listings of those programs are included in Appendix B. All programs are written in FORTRAN77.

2.2 Earthquake Input Data - ACCELN

2.2.1 Description of the ACCELN Program

This FORTRAN computer program can be used to calculate the input data file INPS1. The program could accommodate any number of points used in the original earthquake record. It can read also any specified part from this record.

2.2.2 Input Data

The following parameters could be changed at the beginning and before running the ACCELN program:

NT = number of time step (now set at 4096)
NI = initial step number to be read from the record file (now set at 0)
NL = last step number to be read from the same record (any number depending on number of steps in the earthquake record).

Input data is stored in the vector AC (9000). The dimension of the vector can be also changed. Communication between input data file and the program is maintained by the OPEN statement in the form:

OPEN (LEC, FORM = 'FORMATTED', STATUS = 'OLD', FILE = DATAS).

The input data file DATAS INPUT A1 is a modified version of the considered original earthquake data (in acceleration history). Depending on the time step and the number of points in the Fast Fourier Transformation subroutines, the specific total time (duration) is chosen. Creation of that file may be done using XEDIT (online editor on VM). READ statement for the DATAS should be changed according to the format used in the earthquake record.

2.2.3 Output

The program ACCELN creates an output file called INPS1 INPUT A1. Output data is stored in the vector A (9000). WRITE statement for the INPS1 should be changed accordingly to the required format.

2.2.4 Procedure to run ACCELN on VM

To run the program ACCELN simply invoke the execution file EQN EXEC A1. The execution file EQN EXEC A1 is given below:

```
/* EXEC FILE EQN */
SETLEVEL FORTRAN NEW
'FI * CLEAR'
FORTVSL
FORTVS2 ACCELN '( NOFIPS NOPRI'
LOAD ACCELN '( CLEAR START'
SENDFILE INPS1 INPUT A1 account name
EXIT
```

2.3 Harmonic Response - BEMC2DN

2.3.1 Description of the BEMC2DN Program

The constant boundary element program for 2D reservoirs - BEMC2DN FORTRAN A1 is a modified version of the previously developed program BEMC2D FORTRAN A1. The program BEMC2D FORTRAN A1 constitutes a VM/IBM version of the original program BEMC.2D developed for the APOLLO/DOMAIN computer [1].

This program produces harmonic responses due to horizontal or vertical excitation in hydrodynamic forces. Those results are later used as an input in the Fast Fourier Transform analyses included in the next program FOURTN FORTRAN A1.

The BEMC2DN program can be used to calculate the hydrodynamic pressure on a rigid dam storing a two-dimensional finite or infinite water reservoir and subjected to a horizontal or/and vertical harmonic excitation.

The program uses the constant boundary elements. The pressures and the pressure derivatives are assumed to have constant values over the element and equal to their values at the mid-node of each element. The development of this 2D boundary element model of a gravity dam-reservoir-foundation system subjected to ground motion has been presented by Liu and Cheng [2], Humar and Jablonski [3], and Jablonski [4].

There are no limitations on the shape of the dam face or the finite reservoir. The infinite reservoir, however, has to be divided into two parts: one finite and irregular and other regular but infinite. The upstream face of the dam, the reservoir surface, the bottom and the upstream and of the reservoir (in case of the infinite reservoir this end is an interface between irregular and regular parts), are to be covered by a constant boundary element mesh.

The main program can handle up to 100 boundary elements. However, this could be extended to any number (depending on the computer capacity).

The main program calls the eight following subroutines:

INPUT - to read all data and print the first part of the output.

EIGEN - to set and calculate the complex eigenvalue problem in case of infinite radiation with foundation damping included.

EIGENR - to set and calculate the real eigenvalue program in case of infinite radiation only and without foundation damping.

FMAT - to form matrices **G** and **H** and rearrange them accordingly to the boundary conditions on the top of the reservoir.

ADDFBC - to modify the matrix **H** for the boundary conditions on the bottom of the reservoir and at its far boundary (transmitting boundary), and to compute the vector of known values of the R.H.S. of the system of equations. This subroutine forms the system of equations for the damped case.

ADDUND - to form the system of equations and apply all B.C.'s for the undamped case.

SLNPD - the solver of linear equations systems (Gauss elimination)

OUTPUT - to reorder the vector of computed unknowns and print out the rest of the output file.

The subroutine EIGEN calls the subroutine:

FSFEM - to compute the real and imaginary portions of the eigenvector matrix and at the same time to create the real and imaginary parts of the eigenvalue vector.

The subroutine FSFEM calls the following subroutines:

SET - to form the complex matrices for the eigenvalue problem, which governs the damped vibrations in the infinite part of the reservoir $(\mathbf{A} + i\omega\gamma\,\mathbf{B})\,\mathbf{p} = \lambda^2\mathbf{F}\,\mathbf{p}$, where matrices **A**, **B** and **F** are described in detail in Refs. 4 and 5.

CQZHES - this subroutine is a complex analogue of the first step of the QZ algorithm for solving generalized real matrix eigenvalue problems [5]. This subroutine accepts a pair of complex general matrices and reduces one of them to upper Hessenberg form with real (and non-negative) subdiagonal elements and the other to upper triangular form using unitary transformations.

CQZVAL - this subroutine is a complex analogue of steps 2 and 3 of the QZ algorithm for solving generalized real matrix eigenvalue problems. This subroutine accepts a pair of complex matrices, one of them in upper Hessenberg form and the other in upper triangular form. The Hessenberg matrix must further have real subdiagonal elements. It reduces the Hessenberg matrix to triangular form using unitary transformations while maintaining the triangular form of the other matrix and making its diagonal elements real and non-negative. It then returns quantities whose ratios give the generalized eigenvalues.

CQZVEC - is a complex analogue of the fourth step for solving generalized real matrix eigenvalue problems. This subroutine accepts a pair of complex matrices in upper triangular form, where one of them further must have real diagonal elements. It computes the eigenvectors of the triangular problem and transforms the result back to the original system. This is the last step of QZ algorithm originally developed by Stewart and Moler [5] and later modified by B.S. Garbov from Applied Mathematics Division, Argonne National Laboratory, Illinois, U.S.A.

NORMA - normalizes the eigenvectors obtained from the QZ algorithm w.r.t matrix **F**.

The subroutine EIGENR calls the following subroutines:

SETR - forms the real matrices for the eigenvalue problem, wich governs the undamped vibrations in the infinite part of the reservoir: $\mathbf{A}\,\mathbf{p} = \lambda^2 \mathbf{F}\,\mathbf{p}$.

JACOBI - uses the generalized Jacobi method to solve the generalized real matrix eigenvalue problem [6]. Eigenvectors and eigenvalues are presented in a normalized form in its output.

The subroutine FMAT calls the following subroutines:

INTE - computes the **G** and **H** matrix elements except those on the diagonal by means of numerical integration along the boundary elements. The numerical integration uses the 4-point Gauss quadrature scheme.

INLO - calculates the diagonal elements of matrix **G**.

The subroutine INTE calls the subroutine:

BFUNCT - computes the values of Bessel function Yo (kr) and its derivative using the series representation of the Bessel functions [7,8].

Subroutines ADDFBC, ADDUND and OUTPUT call the subroutine:

TP - calculates the transformation matrix $\mathbf{T_p}$ for adjustment of linearly formed for end condition into content element formulation.

2.3.2 Input data

Input data is read from the file DATAM INPUT A1 (unit No. LEC = 2).

The following table shows the order of presentation of data in the program BEMC2DN FORTRAN A1:

TABLE 2-1

Input Data for the BEMC2DN FORTRAN

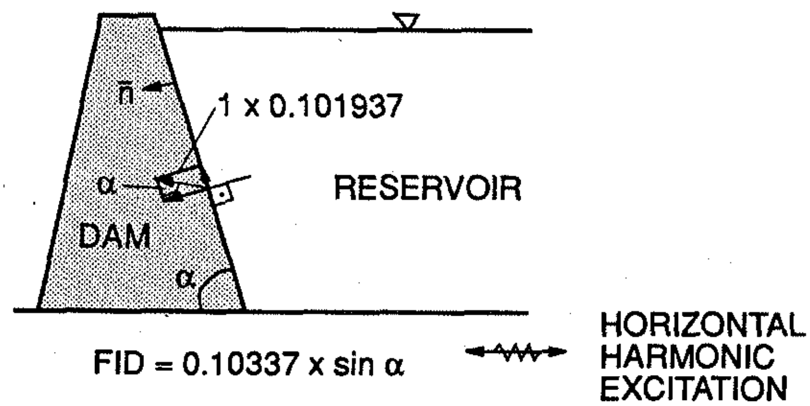
Card Group	No. of Cards	Format	FORTRAN Variables
A	1	18A4	TITLE
B1	1	General Format	N = number of boundary elements N0 = last element number at top near the dam N1 = last element number at the base of the dam N2 = last element number at the reservoir bottom near the end NOUT = 1, flag for detailed output, 0, flag for output at the dam only
B2	1	General Format	HY = height of the dam C = velocity of sound in water ALFA = wave reflection coefficient DF = vertical acceleration of the first mode at the far end
C	N	General Format	X,Y = coordinates of nodal points
D	N1-N0	General Format	FID = boundary conditions on the dam face
E	N2-N1	General Format	FIB = boundary conditions on the foundation base
F	No. of cases	General Format	OMEGA = excitation frequency

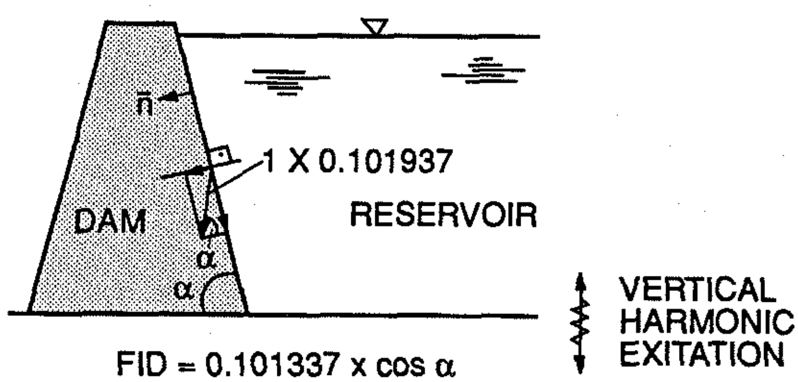
The node numbers must start from the element at the top of the reservoir adjacent to the far end and numbered in counter-clockwise direction.

Units:

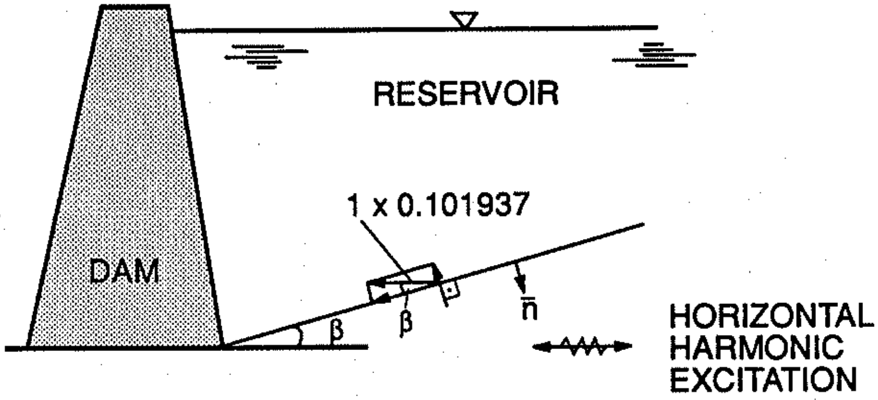
HY	=	height of the dam in m;
C	=	1440 m/sec, velocity of sound in water;
ALFA	=	wave reflection coefficient, dimensionless value from 0 to 1.0;
DF	=	0.101937, unit vertical ground acceleration (w/g, where w = 1 metric tonne and g = 9.81 m/s^2);
X,Y	=	coordinates of modal points in m;
*FID	=	0, for vertical excitation; 0.101937, unit horizontal acceleration (w/g);
*F1B	=	0, for a rigid reservoir bottom; 0.101937, unit vertical ground acceleration;
OMEGA	=	in rad/s

BOUNDARY CONDITIONS ON INCLINED DAM FACE

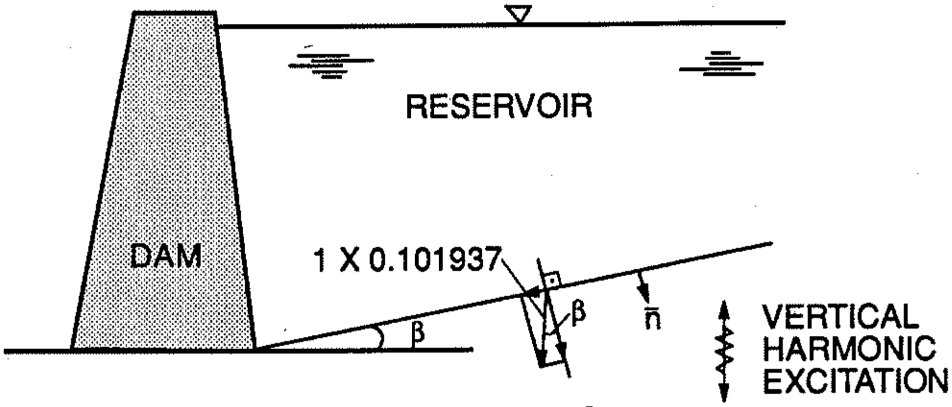




BOUNDARY CONDITIONS ON INCLINED RESERVOIR BOTTOM



FIB = -0.101937 x sin β



FIB = 0.101937 x cos β

Whenever the direction of excitation component is the same as the one of the outward normal to the reservoir boundary, the input value should be positive, and negative otherwise.

2.3.3 Output

The program BEMC2DN FORTRAN A1 creates three output files: OUTM OUTPUT A1, EIGENM OUTPUT A1 and INFTM INPUT A1. After changing the name of the last file to INFT1 INPUT A1, it can be used as an input to calculate the specific earthquake response.

2.3.4 Procedure to run BEMC2DN on VM

To run the program BEMC2DN FORTRAN A1 simply invoke the batch file BATCHS BATCH A1. The batch file BATCHS BATCH A1 is given below:

```
/* BATCHS */
/* TESTING BEMC2DN PROGRAM */
EXEC SYSPROF
ADDRESS COMMAND CP LINK account name 191 196 RR password
ACCESS 196 'B/A'
TAGPRINT
TRACE C
SETLEVEL FORTRAN NEW
'F1 * CLEAR'
TEMPDISK C10
FI 3 DISK OUTM OUTPUT C
FI 6 DISK EIGENM OUTPUT C
```

```
FI 7 DISK INFTM INPUT A1
FORTVSL
FORTVS2 BEMC2DN '(NOPRI NOFIPS'
LOAD BEMC2DN '(CLEAR START'
/* SENDFILE OUTM OUTPUT C account name */
/* TSSPRT OUTM OUTPUT C '(CL D DEFAULT' */
SENDFILE INFTM INPUT A1 account name
```

2.4 Earthquake Response - FOURTN

2.4.1 Description of the FOURTN Program

The program FOURTN calculates the response in hydrodynamic pressures of the linear dam-reservoir-foundation system to the excitation history given in the discrete form. The program FOURTN FORTRAN A1 employs the FFT routines originally developed by Paul Swarztrauber, National Center for Atmospheric Research, Denver, Colorado, USA, and in the form provided by the new version 1.0 of the math Library, IMSL [9].

For obtaining the response to an arbitrary ground motion through the frequency domain analysis, the standard procedure can be summarized in the following steps:

1. The excitation history has to be expressed in terms of the harmonic components.
2. The evaluation of the response of the linear dam-reservoir-foundation system is done next.
3. The superposition of the harmonic responses to obtain the total response completes the solution.

When the excitation history is given in terms of function of the acceleration a(t) and F(ω) represents the harmonic response of the system to unit excitation $e^{i\omega t}$, the so-called harmonic synthesis is usually expressed as:

$$F(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} A(\omega)\ \bar{F}(\omega)\ e^{i\omega t}\ d\omega \tag{1}$$

where

$$A(\omega) = \int_{-\infty}^{\infty} a(t)\ e^{-i\omega t}\ dt \tag{2}$$

Equation 2 represents the Fourier Transform of the acceleration function a(t), while Eq. 1 is the Inverse Fourier Transform of the product of the frequency functions A(ω) and F(ω).

Both equations can be solved by a numerical technique based on the Discrete Fourier Transform (DFT).

The first step is to assure the periodic type of excitation, which implies an approximation of an arbitrary motion, which is not periodic. The infinite integrals 1 and 2 are replaced by the finite sums. The selected period of excitation, T_o, serves to define the lowest frequency in the analysis.

$$\Delta\omega = \frac{2\pi}{T_0} \tag{3}$$

The excitation period is then divited into N equal time steps Δt, and the acceleration function a(t) is defined for the discrete time steps $t_m = m\Delta t$.

The discrete forms of the expressions given by Eqs. 1 and 2 can be rewritten as follow:

$$F(m\Delta t) = \frac{\Delta\omega}{2\pi} \sum_{n=0}^{N-1} [A(n\Delta\omega)\ \bar{F}(n\Delta\omega)]\ e^{i\frac{2\pi nm}{N}} \tag{4}$$

and

$$A(n\Delta\omega) = \Delta t \sum_{m=0}^{N-1} [a(m\Delta t)]\ e^{-i\frac{2\pi nm}{N}} \tag{5}$$

Then, the Fast Fourier Transform (FFT) technique is used to evaluate Eqs. 4 and 5. The computer program FOURTN FORTRAN A1 calculates the response of the system subjected to an arbitrary excitation history given in the discrete form $a(m\Delta t)$.

Using the DFT implies that both the excitation history and the response are periodic functions of period T_o. This results in errors in evaluation of the transient response of the system. The excitation period is elongated by adding an additional grace band of zero excitations to the end. It ensures that the system will have a period of free vibration so that the system will eventually come to the rest at the end of each period.

The program FOURTN FORTRAN A1 has three parameters, which can be easily changed.

NT = 4096, Number of steps to be employed in the FFT;
NC = 200, Initial number of frequency steps employed to calculate harmonic response;
ND = 10, Number of nodes at dam face introduced in the BEMC2DN program.

This program uses two input files: INPS1 INPUT A1 and INFT1 INPUT A1. Reading statements should be checked for both of them and replaced by required formats used in these files.

The following steps are included in the main program:

1. Read an input data created by program ACCELN FORTRAN A1;
2. Divide the acceleration vector by 100 to have it in m/s^2;
3. Carry out forward FFT;
4. Read required harmonic responses;
5. Modify harmonic response to have it in the form ready for FFT routine;
6. Carry out inverse FFT for a chosen harmonic response vector.

The main program calls the three following subroutines.

IWKIN (224611) - to change the amount of space allocated. FORTRAN subroutines that work with matrices as input or output often require extra arrays for use as workspace. IMSL routines usually do not require the user explicitly to allocate such arrays for use as workspace (it is done automatically). By default, the total amount of space allocated in the common area for storage of numeric state is 5000 numeric storage units. This space is allocated as needed for all variables. The user can change the allocation by supplying the following FORTRAN statements in conjunction with this subroutine:

COMMON /WORKSR/ RWKSP

REAL RWKSP (24611)

CALL IWKIN (24611) It will request 24611 units (this number can be changed if required).

If an IMSL routine attempts to allocate workspace in excess of the amount available in common stack (5000), the routine issues an error message that indicates how much space is required and prints statements like above. Other details may be found in Ref. 9.

FFTCF (NT, AC, AN) - computes the Fourier coefficients of a complex periodic vector AC. Where:

NT = length of the sequence or number of steps to be employed in the FFT;
AC(I) = processed acceleration vector from earthquake records of length NT (input);
AN(I) = complex vector of length NT containing Fourier coefficients (output)

FFTCF computes the discrete complex Fourier transform of a complex vectof of size NT. The method used is a variant of the Cooley-Tukey algorithm which is most efficient when NT is a product of small prime factors. If NT satisfies this condition then the computational effort is proportional to NT Log (NT) [9].

FFTCB (NT, Y2, Y2N) - computes the inverse Fourier transform of the vector Y2(I) = AN(I) * Y1(I). AN(I) is a complex vector containing Fourier coefficients of the acceleration record. Y1(I) is a complex vector of total hydrodynamic forces normalized w.r.t. the hydrostatic force (in this case 5000 for the dam height 100 m). The method used by the FFTCB subroutine is the same as above [9].

2.4.2 Input Data

The following parameters could be changed at the beginning and before running the FOURTN program:

NT = number of steps in the FFT;
NC = initial number of frequency steps to calculate harmonic response;
ND = number of nodes at dam face (in the BEMC2DN program).

Two input files along with one output file are required and they are linked through the OPEN statements in the form:

```
DATA INPS1/'/INPS1 INPUT A1'/
DATA INFT1/'/INFT1 INPUT A1'/
DATA OUTFFT/'/OUTFFT OUTPUT A1'/

OPEN (2, FORM = 'FORMATTED', STATUS = 'OLD', FILE = INPS1)
OPEN (3, FORM = 'FORMATTED', STATUS = 'OLD', FILE = INFT1)
OPEN (5, FORM = 'FORMATTED', STATUS = 'NEW', FILE = OUTFFT)
```

The input file INPS1 INPUT A1 is an output from the ACCELN program, and INFT1 INPUT A1 is an output created by the BEMC2DN program. READ statements for the INPS1 should be changed according to the format used in this particular file.

2.4.3 Output

The program FOURTN FORTRAN A1 creates an output file called OUTFFT OUTPUT (very large file). That file should be deleted after being used in the next step (i.e. in plotting graphs).

2.4.4 Procedure to run FOURTN on VM

To run the program FOURTN simply invoke the execution file FOURTN2 EXEC A1. The execution file FOURTN2 EXEC A1 is given below:

```
/* FOURT */
'FI * CLEAR'
FORTVSL IMSL1 IMSL2
FORTVS2 FOURTN '(NOPRI NOFIPS'
FI 2 DISK INPS1 INPUT A1
FI 3 DISK INFT1 INPUT A1
FI 5 DISK OUTFFT OUTPUT A1 '(RECFM F LRECL 80'
LOAD FOURTN '(CLEAR START'
EXIT
```

2.5 Plotting Program - SPLOTN2

2.5.1 Description of the SPLOTN2 Program

The program SPLOTN2 produces following graphs (on screen of a terminal or printed by a laser printer):

1. Acceleration record (time history) - PLOT1
2. Real part of DFT of acceleration - PLOT2
3. Imaginary part of DFT of acceleration - PLOT3
4. Absolute value of DFT of acceleration - PLOT31
5. Real part of harmonic response - PLOT4
6. Imaginary part of harmonic response - PLOT5
7. Absolute value of harmonic response - PLOT51
8. Earthquake response in hydrodynamic pressure (time history) - PLOT6

The SPLOTN2 FORTRAN A1 uses the routines from the DISSPLA package installed on VM [10].

The program has two parameters:

NT = 4086, number of frequency steps used in the FFT procedure;
NE = 4096, maximum dimension of arrays used in the program.

The reading of the acceleration history vector should be based on an appropriate format in READ statement.

The main program calls the nine following subroutines:

NOMDEV(1)	-	to nominate a graphics device named in the DISSPLA command in the execution file DUM2 EXEC A1 (level 0 → 1);
PLOT1 (NE, X, Y)	-	to plot acceleration history record;
PLOT2 (NE, X, ANR)	-	to plot real part of DFT of acceleration history record;
PLOT3 (NE, X, ANI)	-	to plot imaginary part of DFT of acceleration history record;
PLOT4(NX, X, Y1R)	-	to plot real part of harmonic response (NX = 200 steps);
PLOT5 (NX, X, Y1I)	-	to plot imaginary part of harmonic response;
PLOT51 (NX, X, Z)	-	to plot absolute value of harmonic response;
PLOT31 (NE, X, AA)	-	to plot absolute value of DFT of acceleration history;
PLOT6 (NE, X, Y2NR)	-	to plot hydrodynamic force time history (earthquake response);
DONEPL	-	to terminate DISSPLA (level 1 → 0);

Each of the plotting subroutines uses the DISSPLA hierarchy along with a number of DISSPLA (old and new) routines.

The subroutine PLOT1 is described in detail as an example. The rest of subroutines follows this example.

The subroutine PLOT1 (N, X, Y) has a number of steps. First, it sets steps and maximum values on X and Y axes. It finds also maximum and minimum of the particular acceleration history record.

The subroutine PLOT1 calls the following DISSPLA routines:

PAGE (11.0, 8.5)	-	to set the page limits (DISSPLA sets the page limits to the standard 8.5 by 11 inch page size). This subroutine allows a change in the page limits;
NOBRDR	-	to suppress the default page border in order to draw an enhanced border;

PHYSOR (1.0, 1.0) - to specify a physical origin on the page;

TRIPLX - to specify character style (fonts) - TRIPLX is one from eight options;

TITLE ('MANIC-3 STA2-A HOR.', 19, 'STEPS, 1 STEP = 0.02 SEC', 21, 'ACCELERATION M/SEC/SEC', 22, 9.0, 6.5)
- to specify titles and captions;

GRAF (XMIN, XSTP, XMAX, YMIN, YSTP, YMAX) - to draw axes;

MESSAG ('MAX = ',6,4.0,6) - to write specific captions and messages relative to physical origin (example is given);

REALNO (RMAX, +2,4.8,6) - to place real numbers anywhere on a plot relative to the physical origin (example is given);

INTNO (I1, 6.6, 6) - to position integer numbers anywhere on a plot relative to the physical origin (example is given);

STRTPT (0.0, 3.25) - to move to the point without drawing the line from the physical origin to desired point (example is given);

CONNPT (9.0, 3.25) - to connect successive points (in this case - (9.0, 3.25)) with straight lines starting from the previous position (in this case - (0.0, 3.25));

CURVE (X, Y, N, O) - to plot data (after the axis system has been defined and CA-DISSPLA). IMARK = 0 (marker specification means points connected without symbols drawn).

2.5.2 Input Data

Two initial parameters NT = 4096 and NE = 4096 may be changed depending on the number of frequency steps used in the FFT procedure.

The SPLOTN2 FORTRAN A1 uses two input files and creates one output file. they are linked to the main program through three OPEN statements in the following form:

```
DATA INPS1/'/INPS1 INPUT A1'/
DATA OUTFFT/'/OUTFFT OUTPUT A1'/
DATA SUM1/'/SUM1 OUTPUT A1'/.

OPEN (2, FORM = 'FORMATTED', STATUS = 'OLD', FILE = INPS1)
OPEN (3, FORM = 'FORMATTED', STATUS = 'OLD', FILE = OUTFFT)
OPEN (4, FORM = 'FORMATTED', STATUS = 'NEW', FILE = SUM1)
```

All input data is provided by two earlier programs ACCELN and FOURTN.

When all graphs are obtained, the output file OUTFFT can be deleted.

2.5.3 Output

The program SPLOTN2 FORTRAN A1 creates an output file called SUM1 OUTPUT A1. For horizontal response this name should be changed to SUM1H OUTPUT A1 and for vertical to SUM1V OUTPUT A1, respectively.

2.5.4 Procedure to run SPLOTN2 on VM

To run the program SPLOTN2 FORTRAN A1 invoke the given execution file DUM2 EXEC A1 in two versions.

First version of the DUM2 EXEC A1 is directed to review plots on screen of the terminal interfacing VM:

```
/* DUM2 EXEC FILE */
'FI * CLEAR'
SETLEVEL FORTRAN NEW
FORTVS2
FORTVS2 SPLOTN2'(NOFIPS NOPRINT'
DISSPLA SPLOTN2 FILE '('DUM2')
PLOT DUM2 TEKALL' (4010, 960)'
/* PLOT DUM2 LASER */
'FI * CLEAR'
EXIT
```

Second version of the DUM2 EXEC A1 plots the final sets of graphs on the laser printer located in the computer center.

```
/* DUM2 EXEC FILE */
'FI * CLEAR'
SETLEVEL FORTRAN NEW
FORTVSL
FORTVS2 SPLOTN2 '(NOFIPS NOPRINT'
DISSPLA SPLOTN2 FILE '('DUM2')'
/* PLOT DUM2 TEKALL '(4010, 960)' */
PLOT DUM2 LASER
'FI * CLEAR'
EXIT
```

2.6 Combined Response - SUM

2.6.1 Description of the SUM Program

The program SUM FORTRAN A1 combines horizontal and vertical time histories of the hydrodynamic force and plots them in the form of the combined response.

This program has one parameter:

NT = 4096, number of time steps in the time history of the hydrodynamic force.

It reads two vectors and adds them together. It uses also one subroutine PLOT1 (N,X,Y) which is similar to one used in the program SPLOTN2.

The main program calls the three following subroutines:

NOMDEV(1)	-	to nominate a graphics device named in the DISSPLA command in the execution file SUM1 EXEC A1 (level 0 → 1);
PLOT1 (NT, X, E)	-	to plot the final combined response in hydrodynamic forces;
DONEPL	-	to terminate DISSPLA (level 1 → 0).

The subroutine PLOT1 uses the DISSPLA hierarchy along with a number of DISSPLA routines (see Section 2.5.1).

2.6.2 Input Data

The initial parameter, NT = 4096 can be changed depending on the needs and based on calculations carried out previously.

The SUM FORTRAN A1 uses two input files and creates one plot. They are linked to the main program through three OPEN statements in the following form:

```
DATA SUM1H/'/SUM1H OUTPUT A1'/
DATA SUM1V/'/SUM1V OUTPUT A1'/

OPEN (2, FORM = 'FORMATTED', STATUS = 'OLD', FILE = SUM1H)
OPEN (3, FORM = 'FORMATTED', STATUS = 'OLD', FILE = SUM1V)
```

All input data is provided by the program SPLOTN2.

2.6.3 Output

The program SUM produces one plot only for a combined response in hydrodynamic pressure (time history of hydrodynamic pressure). It does not create any output file.

2.6.4 Procedure to run SUM on VM

To run the program SUM1 invoke the given below execution file SUM1 EXEC A1.

```
/* SUM 1 EXEC FILE */
'F1 * CLEAR'
SETLEVEL FORTRAN NEW
FORTVSL
FORTVS2 SUM '(NOFIPS NOPRINT'
DISSPLA SUM FILE '('SUM1')'
PLOT SUM1 TEKALL '(4010, 9600)'
/* PLOT SUM 1 LASER */
'FI * CLEAR'
EXIT
```

The above version serves for checking actual graph on screen of the terminal interfacing VM. In order to dump it on the laser printer change two following statements to the given form:

```
/* PLOT SUM1 TEKALL '(4010, 960)'
PLOT SUM1 LASER
```

3. *REFERENCES*

[1] Jablonski, A.M., "Boundary Element Analysis of Earthquake Induced Hydrodynamic Pressures in a Water Reservoir", Vol. 2, User's Guides to Computer Programs, Ph.D. Thesis Submitted to Carleton University, Ottawa, Canada, 1988.

[2] Liu, P.L.-F., Cheng, A.H.-D., "Boundary Solutions for Fluid-Structure Interaction", Journal of Hydraulic Engineering, ASCE, Vol. 110, 1984, pp. 51-64.

[3] Humar, J.L., Jablonski, A.M., "Boundary Element Reservoir Model for Seismic Analysis of Gravity Dams", Earthquake Engineering and Structural Dynamics, Vol. 16, 1988, pp. 1129-1156.

[4] Jablonski, A.M., "Boundary Element Analysis of Earthquake Induced Hydrodynamic Pressures in a Water Reservoir", Vol. 1, Ph.D. Thesis Submitted to Carleton University, Ottawa, Canada, 1988.

[5] Moler, C.B., Stewart, G.W., "An Algorithm for Generalized Matrix Eigenvalues Problems", Journal of Numerical Analysis, SIAM, Vol. 10, 1973, pp. 241-265.

[6] Bathe, K.-J., "Finite Element Procedures in Engineering Analysis", Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[7] MacLachlan, N.W., "Bessel Functions for Engineers", Oxford at the Clarendon Press, Oxford, England, 1955.

[8] Hanna, Y.G., "Application of Boundary Element Method to Certain Problems in Structural Dynamics", M.Eng. Thesis Submitted to Carleton University, Ottawa, Canada, 1980.

[9] User's Manual, Math/Library, Fortran Subroutines for Mathematical Applications, Version 1.0, IMSL, Problem-Solving Software Systems, Houston, Texas, 1987.

[10] CA-DISSPLA User Manual, Version 11.0, Volume 1 and 2, Computer Associates International, Inc., NY, USA, 1988.

National Research Council Canada **Conseil national de recherches Canada**

Institute for Research in Construction
Institut de recherche en construction

Seismic Hydrodynamic Forces Acting on 2D Models of Gravity Dams

Volume 2 - Appendices

by A.M. Jablonski

Internal Report No. 602

Date of issue: November 1990

This is an internal report of the Institute for Research in Construction. It is not to be cited as a reference in other publications.

Canada

SEISMIC HYDRODYNAMIC FORCES ACTING ON 2D MODELS OF GRAVITY DAMS

User's Guide to Computer Programs

Appendices

by

Alexander M. Jablonski

September 1990

Institute for Research in Construction
National Research Council of Canada
Ottawa, Ontario, Canada K1A 0R6

CONTENTS

APPENDIX A

NUMERICAL EXAMPLE

<u>Input Data</u>

For the purpose of illustration, a simple example is presented in this Appendix.

A two-dimensional infinite reservoir impounded by a straight gravity dam is subjected to an El Centro horizontal acceleration (file INPSHE INPUT A1). The reservoir is modelled using 55 constant boundary element mesh as shown in Fig. A.1

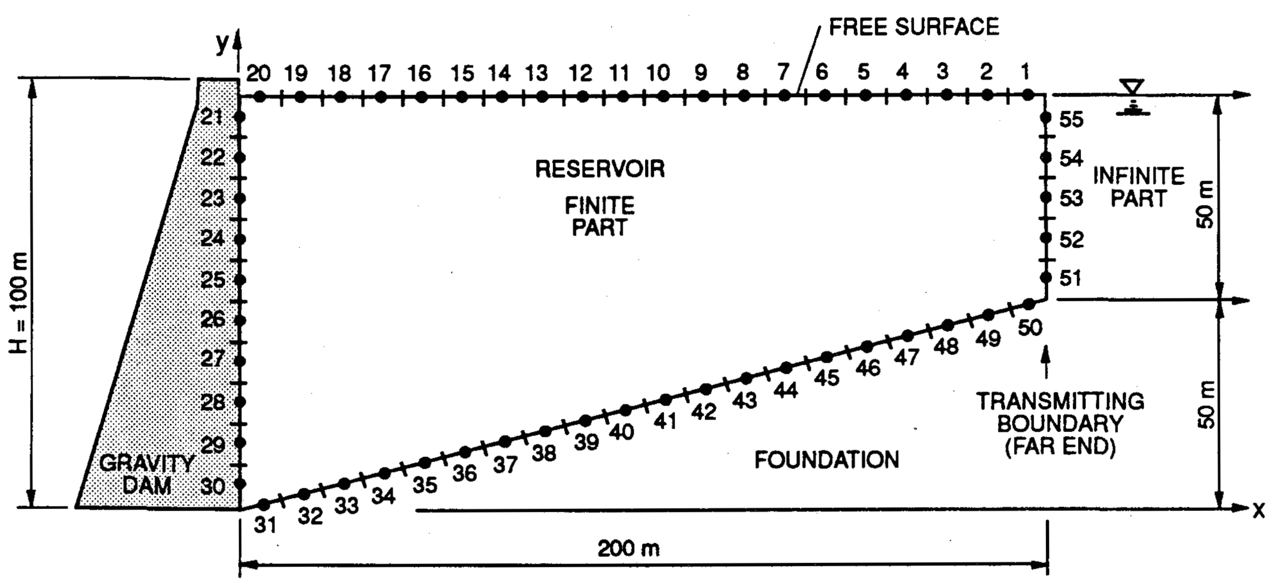


Fig. A.1 The Constant Boundary Element Model of Infinite 2D Reservoir - N = 55

The list of the parameters used in the input data file, TESTM INPUT A1 is as follows:

N = 55 (elements)
N0 = 20 (last elem. number at the top near a dam)
N1 = 30 (last elem. number at the base of the dam)
N2 = 50 (last elem. number at the base near far end)
NOUT = 1 (flag for detailed output)

HY = 50 (height of the infinite part of the reservoir in meters)
C = 1440 (velocity of sound in water in m/s^2)
ALFA = 0.0 (coefficient of reflection in full damping case)
DF = 0.0 (vertical acceleration of the first mode at the far end)

Coordinates of the nodal points together with boundary conditions on the dam face and on the foundation base together with excitation frequencies are listed in a copy of the original file TESTM INPUT A1.

Listing of the file TESTM INPUT A1

```
RRRRRRRRRRR   MM          MM                    11         555555555555     11
RRRRRRRRRRRR  MMM        MMM                   111         555555555555    111
RR         RR MMMM      MMMM                  1111         55             1111
RR         RR MM MM    MM MM                    11         55               11
RR         RR MM  MMMM   MM                     11         55               11
RRRRRRRRRRRR  MM    MM    MM                    11         555555555        11
RRRRRRRRRRR   MM          MM                    11         5555555555       11
RR     RR     MM          MM                    11                 55       11
RR      RR    MM          MM                    11                  55      11
RR       RR   MM          MM                    11                  55      11
RR        RR  MM          MM              1111111111   555555555555   1111111111
RR         RR MM          MM              1111111111   55555555555    1111111111

                    MM          MM          2222222222     00000000
                    MMM        MMM         222222222222   0000000000
                    MMMM      MMMM         22         22 00        0000
                    MM MM    MM MM                    22 00       00 00
                    MM  MMMM   MM                     22 00      00  00
                    MM    MM    MM                   22  00     00   00
                    MM          MM                 22    00   00     00
                    MM          MM               22      00 00       00
                    MM          MM             22        0000        00
                    MM          MM           22          000         00
                    MM          MM         222222222222   0000000000
                    MM          MM         222222222222    00000000

  JJJJJJJJJJ  AAAAAAAAAA   BBBBBBBBBBB  LL           000000000000 NN         NN  SSSSSSSSSS  KK         KK
  JJJJJJJJJJ AAAAAAAAAAAA  BBBBBBBBBBBB LL           000000000000 NNN        NN SSSSSSSSSSSS KK        KK
      JJ     AA        AA  BB        BB LL           00        00 NNNN       NN SS        SS KK       KK
      JJ     AA        AA  BB        BB LL           00        00 NN NN      NN SS           KK      KK
      JJ     AA        AA  BB       BB  LL           00        00 NN  NN     NN SSS          KK     KK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB   LL           00        00 NN   NN    NN  SSSSSSSSS   KKKKKKK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB   LL           00        00 NN    NN   NN   SSSSSSSSS  KKKKKKK
      JJ     AA        AA  BB       BB  LL           00        00 NN     NN  NN         SSS  KK     KK
JJ    JJ     AA        AA  BB        BB LL           00        00 NN      NNNN           SS  KK      KK
JJ    JJ     AA        AA  BB        BB LL           00        00 NN       NNN  SS       SS  KK       KK
JJJJJJJJ     AA        AA  BBBBBBBBBBBB LLLLLLLLLLLL 000000000000 NN        NN  SSSSSSSSSSSS KK        KK
 JJJJJJ      AA        AA  BBBBBBBBBBB  LLLLLLLLLLLL 000000000000 NN         N   SSSSSSSSSS  KK         KK

RRRRRRRRRRR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS    8888888888    3333333333    2222222222     00000000
RRRRRRRRRRRR SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  888888888888  333333333333  222222222222   0000000000
RR        RR SS        SS  CC        CC  SS        SS  88        88  33        33  22        22  00        0000
RR        RR SS            CC            SS            88        88            33            22  00       00 00
RR        RR SSS           CC            SSS            88      88             33            22  00      00  00
RRRRRRRRRRRR  SSSSSSSSS    CC             SSSSSSSSS      88888888            3333           22   00     00   00
RRRRRRRRRRR    SSSSSSSSS   CC              SSSSSSSSS     88888888            3333         22     00    00    00
RR    RR             SSS   CC                    SSS  88        88             33       22       00  00     00
RR     RR             SS   CC                     SS  88        88             33     22         0000       00
RR      RR   SS       SS   CC        CC  SS       SS  88        88  33         33   22           000        00
RR       RR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  888888888888  333333333333  222222222222   0000000000
RR        RR  SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS    8888888888    3333333333   222222222222    00000000
```

```
DATE/TIME= AUG 13, 1990 13:03:40        | PRINT: CLASS= E      | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP                   |    FORMS= S1PT       | RM. 151
NODES O\E\C=NRCVM01 \NRCVM01 \NRCVM01   |    FCB=  STD8        | M-20
JOB = J8320 START                       |    UCS=  GT15        |
```

```
2D INFINITE RESERVOIR BY THE BEMC                                  TES00010
55,20,30,50,1                                                      TES00020
50,1440,0.00,0.0                                                   TES0003
200.0,100.0                                                        TES00040
190.0,100.0                                                        TES00050
180.0,100.0                                                        TES00060
170.0,100.0                                                        TES00070
160.0,100.0                                                        TES00080
150.0,100.0                                                        TES00090
140.0,100.0                                                        TES00100
130.0,100.0                                                        TES00110
120.0,100.0                                                        TES00120
110.0,100.0                                                        TES00130
100.0,100.0                                                        TES00140
90.0,100.0                                                         TES00150
80.0,100.0                                                         TES00160
70.0,100.0                                                         TES00170
60.0,100.0                                                         TES00180
50.0,100.0                                                         TES00190
40.0,100.0                                                         TES00200
30.0,100.0                                                         TES00210
20.0,100.0                                                         TES00220
10.0,100.0                                                         TES00230
0.0,100.0                                                          TES00240
0.0,90.0                                                           TES00250
0.0,80.0                                                           TES00260
0.0,70.0                                                           TES00270
0.0,60.0                                                           TES00280
0.0,50.0                                                           TES00290
0.0,40.0                                                           TES00300
0.0,30.0                                                           TES00310
0.0,20.0                                                           TES00320
0.0,10.0                                                           TES00330
0.0,0.0                                                            TES00340
10.0,2.5                                                           TES00350
20.0,5.0                                                           TES00360
30.0,7.5                                                           TES00370
40.0,10.0                                                          TES00380
50.0,12.5                                                          TES00390
60.0,15.0                                                          TES00400
70.0,17.5                                                          TES00410
80.0,20.0                                                          TES00420
90.0,22.5                                                          TES00430
100.0,25.0                                                         TES00440
110.0,27.5                                                         TES00450
120.0,30.0                                                         TES00460
130.0,32.5                                                         TES00470
140.0,35.0                                                         TES00480
150.0,37.5                                                         TES00490
160.0,40.0                                                         TES00500
170.0,42.5                                                         TES00510
180.0,45.0                                                         TES00520
190.0,47.5                                                         TES00530
200.0,50.0                                                         TES00540
200.0,60.0                                                         TES00550
```

```
200.0,70.0                                                   TES00560
200.0,80.0                                                   TES00570
200.0,90.0                                                   TES00580
0.101937                                                     TES00600
0.101937                                                     TES00610
0.101937                                                     TES00620
0.101937                                                     TES00630
0.101937                                                     TES00640
0.101937                                                     TES00650
0.101937                                                     TES00660
0.101937                                                     TES00670
0.101937                                                     TES00680
0.101937                                                     TES00690
-0.024723                                                     TES0070
-0.024723                                                     TES0071
-0.024723                                                     TES0072
-0.024723                                                     TES0073
-0.024723                                                     TES0074
-0.024723                                                     TES0075
-0.024723                                                     TES0076
-0.024723                                                     TES0077
-0.024723                                                    TES00780
-0.024723                                                     TES0079
-0.024723                                                     TES0080
-0.024723                                                     TES0081
-0.024723                                                     TES0082
-0.024723                                                     TES0083
-0.024723                                                     TES0084
-0.024723                                                     TES0085
-0.024723                                                     TES0086
-0.024723                                                     TES0087
-0.024723                                                     TES0088
-0.024723                                                     TES0089
0.3068
0.6136
0.9204
1.2272
1.5340
1.8408
2.1476
2.4544
2.7612
3.0680
3.3748
3.6816
3.9884                                                       TES01020
4.2952                                                       TES01030
4.6020                                                       TES01040
4.9088                                                       TES01050
5.2156                                                       TES01060
5.5224                                                       TES01070
5.8292                                                       TES01080
6.1360                                                       TES01090
6.4428                                                       TES01100
6.7496                                                       TES01110
```

```
7.0564                                                                   TES01120
7.3632                                                                   TES01130
7.6700                                                                   TES01140
7.9768                                                                   TES01180
8.2836                                                                   TES01190
8.5904                                                                   TES01200
8.8972                                                                   TES01210
9.2040                                                                   TES01220
9.5108
9.8176
10.1244
10.4312
10.7380
11.0448                                                                  TES01230
11.3516                                                                  TES01240
11.6584                                                                  TES01250
11.9652                                                                  TES01260
12.2720                                                                  TES01270
12.5788                                                                  TES01280
12.8856                                                                  TES01290
13.1924                                                                  TES01300
13.4992                                                                  TES01310
13.8060                                                                  TES01320
14.1128                                                                  TES01330
14.4196                                                                  TES01340
14.7264                                                                  TES01350
15.0332                                                                  TES01360
15.3400                                                                  TES01370
15.6468                                                                  TES01380
15.9536                                                                  TES01390
16.2604                                                                  TES01400
16.5672                                                                  TES01420
16.8740                                                                  TES01430
17.1808                                                                  TES01440
17.4876
17.7944
18.1012
18.4080
18.7148
19.0216
19.3284
19.6352
19.9420
20.2488
20.5556
20.8624
21.1692
21.4760
21.7828
22.0896
22.3964
22.7032
23.0100
23.3168
23.6236
```

```
23.9304
24.2372
24.5440
24.8508
25.1576
25.4644
25.7712
26.0780
26.3848
26.6916
26.9984
27.3052
27.6120
27.9188
28.2256
28.5324
28.8392
29.1460
29.4528
29.7596
30.0664
30.3732
30.6800
30.9868                                                    TES00900
31.2936                                                    TES00910
31.6004                                                    TES00920
31.9072                                                    TES00930
32.2140                                                    TES00940
32.5208                                                    TES00950
32.8276                                                    TES00960
33.1344                                                    TES00970
33.4412                                                    TES00980
33.7480                                                    TES00990
34.0548                                                    TES01000
34.3616                                                    TES01010
34.6684                                                    TES01020
34.9752                                                    TES01030
35.2820                                                    TES01040
35.5888                                                    TES01050
35.8956                                                    TES01060
36.2024                                                    TES01070
36.5092                                                    TES01080
36.8160                                                    TES01090
37.1228                                                    TES01100
37.4296                                                    TES01110
37.7364                                                    TES01120
38.0432                                                    TES01130
38.3500                                                    TES01140
38.6568                                                    TES01180
38.9636                                                    TES01190
39.2704                                                    TES01200
39.5772                                                    TES01210
39.8840                                                    TES01220
40.1908
40.4976
```

```
40.8044
41.1112
41.4180
41.7248                                                                   TES01230
42.0316                                                                   TES01240
42.3384                                                                   TES01250
42.6452                                                                   TES01260
42.9520                                                                   TES01270
43.2588                                                                   TES01280
43.5656                                                                   TES01290
43.8724                                                                   TES01300
44.1792                                                                   TES01310
44.4860                                                                   TES01320
44.7928                                                                   TES01330
45.0996                                                                   TES01340
45.4064                                                                   TES01350
45.7132                                                                   TES01360
46.0200                                                                   TES01370
46.3268                                                                   TES01380
46.6336                                                                   TES01390
46.9404                                                                   TES01400
47.2472                                                                   TES01410
47.5540                                                                   TES01420
47.8608                                                                   TES01430
48.1676                                                                   TES01440
48.4744
48.7812
49.0880
49.3948
49.7016
50.0084
50.3152
50.6220
50.9288
51.2356
51.5424
51.8492
52.1560
52.4628
52.7696
53.0764
53.3832
53.6900
53.9968
54.3036
54.6104
54.9172
55.2240
55.5308
55.8376
56.1444
56.4512
56.7580
57.0648
57.3716
```

```
57.6784
57.9852
58.2920
58.5988
58.9056
59.2124
59.5192
59.8260
60.1328
60.4396
60.7464
61.0532
61.3600
```

```
RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS    8888888888    3333333333    2222222222     00000000
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  888888888888  333333333333  222222222222  0000000000
RR        RR  SS        SS  CC        CC  SS        SS  88        88  33        33  22        22  00      0000
RR        RR  SS            CC            SS            88        88            33            22  00     00 00
RR        RR  SSS           CC            SSS            88      88             33            22  00    00  00
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS      88888888           3333            22   00   00   00
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS     88888888           3333          22     00  00    00
RR    RR              SSS   CC                    SSS  88        88            33        22       00 00     00
RR     RR              SS   CC                     SS  88        88            33      22         0000      00
RR      RR    SS       SS   CC        CC  SS       SS  88        88  33        33    22           000       00
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  888888888888  333333333333  222222222222   0000000000
RR        RR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS    8888888888    3333333333   222222222222    00000000
```

```
EEEEEEEEEEEE  NN        NN  DDDDDDDDD                    8888888888    3333333333    2222222222     00000000
EEEEEEEEEEEE  NNN       NN  DDDDDDDDDD                  888888888888  333333333333  222222222222  0000000000
EE            NNNN      NN  DD       DD                 88        88  33        33  22        22  00      0000
EE            NN NN     NN  DD        DD                88        88            33            22  00     00 00
EE            NN  NN    NN  DD        DD                 88      88             33            22  00    00  00
EEEEEEEE      NN   NN   NN  DD        DD                  88888888           3333            22   00   00   00
EEEEEEEE      NN    NN  NN  DD        DD                  88888888           3333          22     00  00    00
EE            NN     NN NN  DD        DD                 88      88             33       22       00 00     00
EE            NN      NNNN  DD        DD                88        88            33     22         0000      00
EE            NN       NNN  DD       DD                 88        88  33        33   22           000       00
EEEEEEEEEEEE  NN        NN  DDDDDDDDDD                  888888888888  333333333333  222222222222   0000000000
EEEEEEEEEEEE  NN         N  DDDDDDDDD                    8888888888    3333333333   222222222222    00000000
```

```
************************************************************************************************************************************
************************************************************************************************************************************
************************************************************************************************************************************
************************************************************************************************************************************
```

Final Output Graphs from SPLOTN2 FORTRAN A1

The El Centro horizontal component of the acceleration record and the response to it in the hydrodynamic pressures (hydrodynamic pressure history) and other graphs.

α_R = 0.00 (full damping)
NT = 4096 (number of points in the acceleration record and also number of steps used in the FFT analysis)
Δw = 0.3068 rad/s (frequency step)

The El Centro component was recorded during the Imperial Valley earthquake in California on May 18, 1940. It had a magnitude of M6.6. The measured peak acceleration was PHA = -3.41705 m/s^2 and number of points in the record was equal to 2687.

ALEXNMP

JABLONSK

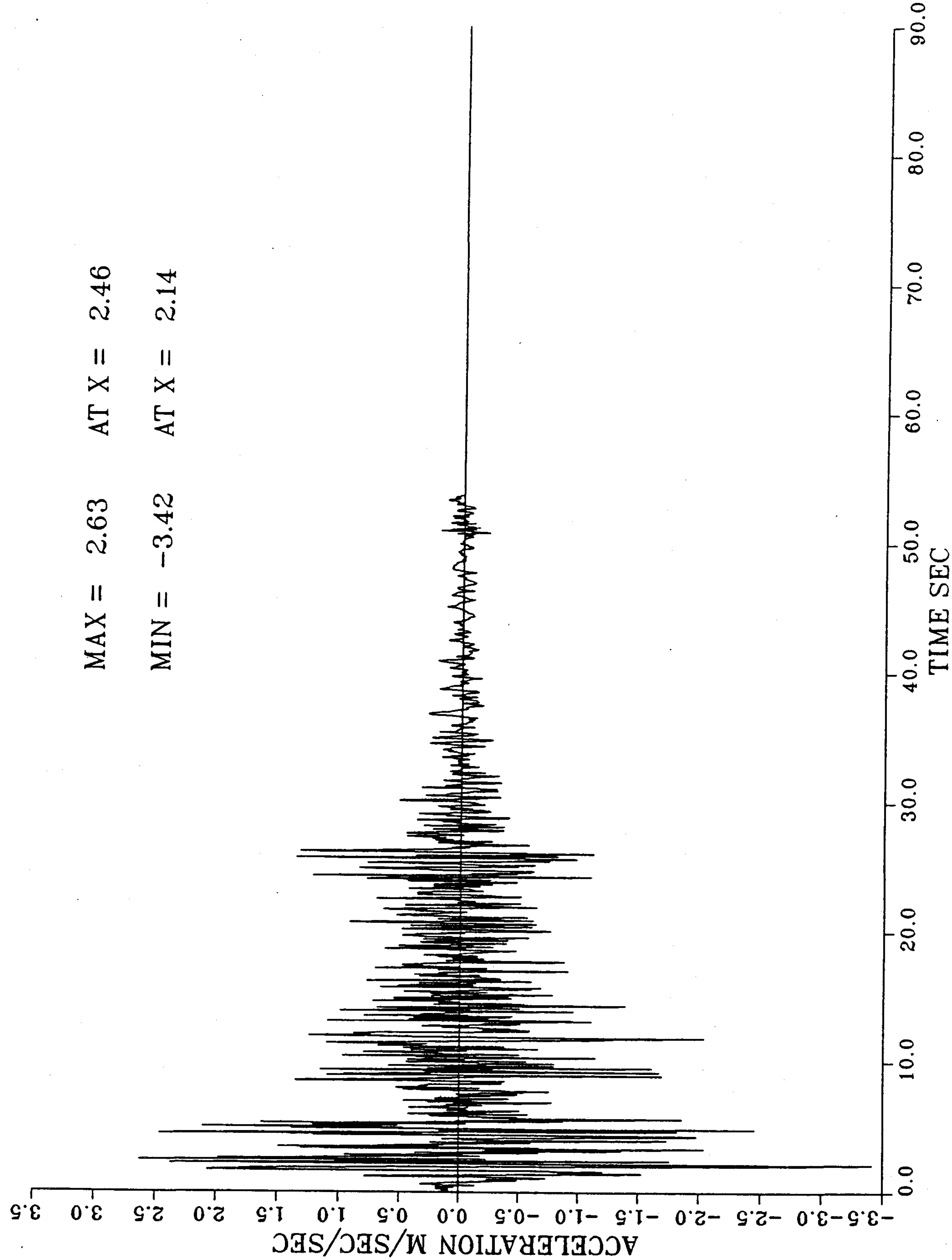
TAG DATA: DR. ALEXANDER M. JABLONSKI RM 151 M-20

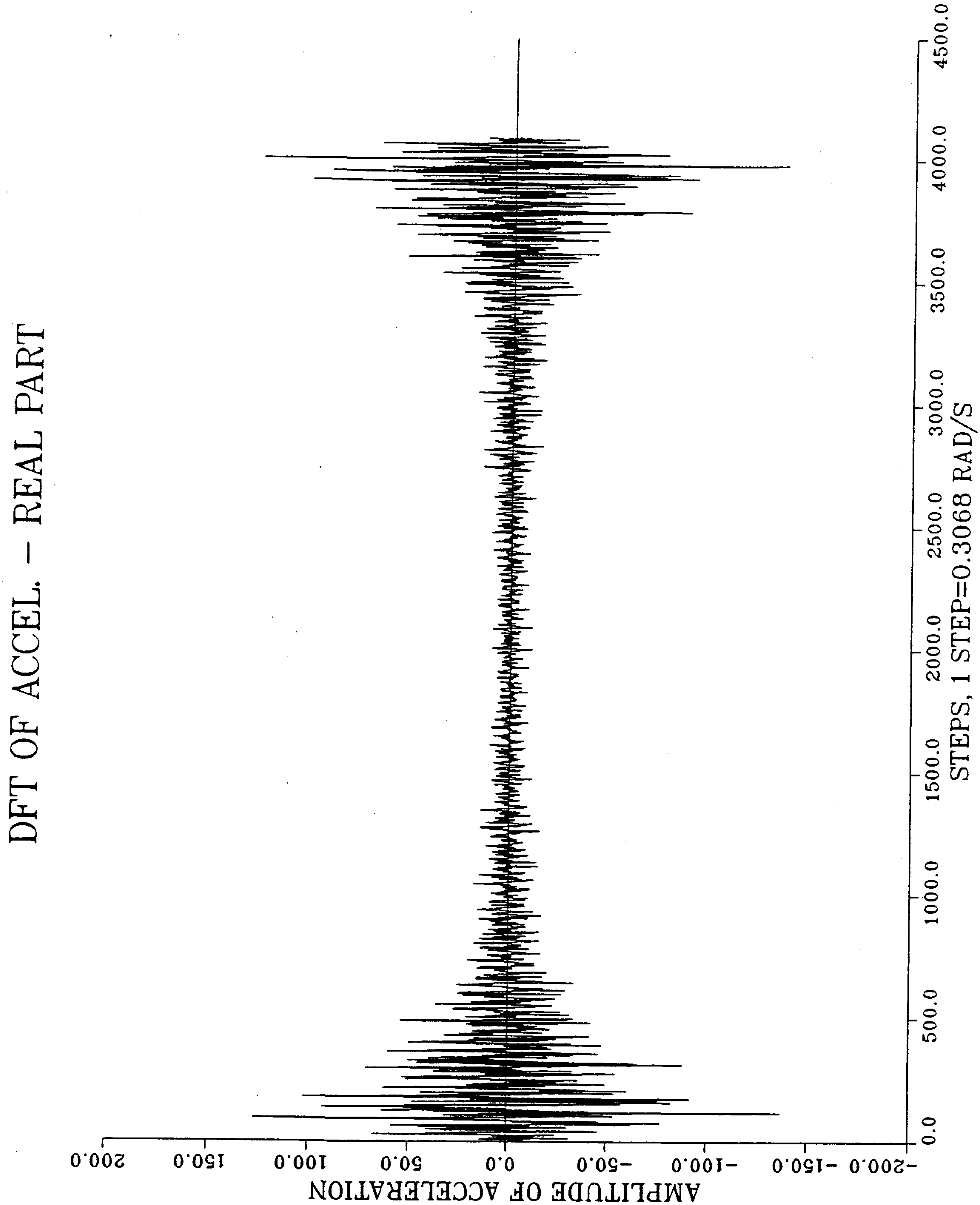
■■■■■■ NRC/CSB VM/CMS ■■■

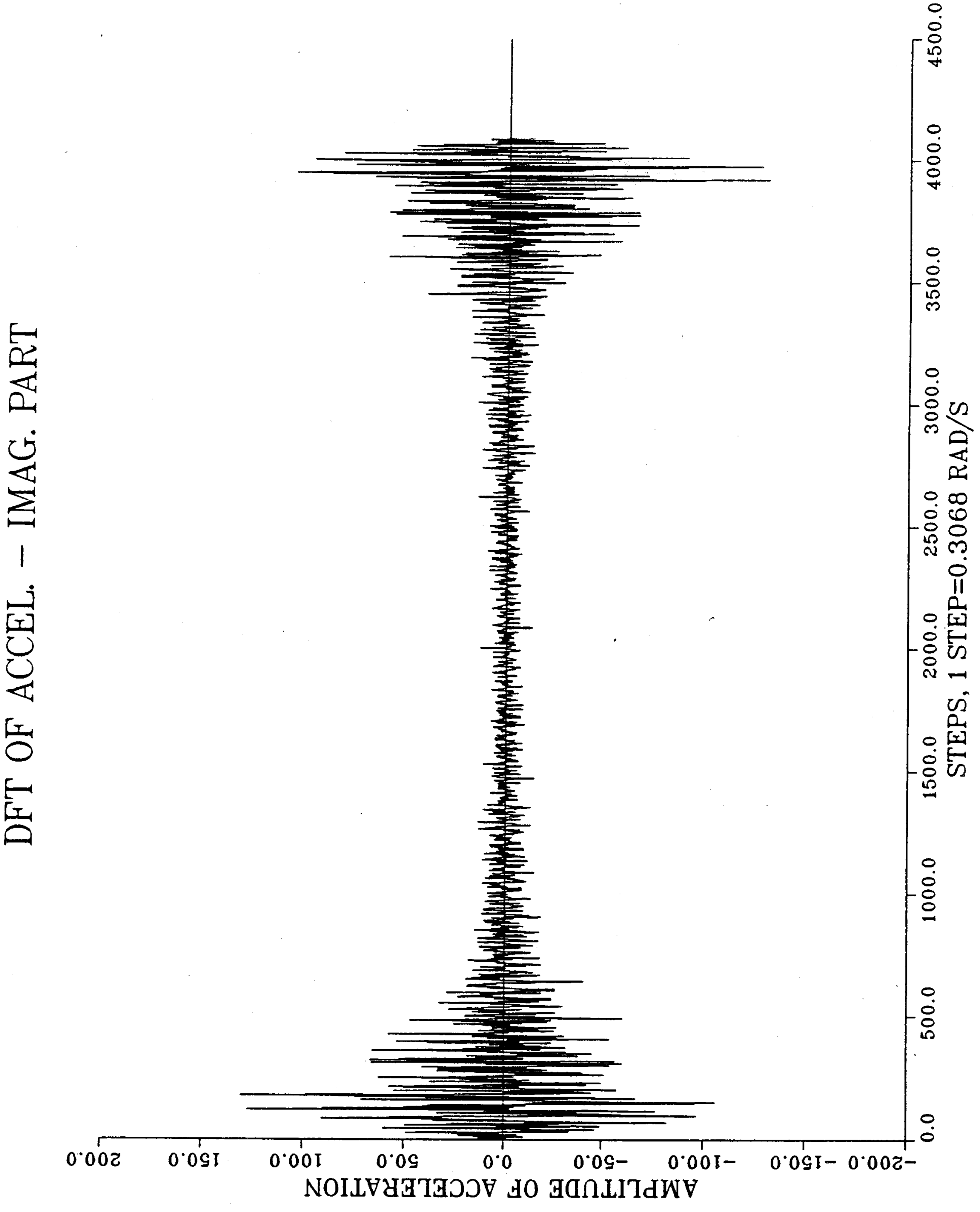
USERID:	ALEXNMP	DEVICE:	221LASER
DIST:	JABLONSK	TYPE:	PLOT-POP <THICKPEN 2>
FILEID:	DUM2 PLOTFILE A4	FORM:	PAPER11
SPOOLID:	4366	CLASS:	A
SYSTEM:	NRCVM01	COPY:	1
SUBMITTED:	4 SEP 1990 15:40:16	STARTED:	90/09/04 15:40:32

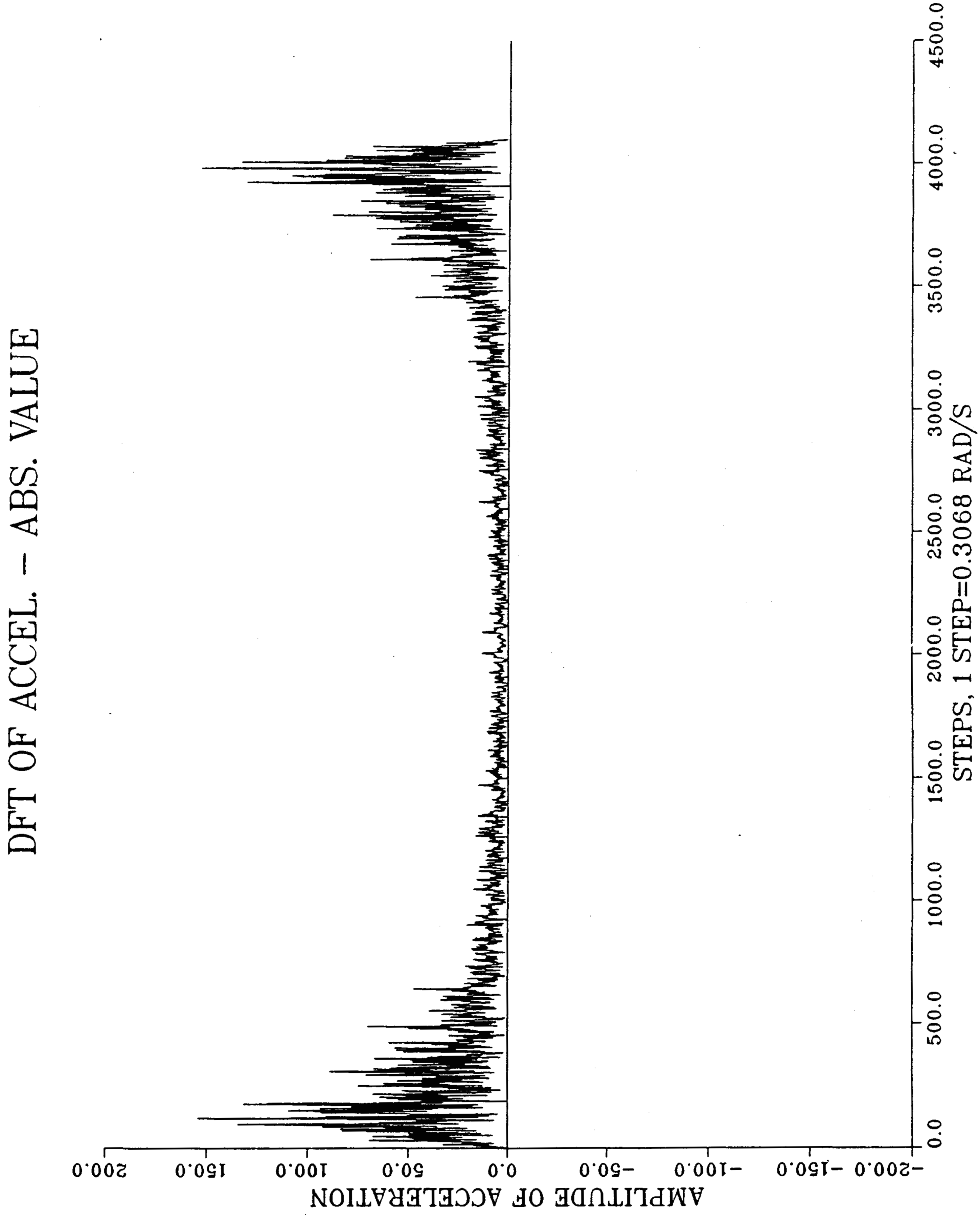
■■

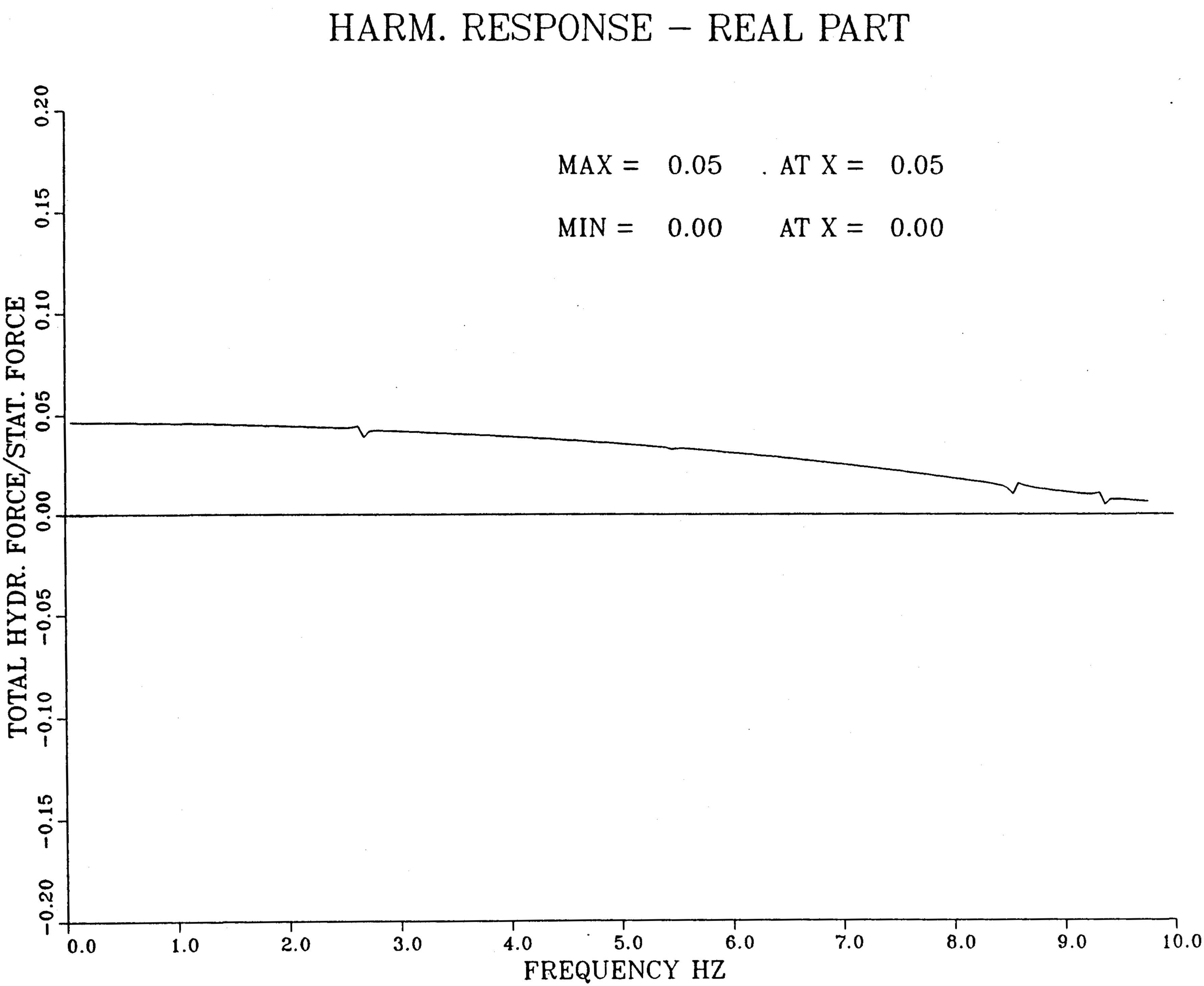
EL CENTRO HOR

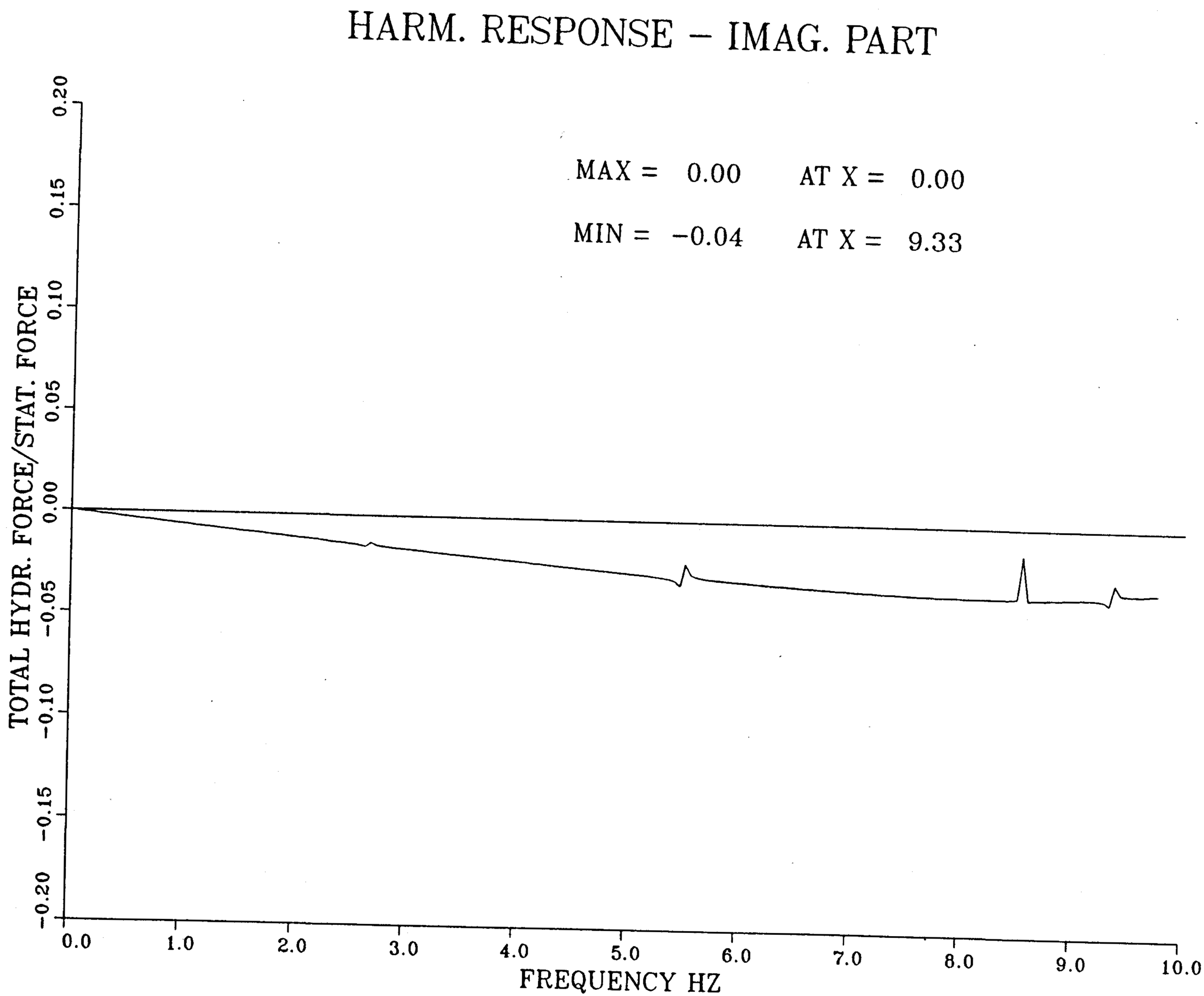


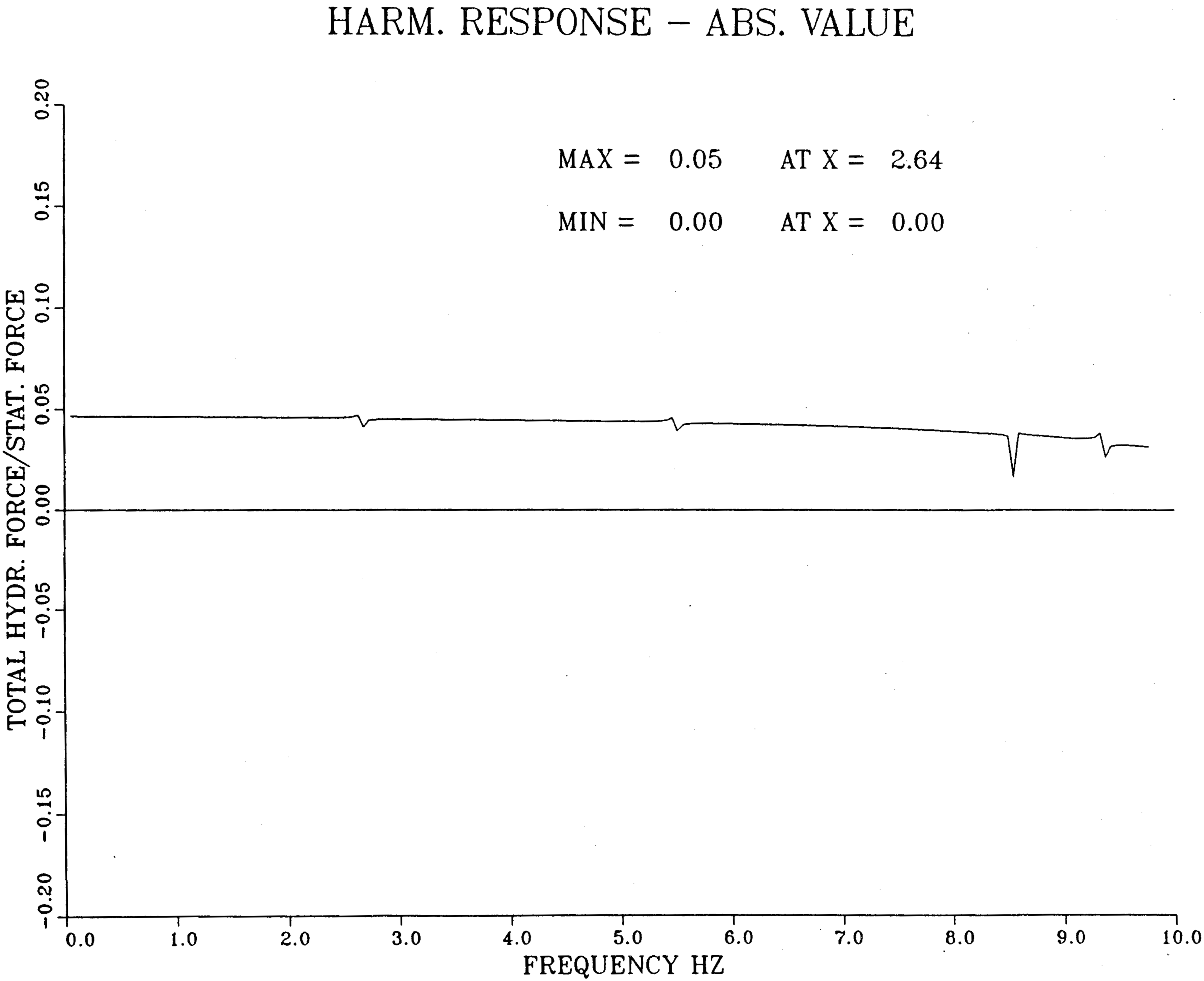

DFT OF ACCEL. - REAL PART
STEPS, 1 STEP=0.3068 RAD/S
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0
3500.0
4000.0
4500.0
AMPLITUDE OF ACCELERATION
-200.0
-150.0
-100.0
-50.0
0.0
50.0
100.0
150.0
200.0

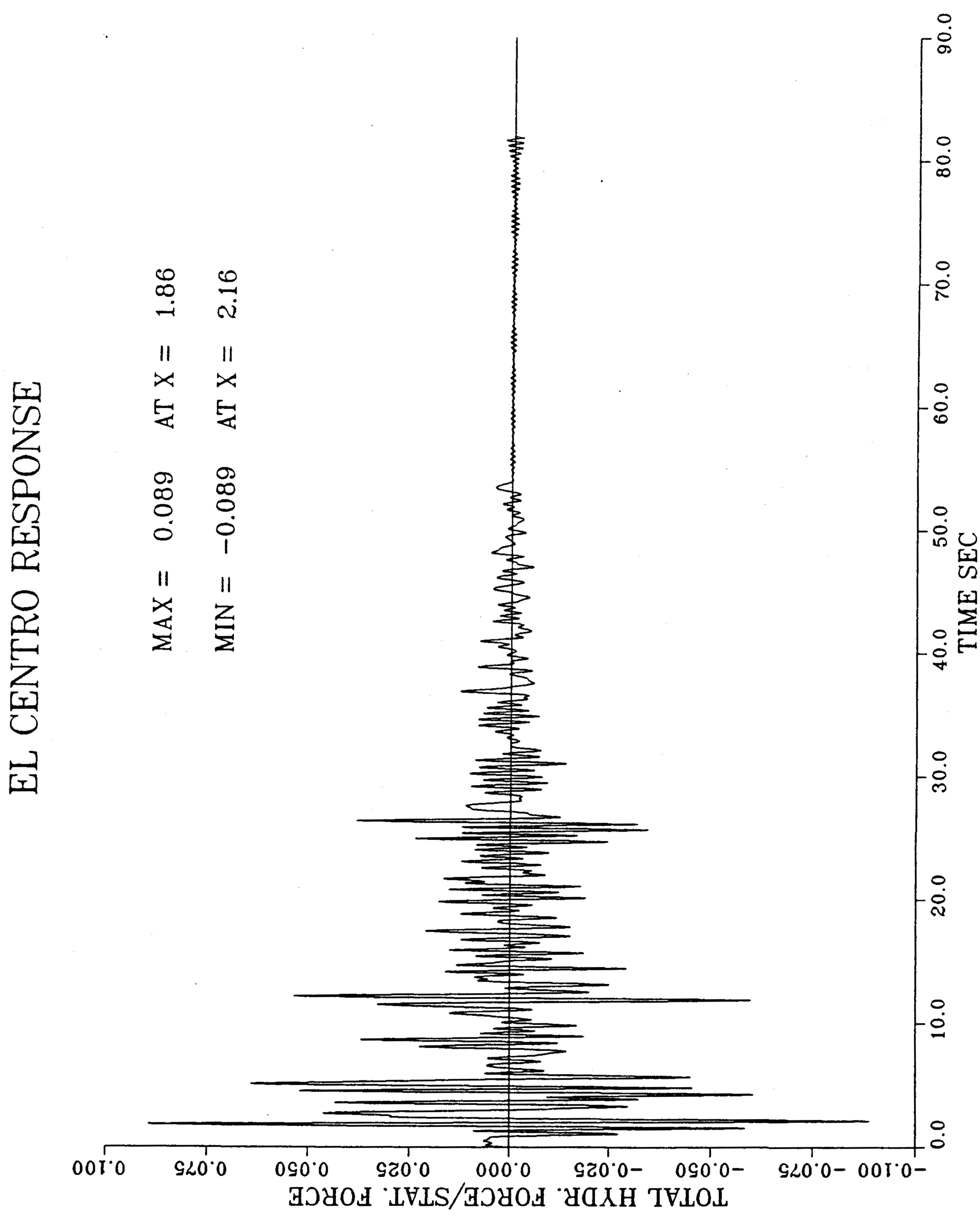

DFT OF ACCEL. – IMAG. PART
AMPLITUDE OF ACCELERATION
200.0
150.0
100.0
50.0
0.0
-50.0
-100.0
-150.0
-200.0
STEPS, 1 STEP=0.3068 RAD/S
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0
3500.0
4000.0
4500.0


DFT OF ACCEL. - ABS. VALUE
AMPLITUDE OF ACCELERATION
200.0
150.0
100.0
50.0
0.0
-50.0
-100.0
-150.0
-200.0
STEPS, 1 STEP=0.3068 RAD/S
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0
3500.0
4000.0
4500.0


HARM. RESPONSE - REAL PART
MAX = 0.05 . AT X = 0.05
MIN = 0.00 AT X = 0.00
TOTAL HYDR. FORCE/STAT. FORCE
0.20
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.20
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
FREQUENCY HZ


HARM. RESPONSE – IMAG. PART
MAX = 0.00 AT X = 0.00
MIN = −0.04 AT X = 9.33
TOTAL HYDR. FORCE/STAT. FORCE
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
FREQUENCY HZ


HARM. RESPONSE – ABS. VALUE
MAX = 0.05 AT X = 2.64
MIN = 0.00 AT X = 0.00
TOTAL HYDR. FORCE/STAT. FORCE
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
FREQUENCY HZ


EL CENTRO RESPONSE
MAX = 0.089 AT X = 1.86
MIN = -0.089 AT X = 2.16
TIME SEC
TOTAL HYDR. FORCE/STAT. FORCE
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
-0.100
-0.075
-0.050
-0.025
0.000
0.025
0.050
0.075
0.100

APPENDIX B

LISTINGS OF COMPUTER PROGRAMS

B.1 Program ACCELN FORTRA A1

```
RRRRRRRRRRR    MM          MM                            11      555555555555      11
RRRRRRRRRRRR   MMM        MMM                           111      555555555555     111
RR         RR  MMMM      MMMM                          1111      55              1111
RR         RR  MM MM    MM MM                            11      55                11
RR         RR  MM  MMMM    MM                            11      55                11
RRRRRRRRRRRR   MM    MM    MM                            11      555555555         11
RRRRRRRRRRR    MM          MM                            11      5555555555        11
RR     RR      MM          MM                            11              55        11
RR      RR     MM          MM                            11               55       11
RR       RR    MM          MM                            11               55       11
RR        RR   MM          MM                    1111111111      555555555555  1111111111
RR         RR  MM          MM                    1111111111      55555555555   1111111111

                      MM         MM          2222222222     00000000
                      MMM       MMM         222222222222   0000000000
                      MMMM     MMMM         22        22  00       0000
                      MM MM   MM MM                   22  00      00 00
                      MM  MMMM   MM                   22  00     00  00
                      MM    MM   MM                  22   00    00   00
                      MM         MM                22     00   00    00
                      MM         MM              22       00 00      00
                      MM         MM            22         0000       00
                      MM         MM          22           000        00
                      MM         MM         222222222222   0000000000
                      MM         MM         222222222222    00000000

  JJJJJJJJJ   AAAAAAAAAA   BBBBBBBBBBB   LL           000000000000  NN         NN   SSSSSSSSSS   KK         KK
  JJJJJJJJJ  AAAAAAAAAAAA  BBBBBBBBBBBB  LL           000000000000  NNN        NN  SSSSSSSSSSSS  KK        KK
       JJ    AA        AA  BB         BB LL           00        00  NNNN       NN  SS         SS KK       KK
       JJ    AA        AA  BB         BB LL           00        00  NN NN      NN  SS            KK      KK
       JJ    AA        AA  BB        BB  LL           00        00  NN  NN     NN  SSS           KK     KK
       JJ    AAAAAAAAAAAA  BBBBBBBBBB    LL           00        00  NN   NN    NN   SSSSSSSSS    KKKKKKK
       JJ    AAAAAAAAAAAA  BBBBBBBBBB    LL           00        00  NN    NN   NN    SSSSSSSSS   KKKKKKK
       JJ    AA        AA  BB        BB  LL           00        00  NN     NN NN           SSS   KK     KK
JJ     JJ    AA        AA  BB         BB LL           00        00  NN      NNNN            SS   KK      KK
JJ     JJ    AA        AA  BB         BB LL           00        00  NN       NNN   SS         SS KK       KK
JJJJJJJJ     AA        AA  BBBBBBBBBBBB  LLLLLLLLLLLL 000000000000  NN         NN  SSSSSSSSSSSS  KK        KK
 JJJJJJ      AA        AA  BBBBBBBBBBB   LLLLLLLLLLLL 000000000000  NN          N   SSSSSSSSSS   KK         KK

RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS     00000000      3333333333    8888888888       11
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS   0000000000    333333333333  888888888888     111
RR         RR SS         SS CC         CC SS         SS 00       0000  33         33 88         88   1111
RR         RR SS            CC            SS            00      00 00             33 88         88     11
RR         RR SSS           CC            SSS           00     00  00             33  88       88      11
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS    00    00   00          3333    88888888        11
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS   00   00    00          3333    88888888        11
RR     RR             SSS   CC                    SSS   00 00      00             33  88       88      11
RR      RR             SS   CC                     SS   0000       00             33 88         88     11
RR       RR   SS        SS  CC         CC SS       SS   000        00  33         33 88         88     11
RR        RR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS   0000000000    333333333333  888888888888  1111111111
RR         RR  SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS     00000000      3333333333    8888888888   1111111111
```

DATE/TIME= SEP 14, 1990 13:39:12 | PRINT: CLASS= E | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP | FORMS= S1PT | RM. 151
NODES D\E\C=NRCVM01 \NRCVM01 \NRCVM01 | FCB= STD8 | M-20
JOB = J 381 START | UCS= GT15 |

```
C     SUBROUTINE ACCEL(NT,DT,AC,A)
C
C     PROGRAMMER: ALEXANDER M. JABLONSKI
C
C     DATE: JULY 5, 1989
C     MODIFIED: JULY 6, 1989
C
C     CENTER: STRUCTURES SECTION
C             INSTITUTE FOT RESEARCH IN CONSTRUCTION
C             NATIONAL RESEARCH COUNCIL, CANADA
C
C     NOTE: THIS SUBROUTINE READS A PROCESSED EARTHQUAKE DATA
C           AND CONVERTS IT INTO INPUT DATA FILE FOR USE IN
C           FFT PROGRAM FOURT21
C
C
      PROGRAM ACCELN
      PARAMETER(NT=4096)
      PARAMETER(NI=0)
      PARAMETER(NL=2324)
C     NT=NUMBER OF TIME STEP IN FFT
C     NI=INITIAL STEP NUMBER TO BE READ FROM THE RECORD FILE
C     NL=LAST STEP NUMBER TO BE READ FROM THE SAME RECORD
C
      REAL AC(9000)
      REAL A(9000)
C
      CHARACTER*22 DATAS,INPS1
C
      DATA DATAS/'/DATAS INPUT A1'/
      DATA INPS1/'/INPS1 INPUT A1'/
      LEC=2
      IMP=3
      OPEN(LEC,FORM='FORMATTED',STATUS='OLD',FILE=DATAS)
      OPEN(IMP,FORM='FORMATTED',STATUS='NEW',FILE=INPS1)
C
C     READ DATA FROM INPUT FILE DATAS
C
C
C
      DO 10 I=1,NL
      READ(LEC,200)AC(I)
   10 CONTINUE
  200 FORMAT(E13.6)
C
      DO 11 I=1,NT
      A(I)=0.0
   11 CONTINUE
C
      DO 20 I=1,NT
      J=I+NI
      IF(J.GT.NL) THEN
      A(I)=0.0
      GO TO 20
      END IF
```

```
      A(I)=AC(J)                                                        ACC00560
   20 CONTINUE                                                          ACC00570
C                                                                       ACC00580
      NN=1+(NT/8)                                                       ACC00590
      DO 30 J=1,NN                                                      ACC00600
      NX=(J-1)*8                                                        ACC00610
      WRITE(IMP,100)(A(I+NX),I=1,8)                                     ACC00620
   30 CONTINUE                                                          ACC00630
  100 FORMAT(8F9.3)                                                     ACC00640
      STOP                                                              ACC00650
      END                                                               ACC00660
```

```
RRRRRRRRRRR   SSSSSSSSSS   CCCCCCCCCC   SSSSSSSSSS    00000000    3333333333   8888888888      11
RRRRRRRRRRRR SSSSSSSSSSSS CCCCCCCCCCCC SSSSSSSSSSSS  0000000000  333333333333 888888888888    111
RR        RR SS        SS CC        CC SS        SS 00      0000 33        33 88        88   1111
RR        RR SS           CC           SS           00     00 00           33 88        88     11
RR        RR SSS          CC           SSS          00    00  00           33  88      88      11
RRRRRRRRRRRR  SSSSSSSSS   CC            SSSSSSSSS   00   00   00        3333    88888888       11
RRRRRRRRRRR    SSSSSSSSS  CC             SSSSSSSSS  00  00    00        3333    88888888       11
RR    RR             SSS  CC                   SSS  00 00     00           33  88      88      11
RR     RR             SS  CC                    SS  0000      00           33 88        88     11
RR      RR   SS       SS  CC        CC SS       SS  000       00 33        33 88        88     11
RR       RR  SSSSSSSSSSSS CCCCCCCCCCCC SSSSSSSSSSSS  0000000000  333333333333 888888888888 1111111111
RR        RR  SSSSSSSSSS   CCCCCCCCCC   SSSSSSSSSS    00000000    3333333333   8888888888  1111111111
```

```
EEEEEEEEEEEE NN        NN DDDDDDDDD      3333333333   8888888888      11
EEEEEEEEEEEE NNN       NN DDDDDDDDDD   333333333333 888888888888    111
EE           NNNN      NN DD       DD  33        33 88        88   1111
EE           NN NN     NN DD        DD           33 88        88     11
EE           NN  NN    NN DD        DD           33  88      88      11
EEEEEEEE     NN   NN   NN DD        DD        3333    88888888       11
EEEEEEEE     NN    NN  NN DD        DD        3333    88888888       11
EE           NN     NN NN DD        DD           33  88      88      11
EE           NN      NNNN DD        DD           33 88        88     11
EE           NN       NNN DD       DD  33        33 88        88     11
EEEEEEEEEEEE NN        NN DDDDDDDDDD   333333333333 888888888888 1111111111
EEEEEEEEEEEE NN         N DDDDDDDDD      3333333333   8888888888  1111111111
```

```
********************************************************************************************************************************************
********************************************************************************************************************************************
********************************************************************************************************************************************
********************************************************************************************************************************************
```

B.2 Program BEMC2DN FORTRAN A1

```
RRRRRRRRRRR   MM        MM                        11      555555555555     11
RRRRRRRRRRRR  MMM      MMM                       111      555555555555    111
RR        RR  MMMM    MMMM                      1111      55             1111
RR        RR  MM MM  MM MM                        11      55               11
RR        RR  MM  MMMM  MM                        11      55               11
RRRRRRRRRRRR  MM   MM   MM                        11      555555555        11
RRRRRRRRRRR   MM        MM                        11      5555555555       11
RR    RR      MM        MM                        11              55       11
RR     RR     MM        MM                        11               55      11
RR      RR    MM        MM                        11               55      11
RR       RR   MM        MM                    1111111111  555555555555  1111111111
RR        RR  MM        MM                    1111111111  55555555555   1111111111
```

```
MM        MM      2222222222    00000000
MMM      MMM     222222222222  0000000000
MMMM    MMMM     22        22  00      0000
MM MM  MM MM               22  00     00 00
MM  MMMM  MM               22  00    00  00
MM   MM   MM              22   00   00   00
MM        MM            22     00  00    00
MM        MM          22       00 00     00
MM        MM        22         0000      00
MM        MM      22           000       00
MM        MM     222222222222  0000000000
MM        MM     222222222222   00000000
```

```
  JJJJJJJJJ   AAAAAAAAAA   BBBBBBBBBBB   LL            OOOOOOOOOOOO  NN         NN   SSSSSSSSSS   KK        KK
  JJJJJJJJJ  AAAAAAAAAAAA  BBBBBBBBBBBB  LL            OOOOOOOOOOOO  NNN        NN  SSSSSSSSSSSS  KK       KK
      JJ     AA        AA  BB        BB  LL            OO        OO  NNNN       NN  SS        SS  KK      KK
      JJ     AA        AA  BB        BB  LL            OO        OO  NN NN      NN  SS            KK     KK
      JJ     AA        AA  BB       BB   LL            OO        OO  NN  NN     NN  SSS           KK    KK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB    LL            OO        OO  NN   NN    NN   SSSSSSSSS    KKKKKKK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB    LL            OO        OO  NN    NN   NN    SSSSSSSSS   KKKKKKK
      JJ     AA        AA  BB       BB   LL            OO        OO  NN     NN  NN          SSS   KK    KK
JJ    JJ     AA        AA  BB        BB  LL            OO        OO  NN      NNNN            SS   KK     KK
JJ    JJ     AA        AA  BB        BB  LL            OO        OO  NN       NNN   SS       SS   KK      KK
JJJJJJJJ     AA        AA  BBBBBBBBBBBB  LLLLLLLLLLLL  OOOOOOOOOOOO  NN        NN   SSSSSSSSSSSS  KK       KK
 JJJJJJ      AA        AA  BBBBBBBBBBB   LLLLLLLLLLLL  OOOOOOOOOOOO  NN         N    SSSSSSSSSS   KK        KK
```

```
RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS        444          444      2222222222    6666666666
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS      4444         4444     222222222222  666666666666
RR        RR  SS        SS  CC        CC  SS        SS     44 44        44 44     22        22  66        66
RR        RR  SS            CC            SS              44  44       44  44               22  66
RR        RR  SSS           CC            SSS            44   44      44   44               22  66
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS    4444444444   4444444444            22   6666666666
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS  44444444444  44444444444          22     66666666666
RR    RR              SSS   CC                    SSS       44           44            22       66        66
RR     RR              SS   CC                     SS       44           44          22         66        66
RR      RR    SS       SS   CC        CC  SS       SS       44           44        22           66        66
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS      44           44      222222222222   666666666666
RR        RR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS       44           44      222222222222    6666666666
```

```
DATE/TIME= JUL 31, 1990 11:21:02          | PRINT: CLASS= E       | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP                     |    FORMS= S1PT        | RM. 151
NODES O\E\C=NRCVM01 \NRCVM01 \NRCVM01     |    FCB=  STD8         | M-20
JOB = J4426 START                         |    UCS=  GT15         |
```

```
C
C     ******************************************************************
C
C     PROGRAM BEMC.2D.I.FTN
C
C     PROGRAM FOR THE SOLUTION OF A TWO-DIMENSIONAL RESERVOIR
C     PROBLEM WITH ONE INFINITE BOUNDARY AND DAMPING INCLUDED
C     BY CONSTANT BOUNDARY ELEMENT METHOD
C     SPECIAL B.C. AT FAR BOUNDARY THROUGH FEM, DAMPING INCLUDED
C
C     PROGRAMMER: ALEXANDER M. JABLONSKI
C
C     DATE: JULY 9, 1987,
C           SUBSTANTIALLY MODIFIED AUGUST 27, 1987
C           MODIFIED JUNE 27,1989
C           MODIFIED AUGUST 1, 1989
C
C     CENTER: CASCADE, CARLETON UNIVERSITY
C
C     NOTE: THIS PROGRAM IS BASED ON PROGRAM BEML.2D.I.FTN
C
C     ******************************************************************
C
C     N = NUMBER OF BOUNDARY ELEMENTS, READ THROUGH INPUT
C     X = ARRAY CONTAINING X COORDINATES OF THE NODES
C     Y = ARRAY CONTAINING Y COORDINATES OF THE NODES
C     XM = ARRAY CONTAINING X COORDINATES OF THE MIDPOINTS OF THE ELEMEN
C     YM = ARRAY CONTAINING Y COORDINATES OF THE MIDPOINTS OF THE ELEMEN
C     G = MATRIX IN THE BOUNDARY ELEMENT EQUATION
C     H = MATRIX IN THE BOUNDARY ELEMENT EQUATION
C     FID= ARRAY OF PRESCRIBED VALUES OF BOUNDARY CONDITIONS
C          ON THE DAM FACE
C     FIB= ARRAY OF PRESCRIBED VALUES OF BOUNDARY CONDITIONS
C          ON THE BASE
C     DFI = RIGHT HAND SIDE VECTOR IN BOUNDARY INTEGRAL EQUATION;
C           AFTER SOLUTION, IT CONTAINS THE VALUES AT THE BOUNDARY
C           THAT WERE PREVIOUSLY UNKNOWN
C     LEC = UNIT NO. CONTAINING INPUT
C     IMP = UNIT NO. ON WHICH OUTPUT IS WRITTEN
C     NX = 2*N = DIMENSION OF THE SYSTEM OF EQUATIONS
C     NU = DIMENSION OF H AND G MATRICES IN THE MAIN PROGRAM
C     ND =  DIMENSION OF THE SPECIAL BOUNDARY CONDITION MATRIX AT
C           FAR BOUNDARY IN THE MAIN PROGRAM
C     D = VALUE OF THE DETERMINANT OF COEFFICIENT MATRIX
C     CONST = OMEGA/C
C     OMEGA = EXCITING FREQUENCY
C     C = VELOCITY OF SOUND IN WATER
C     CONST1=OMEGA
C     CONST2=FOUNDATION DAMPING COEFFICIENT GAMMA
C     AFR=REAL PART OF EIGENVECTOR SI
C     AFI=IMAGINARY PART OF EIGENVECTOR SI
C     AKR=REAL PART OF KAPPA MATRIX
C     AKI=IMAGINARY PART OF KAPPA MATRIX
C     NO=LAST ELEMENT NUMBER AT TOP OF RESERVOIR NEAR THE DAM
C     N1=LAST ELEMENT NUMBER AT THE BASE OF THE DAM
```

```
C     N2=LAST ELEMENT NUMBER AT THE BASE NEAR FAR END
C     MF=NUMBER OF ELEMENTS ON BASE
C     ME=NUMBER OF ELEMENTS ON THE FAR BOUNDARY
C     HY=HEIGHT OF THE RESERVOIR
C
      PROGRAM BEMC2D
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      DOUBLE PRECISION X(101),Y(101),G(200,200),H(200,200),
     &DFI(200),FID(50),FIB(50),HOLD(200,200),XM(101),YM(101)
      REAL AFI(20,20),AFR(20,20),AKR(20),AKI(20)
      REAL YR(20,500),YI(20,500)
      COMPLEX YC(20,500),TX(500),HX(500)
      LOGICAL FLAG
      CHARACTER*22 TESTM,OUTM,EIGENM,INFTM
      DATA TESTM /'/TESTM INPUT A1'/
      DATA OUTM /'/OUTM OUTPUT C'/
      DATA EIGENM /'/EIGENM OUTPUT C'/
      DATA INFTM /'/INFTM INPUT A1'/
      LEC=2
      IMP=3
      IWRITE=4
      ND=20
      NU=200
      PI=3.141592653
      OPEN (LEC,FORM='FORMATTED',STATUS='OLD',FILE=TESTM)
      OPEN (IMP,FORM='FORMATTED',STATUS='NEW',FILE=OUTM)
      OPEN (IWRITE,FORM='FORMATTED',STATUS='NEW',FILE=EIGENM)
      OPEN (7,FORM='FORMATTED',STATUS='NEW',FILE=INFTM)
C
C     READ DATA FROM INPUT FILE
C
      CALL INPUT(X,Y,FID,FIB,N0,N1,N2,NOUT,C,DF)
      NX=2*N
      NC=0
    5 WRITE(IMP,300)
      READ(LEC,*,END=15)OMEGA
      NC=NC+1
      CONST=OMEGA/C
      WRITE(IMP,350)OMEGA,CONST
      CONST1=OMEGA
C
C     COMPUTE EIGENVALUES AND EIGENVECTORS FOR THE FAR BOUNDARY
C
      IF(ME.GT.0)THEN
      IF(CONST2.GT.1.0E-08)THEN
      FLAG=.TRUE.
      CALL EIGEN(AFI,AFR,ND,ME,CONST,CONST1,CONST2,HY,AKR,AKI)
      ELSE
      CALL EIGENR(AFR,ND,ME,CONST,CONST1,CONST2,HY,AKR,AKI,IMP,FLAG)
      END IF
*     WRITE(6,2001)
*2001 FORMAT(' ')
```

```
*      DO 10 I=1,ME                                                      BEM01110
*      WRITE(6,2010)AKR(I),AKI(I)                                        BEM01120
*  10  CONTINUE                                                          BEM01130
*      write (6,2001)                                                    BEM01140
*      do 2003 i=1,me                                                    BEM01150
*      write(6,2010)(afr(i,j),j=1,me)                                    BEM01160
* 2003 continue                                                          BEM01170
*      write(6,2001)                                                     BEM01180
      END IF                                                             BEM01190
C                                                                        BEM01200
C     FORM SYSTEM OF EQUATIONS                                           BEM01210
C                                                                        BEM01220
C                                                                        BEM01230
      CALL FMAT(X,Y,XM,YM,G,H,HOLD,FID,FIB,DFI,N0,N1)                    BEM01240
*      do 2000 i=1,n                                                     BEM01250
*      write(6,2010)(h(i,j),j=1,n)                                       BEM01260
* 2000 continue                                                          BEM01270
*      write(6,2001)                                                     BEM01280
*      write(6,2001)                                                     BEM01290
*      do 2002 i=1,n                                                     BEM01300
*      write(6,2010)(g(i,j),j=1,n)                                       BEM01310
* 2002 continue                                                          BEM01320
C                                                                        BEM01330
C     MODIFY H MATRIX FOR BOUNDARY CONDITIONS AT BASE AND FAR END        BEM01340
C     AND COMPUTE DFI                                                    BEM01350
C                                                                        BEM01360
      IF(CONST2.GT.1.0E-08)THEN                                          BEM01370
      CALL ADDFBC(H,G,HOLD,AFR,AFI,AKR,AKI,N0,N1,N2,DFI,FID,FIB,DF)      BEM01380
      ELSE                                                               BEM01390
      CALL ADDUND(H,G,HOLD,AFR,AKR,AKI,N0,N1,N2,DFI,FID,FIB,FLAG,DF)     BEM01400
      END IF                                                             BEM01410
*      write(6,2001)                                                     BEM01420
*      write(6,2001)                                                     BEM01430
*      do 2005 i=1,nx                                                    BEM01440
*      write(6,2010)(h(i,j),j=1,nx)                                      BEM01450
* 2005 continue                                                          BEM01460
*      write(6,2001)                                                     BEM01470
*      write(6,2001)                                                     BEM01480
*      write(6,2010)(dfi(i),i=1,nx)                                      BEM01490
C                                                                        BEM01500
C                                                                        BEM01510
C     SOLVE SYSTEM OF EQUATIONS                                          BEM01520
C                                                                        BEM01530
      IF(FLAG)THEN                                                       BEM01540
       CALL SLNPD(H,DFI,D,NX,NU,IMP)                                     BEM01550
      ELSE                                                               BEM01560
       CALL SLNPD(H,DFI,D,N,NU,IMP)                                      BEM01570
      END IF                                                             BEM01580
*       DO 999 I=1,N                                                     BEM01590
*       WRITE(IMP,*)DFI(I)                                               BEM01600
*  999  CONTINUE                                                         BEM01610
*       IF(FLAG)THEN                                                     BEM01620
*       DO 1000 I=1,N                                                    BEM01630
*       WRITE(IMP,*)DFI(I+N)                                             BEM01640
* 1000  CONTINUE                                                         BEM01650
```

```
*        END IF
C
C
C      OUTPUT RESULTS
C
        CALL OUTPUT(XM,YM,FID,FIB,DFI,AFR,AFI,AKR,AKI,
     * N0,N1,N2,NOUT,FLAG,DF,NC,TPR,TPI,HXR,HXI,YR,YI)
        TX(NC)=CMPLX(TPR,TPI)
        HX(NC)=CMPLX(HXR,HXI)
        WRITE(7,*)NC,TX(NC),HX(NC)
        WRITE(7,300)
        DO 30 I=1,N1-N0
        YCR=YR(I,NC)
        YCI=YI(I,NC)
        YC(I,NC)=CMPLX(YCR,YCI)
        WRITE(7,*)YC(I,NC)
   30   CONTINUE
        GO TO 5
C
 2010 format(6e13.5)
  300 FORMAT(//,1X,70('*'),//)
  350 FORMAT(1X,'EXCITING FREQUENCY',E14.5,//,1X,
     *'WAVE CONSTANT',E14.5,//)
   15 STOP
      END
C
C      ------------------------------------------------------------------
C
      SUBROUTINE INPUT(X,Y,FID,FIB,N0,N1,N2,NOUT,C,DF)
C      ******************************************************************
C
C      READS INPUT DATA
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      DOUBLE PRECISION X(1),Y(1),FID(1),FIB(1),C
      CHARACTER*4 TITLE(18)
      WRITE (IMP,100)
C
C      READ TITLE
C
      READ (LEC,150)TITLE
C
C      READ BASIC PARAMETERS
C
      READ(LEC,*)N,N0,N1,N2,NOUT
      READ(LEC,*)HY,C,ALF,DF
      CONST2=((1-ALF)/(1+ALF))*(1/C)
C
      WRITE(IMP,300)N,N0,N1,N2,NOUT
      WRITE(IMP,350)HY,C,ALF,CONST2
C
C
C      READ COORDINATES OF NODAL POINTS OF THE
```

```
C     BOUNDARY ELEMENTS IN ARRAY X AND Y
C
      WRITE(IMP,500)
      DO 10 I=1,N
      READ(LEC,*)X(I),Y(I)
   10 WRITE(IMP,700)I,X(I),Y(I)
C
C
C     READ BOUNDARY CONDITION ON THE DAM FACE
C
      WRITE(IMP,760)
      WRITE(IMP,800)
      ILAST=N1-N0
      DO 20 I=1,ILAST
      READ(LEC,*)FID(I)
      WRITE(IMP,750)I,FID(I)
   20 CONTINUE
C
C
C     READ BOUNDARY CONDITION ON THE BASE OF THE RESERVOIR
C
      WRITE(IMP,770)
      WRITE(IMP,800)
      ILAST=N2-N1
      DO 30 I=1,ILAST
      READ(LEC,*)FIB(I)
      WRITE(IMP,750)I,FIB(I)
   30 CONTINUE
      ME=N-N2
      MF=N2-N1
C
  100 FORMAT(1X,60('*')//)
  150 FORMAT(18A4)
  300 FORMAT(1X,'DATA',//,1X,'NUMBER OF BOUNDARY ELEMENTS',
     *I3,//,1X,'ELEMENT NUMBER AT TOP OF RESERVOIR NEAR THE DAM',I3,//,
     *1X,'ELEMENT NUMBER AT BASE OF DAM',I3,//,1X,
     *'ELEMENT NUMBER AT BOTTOM NEAR FAR END',I3,//,1X,
     *'FLAG FOR DETAILED OUTPUT',I3,/)
  350 FORMAT(1X,'HEIGHT OF DAM',E14.5,//,1X,
     *'VELOCITY OF SOUND IN WATER',E14.5,//,1X,
     *'WAVE REFLECTION COEFFICIENT',E14.5,//,1X,
     *'FOUNDATION DAMPING COEFFICIENT',E14.5//)
  400 FORMAT(2(2X,E14.7,2X))
  500 FORMAT(' COORDINATES OF THE NODAL POINTS',
     *//,4X,'POINT',10X,'X',18X,'Y'/)
  700 FORMAT(5X,I3,2(5X,E14.7))
  800 FORMAT(5X,'NODE',6X,
     *'PRESCRIBED VALUE'/)
  750 FORMAT(5X,I3,8X,E14.7)
  760 FORMAT(//' BOUNDARY CONDITIONS ON THE DAM FACE',//)
  770 FORMAT(//' BOUNDARY CONDITIONS ON THE RESERVOIR BOTTOM',//)
C
C
      RETURN
      END
```

```
C
C     ------------------------------------------------------------------
C
      SUBROUTINE EIGEN(AFI,AFR,ND,ME,CONST,CONST1,CONST2,HY,AKR,AKI)
C     ******************************************************************
      DOUBLE PRECISION CONST,CONST1,CONST2,HY
      REAL AFI(ND,ND),AFR(ND,ND),AKR(ND),AKI(ND)
C
      DY=SNGL(HY/ME)
      T=SNGL(CONST)
      T1=SNGL(CONST1)
      T2=SNGL(CONST2)
C
      CALL FSFEM(AFI,AFR,ME,ND,T,T1,T2,DY,AKR,AKI)
C
      RETURN
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE FSFEM(AFI,AFR,N,NM,T,T1,T2,DY,AKR,AKI)
C     ******************************************************************
C
C     AFR= REAL PART OF THE EIGENVECTOR MATRIX
C     AFI= IMAGINARY PART OF EIGENVECTOR MATRIX
C     AKR=REAL PART OF KAPPA MATRIX DIAGONAL
C     AKI=IMAGINARY PART OF KAPPA MATRIX DIAGONAL
C     N=ME= NO. OF ELEMENTS ON THE FAR BOUNDARY
C     NM=ND= MAX. DIMENSION OF TNE EIGENVECTOR MATRIX
C     DY= PASSED LENGTH OF AN ELEMENT ( DP IN THE MAIN PROGRAM )
C     T=CONST= OMEGA/C ( DP IN THE MAIN PROGRAM )
C     T1=CONST1= OMEGA ( DP IN THE MAIN PROGRAM )
C     T2=CONST2=GAMMA
C
      REAL  A( 20, 20),B( 20, 20),Z( 20, 20),BHOLD( 20, 20),
     $         AHOLD( 20, 20),AHOLDI( 20, 20),BHOLDI( 20,20),
     $        ALFR( 20),ALFI( 20),BETA( 20),
     $        EPS1, AI(20,20),BI(20,20),ZI(20,20)
      REAL AMAX1
      COMPLEX FI(20,20),XM(20,20)
      COMPLEX XKAP(20),XK2(20)
      REAL AFR(NM,NM),AFI(NM,NM),AKR(NM),AKI(NM)
      INTEGER  ERROR
      DIMENSION INDEX(20)
      DATA IWRITE/4/
C
      CALL SET(DY,T1,T2,N,NM,A,AI,B,BI)
      DO 717 I=1,N
      DO 718 J=1,N
      AHOLD(I,J)=A(I,J)
      BHOLD(I,J)=B(I,J)
      AHOLDI(I,J)=AI(I,J)
      BHOLDI(I,J)=BI(I,J)
  718 CONTINUE
  717 CONTINUE
```

```
      WRITE(IWRITE,20)
     $             N
   20 FORMAT(30H1THE FULL MATRIX  A   OF ORDER,
     $       I4,22H  IS (PRINTED BY ROWS)/)
      DO 30 I = 1,N
   30    WRITE(IWRITE,40)
     $               (A(I,J),AI(I,J),J=1,N)
   40    FORMAT(5(2F6.2,1HI,3X))
      WRITE(IWRITE,41)
     $             N
   41 FORMAT(30H0THE FULL MATRIX  B   OF ORDER,
     $       I4,22H  IS (PRINTED BY ROWS)/)
      DO 42 I = 1,N
   42    WRITE(IWRITE,43)
     $               (B(I,J),BI(I,J),J=1,N)
   43    FORMAT(5(2F6.2,1HI,3X))
C
C
C
C
C     EIGENVALUES AND EIGENVECTORS USING  CQZVAL  AND  CQZVEC
C
         EPS1 = 0.0
         CALL  CQZHES(NM,N,A,AI,B,BI,.TRUE.,Z,ZI)
         CALL  CQZVAL(NM,N,A,AI,B,BI,EPS1,ALFR,ALFI,BETA,
     $                .TRUE.,Z,ZI,ERROR)
         CALL  CQZVEC(NM,N,A,AI,B,BI,ALFR,ALFI,BETA,Z,ZI)
      DO 719 I=1,N
      DO 720 J=1,N
      A(I,J)=AHOLD(I,J)
      B(I,J)=BHOLD(I,J)
      AI(I,J)=AHOLDI(I,J)
      BI(I,J)=BHOLDI(I,J)
  720 CONTINUE
  719 CONTINUE
         WRITE(IWRITE,292)
  292    FORMAT(/15X,7HALFR(I),19X,7HALFI(I),19X,7HBETA(I))
         DO 295 I = 1,N
            WRITE(IWRITE,293)
     $                   I,ALFR(I),ALFI(I),BETA(I)
  293       FORMAT(I2,3(1PE23.6,3X))
  295    CONTINUE
         WRITE(IWRITE,300)
  300 FORMAT(/14X,35HCOMPUTED EIGENVALUE AND EIGENVECTOR,20X)
C
         DO 510 K = 1, N
            BETA(K) = AMAX1(BETA(K),1.0E-50)
            ALFR(K) = ALFR(K) / BETA(K)
            ALFI(K) = ALFI(K) / BETA(K)
C
C        ONE EIGENVECTOR.
C
  340    WRITE(IWRITE,350)
     $               K,ALFR(K),ALFI(K),(Z(I,K),
     $               ZI(I,K),I=1,N)
```

```
  350    FORMAT(/I2,1P2E23.6/(5X,2E23.6))
  510 CONTINUE
      DO 610 K=1,N
      DO 620 L=1,N
      FI(K,L)=CMPLX(Z(K,L),ZI(K,L))
  620 CONTINUE
  610 CONTINUE
      DO 611 I=1,N
      WRITE(IWRITE,*) (FI(I,J),J=1,N)
  611 CONTINUE
      CALL NORMA(FI,B,N,NM,XM)
      DO 81 I=1,N
      WRITE(IWRITE,*) (FI(I,J),J=1,N)
   81 CONTINUE
      DO 613 I=1,N
      DO 612 J=1,N
      AFR(I,J)=REAL(FI(I,J))
      AFI(I,J)=AIMAG(FI(I,J))
  612 CONTINUE
  613 CONTINUE
      DO 171 I=1,N
      ALFR(I)=ALFR(I)-T*T
      XK2(I)=CMPLX(ALFR(I),ALFI(I))
      XKAP(I)=CSQRT(XK2(I))
      WRITE(IWRITE,*)XKAP(I)
      AKR(I)=REAL(XKAP(I))
      AKI(I)=AIMAG(XKAP(I))
  171 CONTINUE
      RETURN
      END
C
C      ------------------------------------------------------------------
C
      SUBROUTINE EIGENR(AFR,ND,ME,CONST,CONST1,CONST2,HY,AKR,AKI
     1 ,IMP,FLAG)
c     ******************************************************************
      DOUBLE PRECISION CONST,CONST1,CONST2,HY
      REAL AFR(ND,ND),AKR(ND),AKI(ND)
      REAL ALFR(20),A(20,20),F(20,20)
      LOGICAL FLAG
C
      DO 10 I=1,ME
      AKR(I)=0
      AKI(I)=0
   10 CONTINUE
      DY=SNGL(HY/ME)
      T=SNGL(CONST)
      T1=SNGL(CONST1)
      T2=SNGL(CONST2)
      FLAG=.FALSE.
      CALL SETR(DY,ME,ND,A,F)
      CALL JACOBI(A,F,ALFR,AFR,ME,ND)
      DO 30 I=1,ME
      TEMP=ALFR(I)-T*T
      TX=TEMP
```

```
      IF(TEMP.LT.0) THEN
      FLAG=.TRUE.
      AKI(I)=SQRT(-TX)
      ELSE IF(TEMP.GT.0) THEN
      AKR(I)=SQRT(TX)
      ELSE
      WRITE(IMP,100)
  100 FORMAT(1X,'RESONANCE HAS TAKEN PLACE')
      STOP
      END IF
   30 CONTINUE
      RETURN
      END
C
C     ------------------------------------------------------------------
C
C
      SUBROUTINE CQZHES(NM,N,AR,AI,BR,BI,MATZ,ZR,ZI)
C     ******************************************************************
C     THIS ALGORITHM  QZ  FOR THE CLASS OF COMPLEX GENERALIZED MATRIX
C     SYSTEMS EXHIBITING THE USE OF  QZ  TO FIND ALL THE EIGENVALUES
C     AND EIGENVECTORS FOR THE EIGENPROBLEM   A*X = (LAMBDA)*B*X .
C
C     THE DIMENSION OF  A,AI,B,BI,Z,  AND  ZI  SHOULD BE  NM  BY  NM.
C     THE DIMENSION OF  ALFR,ALFI,BETA,  AND  NORM  SHOULD BE  NM.
C     THE DIMENSION OF  AHOLD  AND  BHOLD  SHOULD BE  NM  BY  NM.
C     THE DIMENSION OF  AHOLDI  AND  BHOLDI  SHOULD BE  NM  BY  NM.
C     HERE NM = 20.
C
      INTEGER I,J,K,L,N,K1,LB,L1,NM,NK1,NM1
      REAL AR(NM,N),AI(NM,N),BR(NM,N),BI(NM,N),ZR(NM,N),ZI(NM,N)
      REAL R,S,T,TI,U1,U2,XI,XR,YI,YR,RHO,U1I
      REAL SQRT,CABS,ABS
      LOGICAL MATZ
      COMPLEX CMPLX
C
C     THIS SUBROUTINE IS A COMPLEX ANALOGUE OF THE FIRST STEP OF THE
C     QZ ALGORITHM FOR SOLVING GENERALIZED MATRIX EIGENVALUE PROBLEMS,
C     SIAM J. NUMER. ANAL. 10, 241-256(1973) BY MOLER AND STEWART.
C
C     THIS SUBROUTINE ACCEPTS A PAIR OF COMPLEX GENERAL MATRICES AND
C     REDUCES ONE OF THEM TO UPPER HESSENBERG FORM WITH REAL (AND NON-
C     NEGATIVE) SUBDIAGONAL ELEMENTS AND THE OTHER TO UPPER TRIANGULAR
C     FORM USING UNITARY TRANSFORMATIONS.  IT IS USUALLY FOLLOWED BY
C     CQZVAL  AND POSSIBLY  CQZVEC.
C
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRICES,
C
C        A=(AR,AI) CONTAINS A COMPLEX GENERAL MATRIX,
```

```
C
C        B=(BR,BI) CONTAINS A COMPLEX GENERAL MATRIX,
C
C        MATZ SHOULD BE SET TO .TRUE. IF THE RIGHT HAND TRANSFORMATIONS
C          ARE TO BE ACCUMULATED FOR LATER USE IN COMPUTING
C          EIGENVECTORS, AND TO .FALSE. OTHERWISE.
C
C     ON OUTPUT-
C
C        A HAS BEEN REDUCED TO UPPER HESSENBERG FORM.  THE ELEMENTS
C          BELOW THE FIRST SUBDIAGONAL HAVE BEEN SET TO ZERO, AND THE
C          SUBDIAGONAL ELEMENTS HAVE BEEN MADE REAL (AND NON-NEGATIVE),
C
C        B HAS BEEN REDUCED TO UPPER TRIANGULAR FORM.  THE ELEMENTS
C          BELOW THE MAIN DIAGONAL HAVE BEEN SET TO ZERO,
C
C        Z=(ZR,ZI) CONTAINS THE PRODUCT OF THE RIGHT HAND
C          TRANSFORMATIONS IF MATZ HAS BEEN SET TO .TRUE.
C          OTHERWISE, Z IS NOT REFERENCED.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
C     ********** INITIALIZE Z **********
      IF (.NOT. MATZ) GO TO 10
C
      DO 3 I = 1, N
C
         DO 2 J = 1, N
            ZR(I,J) = 0.0
            ZI(I,J) = 0.0
    2    CONTINUE
C
         ZR(I,I) = 1.0
    3 CONTINUE
C     ********** REDUCE B TO UPPER TRIANGULAR FORM WITH
C                TEMPORARILY REAL DIAGONAL ELEMENTS **********
   10 IF (N .LE. 1) GO TO 170
      NM1 = N - 1
C
      DO 100 L = 1, NM1
         L1 = L + 1
         S = 0.0
C
         DO 20 I = L, N
            S = S + ABS(BR(I,L)) + ABS(BI(I,L))
   20    CONTINUE
C
         IF (S .EQ. 0.0) GO TO 100
         RHO = 0.0
C
         DO 25 I = L, N
            BR(I,L) = BR(I,L) / S
```

```
         BI(I,L) = BI(I,L) / S
         RHO = RHO + BR(I,L)**2 + BI(I,L)**2
   25 CONTINUE
C
      R = SQRT(RHO)
      XR = CABS(CMPLX(BR(L,L),BI(L,L)))
      IF (XR .EQ. 0.0) GO TO 27
      RHO = RHO + XR * R
      U1 = -BR(L,L) / XR
      U1I = -BI(L,L) / XR
      YR = R / XR + 1.0
      BR(L,L) = YR * BR(L,L)
      BI(L,L) = YR * BI(L,L)
      GO TO 28
C
   27 BR(L,L) = R
      U1 = -1.0
      U1I = 0.0
C
   28 DO 50 J = L1, N
         T = 0.0
         TI = 0.0
C
         DO 30 I = L, N
            T = T + BR(I,L) * BR(I,J) + BI(I,L) * BI(I,J)
            TI = TI + BR(I,L) * BI(I,J) - BI(I,L) * BR(I,J)
   30    CONTINUE
C
         T = T / RHO
         TI = TI / RHO
C
         DO 40 I = L, N
            BR(I,J) = BR(I,J) - T * BR(I,L) + TI * BI(I,L)
            BI(I,J) = BI(I,J) - T * BI(I,L) - TI * BR(I,L)
   40    CONTINUE
C
         XI = U1 * BI(L,J) - U1I * BR(L,J)
         BR(L,J) = U1 * BR(L,J) + U1I * BI(L,J)
         BI(L,J) = XI
   50 CONTINUE
C
      DO 80 J = 1, N
         T = 0.0
         TI = 0.0
C
         DO 60 I = L, N
            T = T + BR(I,L) * AR(I,J) + BI(I,L) * AI(I,J)
            TI = TI + BR(I,L) * AI(I,J) - BI(I,L) * AR(I,J)
   60    CONTINUE
C
         T = T / RHO
         TI = TI / RHO
C
         DO 70 I = L, N
            AR(I,J) = AR(I,J) - T * BR(I,L) + TI * BI(I,L)
```

```
             AI(I,J) = AI(I,J) - T * BI(I,L) - TI * BR(I,L)
   70       CONTINUE
C
            XI = U1 * AI(L,J) - U1I * AR(L,J)
            AR(L,J) = U1 * AR(L,J) + U1I * AI(L,J)
            AI(L,J) = XI
   80    CONTINUE
C
         BR(L,L) = R * S
         BI(L,L) = 0.0
C
         DO 90 I = L1, N
            BR(I,L) = 0.0
            BI(I,L) = 0.0
   90    CONTINUE
C
  100 CONTINUE
C     ********** REDUCE A TO UPPER HESSENBERG FORM WITH REAL SUBDIAGONAL
C                ELEMENTS, WHILE KEEPING B TRIANGULAR **********
      DO 160 K = 1, NM1
         K1 = K + 1
C     ********** SET BOTTOM ELEMENT IN K-TH COLUMN OF A REAL **********
         IF (AI(N,K) .EQ. 0.0) GO TO 105
         R = CABS(CMPLX(AR(N,K),AI(N,K)))
         U1 = AR(N,K) / R
         U1I = AI(N,K) / R
         AR(N,K) = R
         AI(N,K) = 0.0
C
         DO 103 J = K1, N
            XI = U1 * AI(N,J) - U1I * AR(N,J)
            AR(N,J) = U1 * AR(N,J) + U1I * AI(N,J)
            AI(N,J) = XI
  103    CONTINUE
C
         XI = U1 * BI(N,N) - U1I * BR(N,N)
         BR(N,N) = U1 * BR(N,N) + U1I * BI(N,N)
         BI(N,N) = XI
  105    IF (K .EQ. NM1) GO TO 170
         NK1 = NM1 - K
C     ********** FOR L=N-1 STEP -1 UNTIL K+1 DO -- **********
         DO 150 LB = 1, NK1
            L = N - LB
            L1 = L + 1
C     ********** ZERO A(L+1,K) **********
            S = ABS(AR(L,K)) + ABS(AI(L,K)) + AR(L1,K)
            IF (S .EQ. 0.0) GO TO 150
            U1 = AR(L,K) / S
            U1I = AI(L,K) / S
            U2 = AR(L1,K) / S
            R = SQRT(U1*U1+U1I*U1I+U2*U2)
            U1 = U1 / R
            U1I = U1I / R
            U2 = U2 / R
            AR(L,K) = R * S
```

```
      AI(L,K) = 0.0
      AR(L1,K) = 0.0
C
      DO 110 J = K1, N
         XR = AR(L,J)
         XI = AI(L,J)
         YR = AR(L1,J)
         YI = AI(L1,J)
         AR(L,J) = U1 * XR + U1I * XI + U2 * YR
         AI(L,J) = U1 * XI - U1I * XR + U2 * YI
         AR(L1,J) = U1 * YR - U1I * YI - U2 * XR
         AI(L1,J) = U1 * YI + U1I * YR - U2 * XI
  110 CONTINUE
C
      XR = BR(L,L)
      BR(L,L) = U1 * XR
      BI(L,L) = -U1I * XR
      BR(L1,L) = -U2 * XR
C
      DO 120 J = L1, N
         XR = BR(L,J)
         XI = BI(L,J)
         YR = BR(L1,J)
         YI = BI(L1,J)
         BR(L,J) = U1 * XR + U1I * XI + U2 * YR
         BI(L,J) = U1 * XI - U1I * XR + U2 * YI
         BR(L1,J) = U1 * YR - U1I * YI - U2 * XR
         BI(L1,J) = U1 * YI + U1I * YR - U2 * XI
  120 CONTINUE
C     ********** ZERO B(L+1,L) **********
      S = ABS(BR(L1,L1)) + ABS(BI(L1,L1)) + ABS(BR(L1,L))
      IF (S .EQ. 0.0) GO TO 150
      U1 = BR(L1,L1) / S
      U1I = BI(L1,L1) / S
      U2 = BR(L1,L) / S
      R = SQRT(U1*U1+U1I*U1I+U2*U2)
      U1 = U1 / R
      U1I = U1I / R
      U2 = U2 / R
      BR(L1,L1) = R * S
      BI(L1,L1) = 0.0
      BR(L1,L) = 0.0
C
      DO 130 I = 1, L
         XR = BR(I,L1)
         XI = BI(I,L1)
         YR = BR(I,L)
         YI = BI(I,L)
         BR(I,L1) = U1 * XR + U1I * XI + U2 * YR
         BI(I,L1) = U1 * XI - U1I * XR + U2 * YI
         BR(I,L) = U1 * YR - U1I * YI - U2 * XR
         BI(I,L) = U1 * YI + U1I * YR - U2 * XI
  130 CONTINUE
C
      DO 140 I = 1, N
```

```
            XR = AR(I,L1)
            XI = AI(I,L1)
            YR = AR(I,L)
            YI = AI(I,L)
            AR(I,L1) = U1 * XR + U1I * XI + U2 * YR
            AI(I,L1) = U1 * XI - U1I * XR + U2 * YI
            AR(I,L) = U1 * YR - U1I * YI - U2 * XR
            AI(I,L) = U1 * YI + U1I * YR - U2 * XI
  140     CONTINUE
C
          IF (.NOT. MATZ) GO TO 150
C
          DO 145 I = 1, N
            XR = ZR(I,L1)
            XI = ZI(I,L1)
            YR = ZR(I,L)
            YI = ZI(I,L)
            ZR(I,L1) = U1 * XR + U1I * XI + U2 * YR
            ZI(I,L1) = U1 * XI - U1I * XR + U2 * YI
            ZR(I,L) = U1 * YR - U1I * YI - U2 * XR
            ZI(I,L) = U1 * YI + U1I * YR - U2 * XI
  145     CONTINUE
C
  150   CONTINUE
C
  160 CONTINUE
C
  170 RETURN
C     ********** LAST CARD OF CQZHES **********
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE CQZVAL(NM,N,AR,AI,BR,BI,EPS1,ALFR,ALFI,BETA,
     X                                        MATZ,ZR,ZI,IERR)
C     *****************************************************************
C
      INTEGER I,J,K,L,N,EN,K1,K2,LL,L1,NA,NM,ITS,KM1,LM1,
     X        ENM2,IERR,LOR1,ENORN
      REAL AR(NM,N),AI(NM,N),BR(NM,N),BI(NM,N),ALFR(N),ALFI(N),
     X       BETA(N),ZR(NM,N),ZI(NM,N)
      REAL R,S,A1,A2,EP,SH,U1,U2,XI,XR,YI,YR,ANI,A1I,A33,A34,A43,A44,
     X       BNI,B11,B33,B44,SHI,U1I,A33I,A34I,A43I,A44I,B33I,B44I,
     X       EPSA,EPSB,EPS1,ANORM,BNORM,B3344,B3344I
      REAL SQRT,CABS,ABS
      INTEGER MAX0
      LOGICAL MATZ
      COMPLEX Z3
      COMPLEX CSQRT,CMPLX
      REAL REAL,AIMAG
C
C
C
C
C
```

```
C     THIS SUBROUTINE IS A COMPLEX ANALOGUE OF STEPS 2 AND 3 OF THE
C     QZ ALGORITHM FOR SOLVING GENERALIZED MATRIX EIGENVALUE PROBLEMS,
C     SIAM J. NUMER. ANAL. 10, 241-256(1973) BY MOLER AND STEWART,
C     AS MODIFIED IN TECHNICAL NOTE NASA TN E-7305(1973) BY WARD.
C
C     THIS SUBROUTINE ACCEPTS A PAIR OF COMPLEX MATRICES, ONE OF THEM
C     IN UPPER HESSENBERG FORM AND THE OTHER IN UPPER TRIANGULAR FORM,
C     THE HESSENBERG MATRIX MUST FURTHER HAVE REAL SUBDIAGONAL ELEMENTS.
C     IT REDUCES THE HESSENBERG MATRIX TO TRIANGULAR FORM USING
C     UNITARY TRANSFORMATIONS WHILE MAINTAINING THE TRIANGULAR FORM
C     OF THE OTHER MATRIX AND FURTHER MAKING ITS DIAGONAL ELEMENTS
C     REAL AND NON-NEGATIVE.  IT THEN RETURNS QUANTITIES WHOSE RATIOS
C     GIVE THE GENERALIZED EIGENVALUES.  IT IS USUALLY PRECEDED BY
C     CQZHES  AND POSSIBLY FOLLOWED BY  CQZVEC.
C
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRICES,
C
C        A=(AR,AI) CONTAINS A COMPLEX UPPER HESSENBERG MATRIX
C          WITH REAL SUBDIAGONAL ELEMENTS,
C
C        B=(BR,BI) CONTAINS A COMPLEX UPPER TRIANGULAR MATRIX,
C
C        EPS1 IS A TOLERANCE USED TO DETERMINE NEGLIGIBLE ELEMENTS.
C          EPS1 = 0.0 (OR NEGATIVE) MAY BE INPUT, IN WHICH CASE AN
C          ELEMENT WILL BE NEGLECTED ONLY IF IT IS LESS THAN ROUNDOFF
C          ERROR TIMES THE NORM OF ITS MATRIX.  IF THE INPUT EPS1 IS
C          POSITIVE, THEN AN ELEMENT WILL BE CONSIDERED NEGLIGIBLE
C          IF IT IS LESS THAN EPS1 TIMES THE NORM OF ITS MATRIX.  A
C          POSITIVE VALUE OF EPS1 MAY RESULT IN FASTER EXECUTION,
C          BUT LESS ACCURATE RESULTS,
C
C        MATZ SHOULD BE SET TO .TRUE. IF THE RIGHT HAND TRANSFORMATIONS
C          ARE TO BE ACCUMULATED FOR LATER USE IN COMPUTING
C          EIGENVECTORS, AND TO .FALSE. OTHERWISE,
C
C        Z=(ZR,ZI) CONTAINS, IF MATZ HAS BEEN SET TO .TRUE., THE
C          TRANSFORMATION MATRIX PRODUCED IN THE REDUCTION
C          BY  CQZHES, IF PERFORMED, OR ELSE THE IDENTITY MATRIX.
C          IF MATZ HAS BEEN SET TO .FALSE., Z IS NOT REFERENCED.
C
C     ON OUTPUT
C
C        A HAS BEEN REDUCED TO UPPER TRIANGULAR FORM.  THE ELEMENTS
C          BELOW THE MAIN DIAGONAL HAVE BEEN SET TO ZERO,
C
C        B IS STILL IN UPPER TRIANGULAR FORM, ALTHOUGH ITS ELEMENTS
C          HAVE BEEN ALTERED.  IN PARTICULAR, ITS DIAGONAL HAS BEEN SET
C          REAL AND NON-NEGATIVE.  THE LOCATION BR(N,1) IS USED TO
C          STORE EPS1 TIMES THE NORM OF B FOR LATER USE BY  CQZVEC,
```

```
C
C       ALFR AND ALFI CONTAIN THE REAL AND IMAGINARY PARTS OF THE
C         DIAGONAL ELEMENTS OF THE TRIANGULARIZED A MATRIX,
C
C       BETA CONTAINS THE REAL NON-NEGATIVE DIAGONAL ELEMENTS OF THE
C         CORRESPONDING B.  THE GENERALIZED EIGENVALUES ARE THEN
C         THE RATIOS ((ALFR+I*ALFI)/BETA),
C
C       Z CONTAINS THE PRODUCT OF THE RIGHT HAND TRANSFORMATIONS
C         (FOR BOTH STEPS) IF MATZ HAS BEEN SET TO .TRUE.,
C
C       IERR IS SET TO
C         ZERO       FOR NORMAL RETURN,
C         J          IF AR(J,J-1) HAS NOT BECOME
C                    ZERO AFTER 50 ITERATIONS.
C
C    QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C    APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C    ------------------------------------------------------------------
C
      IERR = 0
C     ********** COMPUTE EPSA,EPSB **********
      ANORM = 0.0
      BNORM = 0.0
C
      DO 30 I = 1, N
         ANI = 0.0
         IF (I .NE. 1) ANI = ABS(AR(I,I-1))
         BNI = 0.0
C
         DO 20 J = I, N
            ANI = ANI + ABS(AR(I,J)) + ABS(AI(I,J))
            BNI = BNI + ABS(BR(I,J)) + ABS(BI(I,J))
   20    CONTINUE
C
         IF (ANI .GT. ANORM) ANORM = ANI
         IF (BNI .GT. BNORM) BNORM = BNI
   30 CONTINUE
C
      IF (ANORM .EQ. 0.0) ANORM = 1.0
      IF (BNORM .EQ. 0.0) BNORM = 1.0
      EP = EPS1
      IF (EP .GT. 0.0) GO TO 50
C     ********** COMPUTE ROUNDOFF LEVEL IF EPS1 IS ZERO **********
      EP = 1.0
   40 EP = EP / 2.0
      IF (1.0 + EP .GT. 1.0) GO TO 40
   50 EPSA = EP * ANORM
      EPSB = EP * BNORM
C     ********** REDUCE A TO TRIANGULAR FORM, WHILE
C                KEEPING B TRIANGULAR **********
      LOR1 = 1
      ENORN = N
      EN = N
```

```
C     ********** BEGIN QZ STEP **********
   60 IF (EN .EQ. 0) GO TO 1001
      IF (.NOT. MATZ) ENORN = EN
      ITS = 0
      NA = EN - 1
      ENM2 = NA - 1
C     ********** CHECK FOR CONVERGENCE OR REDUCIBILITY.
C                FOR L=EN STEP -1 UNTIL 1 DO -- **********
   70 DO 80 LL = 1, EN
         LM1 = EN - LL
         L = LM1 + 1
         IF (L .EQ. 1) GO TO 95
         IF (ABS(AR(L,LM1)) .LE. EPSA) GO TO 90
   80 CONTINUE
C
   90 AR(L,LM1) = 0.0
C     ********** SET DIAGONAL ELEMENT AT TOP OF B REAL **********
   95 B11 = CABS(CMPLX(BR(L,L),BI(L,L)))
      IF (B11     .EQ. 0.0) GO TO 98
      U1 = BR(L,L) / B11
      U1I = BI(L,L) / B11
C
      DO 97 J = L, ENORN
         XI = U1 * AI(L,J) - U1I * AR(L,J)
         AR(L,J) = U1 * AR(L,J) + U1I * AI(L,J)
         AI(L,J) = XI
         XI = U1 * BI(L,J) - U1I * BR(L,J)
         BR(L,J) = U1 * BR(L,J) + U1I * BI(L,J)
         BI(L,J) = XI
   97 CONTINUE
C
      BI(L,L) = 0.0
   98 IF (L .NE. EN) GO TO 100
C     ********** 1-BY-1 BLOCK ISOLATED **********
      ALFR(EN) = AR(EN,EN)
      ALFI(EN) = AI(EN,EN)
      BETA(EN) = B11
      EN = NA
      GO TO 60
C     ********** CHECK FOR SMALL TOP OF B **********
  100 L1 = L + 1
      IF (B11 .GT. EPSB) GO TO 120
      BR(L,L) = 0.0
      S = ABS(AR(L,L)) + ABS(AI(L,L)) + ABS(AR(L1,L))
      U1 = AR(L,L) / S
      U1I = AI(L,L) / S
      U2 = AR(L1,L) / S
      R = SQRT(U1*U1+U1I*U1I+U2*U2)
      U1 = U1 / R
      U1I = U1I / R
      U2 = U2 / R
      AR(L,L) = R * S
      AI(L,L) = 0.0
C
      DO 110 J = L1, ENORN
```

```
         XR = AR(L,J)
         XI = AI(L,J)
         YR = AR(L1,J)
         YI = AI(L1,J)
         AR(L,J) = U1 * XR + U1I * XI + U2 * YR
         AI(L,J) = U1 * XI - U1I * XR + U2 * YI
         AR(L1,J) = U1 * YR - U1I * YI - U2 * XR
         AI(L1,J) = U1 * YI + U1I * YR - U2 * XI
         XR = BR(L,J)
         XI = BI(L,J)
         YR = BR(L1,J)
         YI = BI(L1,J)
         BR(L1,J) = U1 * YR - U1I * YI - U2 * XR
         BR(L,J) = U1 * XR + U1I * XI + U2 * YR
         BI(L,J) = U1 * XI - U1I * XR + U2 * YI
         BI(L1,J) = U1 * YI + U1I * YR - U2 * XI
  110 CONTINUE
C
      LM1 = L
      L = L1
      GO TO 90
C     ********** ITERATION STRATEGY **********
  120 IF (ITS .EQ. 50) GO TO 1000
      IF (ITS .EQ. 10) GO TO 135
C     ********** DETERMINE SHIFT **********
      B33 = BR(NA,NA)
      B33I = BI(NA,NA)
      IF (CABS(CMPLX(B33,B33I)) .GE. EPSB) GO TO 122
      B33 = EPSB
      B33I = 0.0
  122 B44 = BR(EN,EN)
      B44I = BI(EN,EN)
      IF (CABS(CMPLX(B44,B44I)) .GE. EPSB) GO TO 124
      B44 = EPSB
      B44I = 0.0
  124 B3344 = B33 * B44 - B33I * B44I
      B3344I = B33 * B44I + B33I * B44
      A33 = AR(NA,NA) * B44 - AI(NA,NA) * B44I
      A33I = AR(NA,NA) * B44I + AI(NA,NA) * B44
      A34 = AR(NA,EN) * B33 - AI(NA,EN) * B33I
     X    - AR(NA,NA) * BR(NA,EN) + AI(NA,NA) * BI(NA,EN)
      A34I = AR(NA,EN) * B33I + AI(NA,EN) * B33
     X     - AR(NA,NA) * BI(NA,EN) - AI(NA,NA) * BR(NA,EN)
      A43 = AR(EN,NA) * B44
      A43I = AR(EN,NA) * B44I
      A44 = AR(EN,EN) * B33 - AI(EN,EN) * B33I - AR(EN,NA) * BR(NA,EN)
      A44I = AR(EN,EN) * B33I + AI(EN,EN) * B33 - AR(EN,NA) * BI(NA,EN)
      SH = A44
      SHI = A44I
      XR = A34 * A43 - A34I * A43I
      XI = A34 * A43I + A34I * A43
      IF (XR .EQ. 0.0 .AND. XI .EQ. 0.0) GO TO 140
      YR = (A33 - SH) / 2.0
      YI = (A33I - SHI) / 2.0
      Z3 = CSQRT(CMPLX(YR**2-YI**2+XR,2.0*YR*YI+XI))
```

```
      U1 = REAL(Z3)
      U1I = AIMAG(Z3)
      IF (YR * U1 + YI * U1I .GE. 0.0) GO TO 125
      U1 = -U1
      U1I = -U1I
  125 Z3 = (CMPLX(SH,SHI) - CMPLX(XR,XI) / CMPLX(YR+U1,YI+U1I))
     X   / CMPLX(B3344,B3344I)
      SH = REAL(Z3)
      SHI = AIMAG(Z3)
      GO TO 140
C     ********** AD HOC SHIFT **********
  135 SH = AR(EN,NA) + AR(NA,ENM2)
      SHI = 0.0
C     ********** DETERMINE ZEROTH COLUMN OF A **********
  140 A1 = AR(L,L) / B11 - SH
      A1I = AI(L,L) / B11 - SHI
      A2 = AR(L1,L) / B11
      ITS = ITS + 1
      IF (.NOT. MATZ) LOR1 = L
C     ********** MAIN LOOP **********
      DO 260 K = L, NA
         K1 = K + 1
         K2 = K + 2
         KM1 = MAX0(K-1,L)
C     ********** ZERO A(K+1,K-1) **********
         IF (K .EQ. L) GO TO 170
         A1 = AR(K,KM1)
         A1I = AI(K,KM1)
         A2 = AR(K1,KM1)
  170    S = ABS(A1) + ABS(A1I) + ABS(A2)
         U1 = A1 / S
         U1I = A1I / S
         U2 = A2 / S
         R = SQRT(U1*U1+U1I*U1I+U2*U2)
         U1 = U1 / R
         U1I = U1I / R
         U2 = U2 / R
C
         DO 180 J = KM1, ENORN
            XR = AR(K,J)
            XI = AI(K,J)
            YR = AR(K1,J)
            YI = AI(K1,J)
            AR(K,J) = U1 * XR + U1I * XI + U2 * YR
            AI(K,J) = U1 * XI - U1I * XR + U2 * YI
            AR(K1,J) = U1 * YR - U1I * YI - U2 * XR
            AI(K1,J) = U1 * YI + U1I * YR - U2 * XI
            XR = BR(K,J)
            XI = BI(K,J)
            YR = BR(K1,J)
            YI = BI(K1,J)
            BR(K,J) = U1 * XR + U1I * XI + U2 * YR
            BI(K,J) = U1 * XI - U1I * XR + U2 * YI
            BR(K1,J) = U1 * YR - U1I * YI - U2 * XR
            BI(K1,J) = U1 * YI + U1I * YR - U2 * XI
```

```
  180   CONTINUE
C .
        IF (K .EQ. L) GO TO 240
        AI(K,KM1) = 0.0
        AR(K1,KM1) = 0.0
        AI(K1,KM1) = 0.0
C     ********** ZERO B(K+1,K) **********
  240   S = ABS(BR(K1,K1)) + ABS(BI(K1,K1)) + ABS(BR(K1,K))
        U1 = BR(K1,K1) / S
        U1I = BI(K1,K1) / S
        U2 = BR(K1,K) / S
        R = SQRT(U1*U1+U1I*U1I+U2*U2)
        U1 = U1 / R
        U1I = U1I / R
        U2 = U2 / R
        IF (K .EQ. NA) GO TO 245
        XR = AR(K2,K1)
        AR(K2,K1) = U1 * XR
        AI(K2,K1) = -U1I * XR
        AR(K2,K) = -U2 * XR
C
  245   DO 250 I = LOR1, K1
           XR = AR(I,K1)
           XI = AI(I,K1)
           YR = AR(I,K)
           YI = AI(I,K)
           AR(I,K1) = U1 * XR + U1I * XI + U2 * YR
           AI(I,K1) = U1 * XI - U1I * XR + U2 * YI
           AR(I,K) = U1 * YR - U1I * YI - U2 * XR
           AI(I,K) = U1 * YI + U1I * YR - U2 * XI
           XR = BR(I,K1)
           XI = BI(I,K1)
           YR = BR(I,K)
           YI = BI(I,K)
           BR(I,K1) = U1 * XR + U1I * XI + U2 * YR
           BI(I,K1) = U1 * XI - U1I * XR + U2 * YI
           BR(I,K) = U1 * YR - U1I * YI - U2 * XR
           BI(I,K) = U1 * YI + U1I * YR - U2 * XI
  250   CONTINUE
C
        BI(K1,K1) = 0.0
        BR(K1,K) = 0.0
        BI(K1,K) = 0.0
        IF (.NOT. MATZ) GO TO 260
C
        DO 255 I = 1, N
           XR = ZR(I,K1)
           XI = ZI(I,K1)
           YR = ZR(I,K)
           YI = ZI(I,K)
           ZR(I,K1) = U1 * XR + U1I * XI + U2 * YR
           ZI(I,K1) = U1 * XI - U1I * XR + U2 * YI
           ZR(I,K) = U1 * YR - U1I * YI - U2 * XR
           ZI(I,K) = U1 * YI + U1I * YR - U2 * XI
  255   CONTINUE
```

```
C
  260 CONTINUE
C     ********** SET LAST A SUBDIAGONAL REAL AND END QZ STEP **********
      IF (AI(EN,NA) .EQ. 0.0) GO TO 70
      R = CABS(CMPLX(AR(EN,NA),AI(EN,NA)))
      U1 = AR(EN,NA) / R
      U1I = AI(EN,NA) / R
      AR(EN,NA) = R
      AI(EN,NA) = 0.0
C
      DO 270 J = EN, ENORN
         XI = U1 * AI(EN,J) - U1I * AR(EN,J)
         AR(EN,J) = U1 * AR(EN,J) + U1I * AI(EN,J)
         AI(EN,J) = XI
         XI = U1 * BI(EN,J) - U1I * BR(EN,J)
         BR(EN,J) = U1 * BR(EN,J) + U1I * BI(EN,J)
         BI(EN,J) = XI
  270 CONTINUE
C
      GO TO 70
C     ********** SET ERROR -- BOTTOM SUBDIAGONAL ELEMENT HAS NOT
C                BECOME NEGLIGIBLE AFTER 50 ITERATIONS **********
 1000 IERR = EN
C     ********** SAVE EPSB FOR USE BY CQZVEC **********
 1001 IF (N .GT. 1) BR(N,1) = EPSB
      RETURN
C     ********** LAST CARD OF CQZVAL **********
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE CQZVEC(NM,N,AR,AI,BR,BI,ALFR,ALFI,BETA,ZR,ZI)
C     ******************************************************************
C
      INTEGER I,J,K,M,N,EN,II,JJ,NA,NM,NN
      REAL AR(NM,N),AI(NM,N),BR(NM,N),BI(NM,N),ALFR(N),ALFI(N),
     X       BETA(N),ZR(NM,N),ZI(NM,N)
      REAL R,T,RI,TI,XI,ALMI,ALMR,BETM,EPSB
      REAL CABS
      COMPLEX Z3
      COMPLEX CMPLX
      REAL REAL,AIMAG
C
C
C
C
C
C     THIS SUBROUTINE IS A COMPLEX ANALOGUE OF THE FOURTH STEP OF THE
C     QZ ALGORITHM FOR SOLVING GENERALIZED MATRIX EIGENVALUE PROBLEMS,
C     SIAM J. NUMER. ANAL. 10, 241-256(1973) BY MOLER AND STEWART.
C
C     THIS SUBROUTINE ACCEPTS A PAIR OF COMPLEX MATRICES IN UPPER
C     TRIANGULAR FORM, WHERE ONE OF THEM FURTHER MUST HAVE REAL DIAGONAL
C     ELEMENTS.  IT COMPUTES THE EIGENVECTORS OF THE TRIANGULAR PROBLEM
C     AND TRANSFORMS THE RESULTS BACK TO THE ORIGINAL COORDINATE SYSTEM.
```

```
C     IT IS USUALLY PRECEDED BY  CQZHES  AND  CQZVAL.
C
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRICES,
C
C        A=(AR,AI) CONTAINS A COMPLEX UPPER TRIANGULAR MATRIX,
C
C        B=(BR,BI) CONTAINS A COMPLEX UPPER TRIANGULAR MATRIX WITH REAL
C          DIAGONAL ELEMENTS.  IN ADDITION, LOCATION BR(N,1) CONTAINS
C          THE TOLERANCE QUANTITY (EPSB) COMPUTED AND SAVED IN  CQZVAL,
C
C        ALFR, ALFI, AND BETA ARE VECTORS WITH COMPONENTS WHOSE
C          RATIOS ((ALFR+I*ALFI)/BETA) ARE THE GENERALIZED
C          EIGENVALUES.  THEY ARE USUALLY OBTAINED FROM  CQZVAL,
C
C        Z=(ZR,ZI) CONTAINS THE TRANSFORMATION MATRIX PRODUCED IN THE
C          REDUCTIONS BY  CQZHES  AND  CQZVAL, IF PERFORMED.
C          IF THE EIGENVECTORS OF THE TRIANGULAR PROBLEM ARE
C          DESIRED, Z MUST CONTAIN THE IDENTITY MATRIX.
C
C     ON OUTPUT-
C
C        A IS UNALTERED,
C
C        B HAS BEEN DESTROYED,
C
C        ALFR, ALFI, AND BETA ARE UNALTERED,
C
C        Z CONTAINS THE EIGENVECTORS.  EACH EIGENVECTOR IS NORMALIZED
C          SO THAT THE MODULUS OF ITS LARGEST COMPONENT IS 1.0 .
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
      IF (N .LE. 1) GO TO 1001
      EPSB = BR(N,1)
C     ********** FOR EN=N STEP -1 UNTIL 2 DO -- **********
      DO 800 NN = 2, N
         EN = N + 2 - NN
         NA = EN - 1
         ALMR = ALFR(EN)
         ALMI = ALFI(EN)
         BETM = BETA(EN)
C     ********** FOR I=EN-1 STEP -1 UNTIL 1 DO -- **********
         DO 700 II = 1, NA
            I = EN - II
            R = 0.0
            RI = 0.0
```

```
            M = I + 1
C
            DO 610 J = M, EN
               T = BETM * AR(I,J) - ALMR * BR(I,J) + ALMI * BI(I,J)
               TI = BETM * AI(I,J) - ALMR * BI(I,J) - ALMI * BR(I,J)
               IF (J .EQ. EN) GO TO 605
               XI = T * BI(J,EN) + TI * BR(J,EN)
               T = T * BR(J,EN) - TI * BI(J,EN)
               TI = XI
  605          R = R + T
               RI = RI + TI
  610       CONTINUE
C
            T = ALMR * BETA(I) - BETM * ALFR(I)
            TI = ALMI * BETA(I) - BETM * ALFI(I)
            IF (T .EQ. 0.0 .AND. TI .EQ. 0.0) T = EPSB
            Z3 = CMPLX(R,RI) / CMPLX(T,TI)
            BR(I,EN) = REAL(Z3)
            BI(I,EN) = AIMAG(Z3)
  700    CONTINUE
C
  800 CONTINUE
C     ********** END BACK SUBSTITUTION.
C                TRANSFORM TO ORIGINAL COORDINATE SYSTEM.
C                FOR J=N STEP -1 UNTIL 2 DO -- **********
      DO 880 JJ = 2, N
         J = N + 2 - JJ
         M = J - 1
C
         DO 880 I = 1, N
C
            DO 860 K = 1, M
               ZR(I,J) = ZR(I,J) + ZR(I,K) * BR(K,J) - ZI(I,K) * BI(K,J)
               ZI(I,J) = ZI(I,J) + ZR(I,K) * BI(K,J) + ZI(I,K) * BR(K,J)
  860       CONTINUE
C
  880 CONTINUE
C     ********** NORMALIZE SO THAT MODULUS OF LARGEST
C                COMPONENT OF EACH VECTOR IS 1 **********
      DO 950 J = 1, N
         T = 0.0
C
         DO 930 I = 1, N
            R = CABS(CMPLX(ZR(I,J),ZI(I,J)))
            IF (R .GT. T) T = R
  930    CONTINUE
C
         DO 940 I = 1, N
            ZR(I,J) = ZR(I,J) / T
            ZI(I,J) = ZI(I,J) / T
  940    CONTINUE
C
  950 CONTINUE
C
 1001 RETURN
```

```
C     ********** LAST CARD OF CQZVEC **********
      END
C     ---------------------------------------------------------------
C

C
      SUBROUTINE NORMA(FI,G,N,NM,XM)
C     *****************************************************************
C
      COMPLEX FI(NM,NM),XM(NM,NM),TEMP(20),SUM1,SUM2
      COMPLEX XI(20,20),X(20,20),X1,X2
      REAL G(NM,NM)
      DATA IWRITE/4/
C
      DO 30 I=1,N
      DO 40 J=1,N
      SUM1=(0.0,0.0)
      DO 50 K=1,N
      SUM1=SUM1+CMPLX(G(J,K))*FI(K,I)
   50 CONTINUE
      TEMP(J)=SUM1
   40 CONTINUE
      SUM2=(0.0,0.0)
      DO 41 J=1,N
      SUM2=SUM2+FI(J,I)*TEMP(J)
   41 CONTINUE
      SUM2=CSQRT(SUM2)
      DO 51 J=1,N
      FI(J,I)=FI(J,I)/SUM2
   51 CONTINUE
   30 CONTINUE
C
C FOR CHECK CARRY OUT [FIT]*[G]*[FI]=[I]
C
      DO 100 I=1,N
      DO 110 J=1,N
      X1=(0.0,0.0)
      DO 120 K=1,N
      X1=X1+FI(K,I)*CMPLX(G(K,J))
  120 CONTINUE
      X(I,J)=X1
  110 CONTINUE
  100 CONTINUE
      DO 130 I=1,N
      DO 140 J=1,N
      X2=(0.0,0.0)
      DO 150 K=1,N
      X2=X2+X(I,K)*FI(K,J)
  150 CONTINUE
      XI(I,J)=X2
  140 CONTINUE
  130 CONTINUE
      DO 161 I=1,N
      WRITE(IWRITE,*) (XI(I,J),J=1,N)
  161 CONTINUE
```

```
      RETURN
      END
C
C     -------------------------------------------------------------------
C
      SUBROUTINE SET(DY,T1,T2,N,NM,H,BO,G,BI)
C     ****************************************************
C
C SET CALCULATES MATRIX [A]p=[H]+i[BO]
C AND MATRIX [B]p=[G]
      REAL H(NM,NM),BO(NM,NM),G(NM,NM)
      REAL BI(NM,NM)
      DO 10 I=1,N
      I1=I-1
      I2=I+1
      DO 20 J=1,N
      IF(I.EQ.1.AND.J.EQ.1) THEN
      H(I,J)=1.0
      G(I,J)=1.0
      ELSE IF (I.NE.1.AND.I.EQ.J) THEN
      H(I,J)=2.0
      G(I,J)=2.0
      ELSE IF (J.EQ.I1.OR.J.EQ.I2) THEN
      H(I,J)=-1.0
      G(I,J)=0.5
      ELSE
      H(I,J)=0.0
      G(I,J)=0.0
      END IF
      H(I,J)=H(I,J)/DY
      G(I,J)=G(I,J)*(DY/3)
   20 CONTINUE
   10 CONTINUE
      DO 30 I=1,N
      DO 40 J=1,N
      IF(I.EQ.1.AND.J.EQ.1) THEN
      BO(I,J)=T2*T1
      ELSE
      BO(I,J)=0.0
      END IF
   40 CONTINUE
   30 CONTINUE
      DO 80 I=1,N
      DO 90 J=1,N
      BI(I,J)=0.0
   90 CONTINUE
   80 CONTINUE
      RETURN
      END
C
C     -------------------------------------------------------------------
C
      SUBROUTINE SETR(DY,ME,ND,A,F)
C     *******************************************************************
C
```

```
      REAL A(ND,ND),F(ND,ND)
      DO 9 I=1,ME
      DO 8 J=1,ME
      A(I,J)=0.
      F(I,J)=0.
    8 CONTINUE
    9 CONTINUE
      A(1,1)=1/DY
      A(1,2)=-1./DY
      F(1,1)=DY/3
      F(1,2)=.5*DY/3
      DO 10 I=2,ME-1
      A(I,I)=2/DY
      F(I,I)=2*DY/3
      A(I,I-1)=-1/DY
      A(I,I+1)=-1/DY
      F(I,I-1)=.5*DY/3
      F(I,I+1)=.5*DY/3
   10 CONTINUE
      A(ME,ME)=2/DY
      A(ME,ME-1)=-1/DY
      F(ME,ME)=2*DY/3
      F(ME,ME-1)=.5*DY/3
      RETURN
      END
C
C     -------------------------------------------------------------------
C
      SUBROUTINE FMAT(X,Y,XM,YM,G,H,HOLD,FID,FIB,DFI,N0,N1)
C     *******************************************************************
C
C     FMAT COMPUTES THE G AND H MATRICES AND FORMS
C     THE SYSTEMS OF EQUATIONS AX=F
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      DOUBLE PRECISION X(1),Y(1),G(NU,NU),H(NU,NU),
     1 FID(1),FIB(1),DFI(1),HOLD(NU,NU),XM(1),YM(1)
C
      NX=2*N
      DO 10 I=1,NX
      DFI(I)=0
      DO 11 J=1,NX
      G(I,J)=0.0
      H(I,J)=0.0
   11 CONTINUE
   10 CONTINUE
C
      X(N+1)=X(1)
      Y(N+1)=Y(1)
C
      DO 12 I=1,N
      XM(I)=(X(I)+X(I+1))/2
      YM(I)=(Y(I)+Y(I+1))/2
```

```
   12 CONTINUE
C
      DO 20 I=1,N
      DO 30 J=1,N
      IF(I-J)40,45,40
   40 CALL INTE(XM(I),YM(I),X(J),Y(J),X(J+1),Y(J+1),A,B)
      H(I,J)=A
      G(I,J)=B
      GO TO 30
   45 CALL INLO(X(J),Y(J),X(J+1),Y(J+1),B)
      G(I,J)=B
      H(I,J)=-0.5
   30 CONTINUE
   20 CONTINUE
C
C ARRANGE THE SYSTEM OF EQUATIONS
C
      DO 150 J=1,NO
      DO 145 I=1,N
      CH=G(I,J)
      G(I,J)=-H(I,J)
      H(I,J)=-CH
  145 CONTINUE
  150 CONTINUE
C
      DO 160 I=1,N
      DO 170 J=1,N
      HOLD(I,J)=H(I,J)
  170 CONTINUE
  160 CONTINUE
C
      RETURN
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE INTE(XP,YP,X1,Y1,X2,Y2,H,G)
C     ******************************************************************
C
C     THIS SUBROUTINE COMPUTES THE VALUES OF THE H AND GMATRIX
C     OFF DIAGONAL ELEMENTS BY MEANS OF NUMERICAL INTEGRATION
C     ALONG AN ELEMENT
C     USING 4-POINT GAUSS QUADRATURE
C     RA=DISTANCE FROM THE POINT UNDER CONSIDERATION
C     TO THE INTEGRATION POINT ON THE BOUNDARY ELEMENT
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      DOUBLE PRECISION XCO(4),YCO(4),GI(4),OME(4),
     &LEN,K
      K=CONST
      GI(1)=0.86113631
      GI(2)=-GI(1)
      GI(3)=0.33998104
```

```
      GI(4)=-GI(3)
      OME(1)=0.34785485
      OME(2)=OME(1)
      OME(3)=0.65214515
      OME(4)=OME(3)
      AX=(X2-X1)/2
      BX=(X2+X1)/2
      AY=(Y2-Y1)/2
      BY=(Y2+Y1)/2
      IF(AX.EQ.0)THEN
        DIST=DABS(XP-X1)
        ELSE
        TA=AY/AX
        DIST=DABS((TA*XP-YP+Y1-TA*X1)/DSQRT(TA**2+1))
        END IF
      SIG=(X1-XP)*(Y2-YP)-(X2-XP)*(Y1-YP)
      IF(SIG.LT.0) DIST=-DIST
      H=0
      G=0
      LEN=DSQRT(AX*AX+AY*AY)
      DO 40 I=1,4
      XCO(I)=AX*GI(I)+BX
      YCO(I)=AY*GI(I)+BY
      RA=DSQRT((XP-XCO(I))**2+(YP-YCO(I))**2)
      CON=K*RA
      CALL BFUNCT(CON,Y0X,Y1X)
      G=G+Y0X*OME(I)*LEN*0.25
   40 H=H+Y1X*OME(I)*DIST*K*(0.25/RA)*LEN
      RETURN
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE INLO(X1,Y1,X2,Y2,G)
C     ******************************************************************
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/MAIN/CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
C
C
C     THIS SUBROUTINE COMPUTES THE VALUES OF THE DIAGONAL
C     ELEMENTS OF THE  G  MATRIX
C
      AX=(X2-X1)/2
      AY=(Y2-Y1)/2
      SR=DSQRT(AX**2+AY**2)
      XXX=SR*CONST
      G=(SR/3.1415927)*(DLOG(XXX)-1.0)
      RETURN
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE BFUNCT(XX,Y0X,Y1X)
```

```
C     ****************************************************************
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/MAIN/CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      REAL*8 XULTO,XOSHO,XOSH1
      INTEGER XF,Z0,Z
C     BESSEL FUNCTION CALCULATIONS
      IF(XX-15.9)1000,1000,2000
 1000 A=0.6366197724
      B=0.5772156646
      XULT0=1.0
      SIGMA=0.0
      TY0=0.0
      TY1=0.0
      DO 7 I=1,500
      XF=I
      Z=-1.0
      Z0=XF+1.0
      SINY0=Z**(XF)
      SINY1=Z**(Z0)
      XOSH0=(XX/2.0)**(2*XF)
      XOSH1=(XX/2.0)**(2*XF-1.0)
      XULT0=XULT0*XF
      XULT1=XULT0/XF
      SIGMA=SIGMA+(1.0/XF)
      VALU0=DLOG(XX/2.0)+B-SIGMA
      VALU1=VALU0+(1.0/(2.0*XF))
      TRMY0=((SINY0*XOSH0)/(XULT0**2))*(VALU0)
      TRMY1=((SINY1*XOSH1)/(XULT0*XULT1))*(VALU1)
      TY0=TY0+TRMY0
      TY1=TY1+TRMY1
      ERY0=DABS(TRMY0/TY0)
      ERY1=DABS(TRMY1/TY1)
      IF(ERY0-ERY1)1,1,2
    1 ER=ERY1
      GO TO 10
    2 ER=ERY0
   10 IF(ER-.001)11,11,7
    7 CONTINUE
   11 Y0X=A*(DLOG(XX/2.0)+B+TY0)
      Y1X=A*((1.0/XX)-TY1)
      GO TO 100
 2000 AH=3.1415926536/4.0
      BH=XX-AH
      CH=XX+AH
      S1H=SIN(BH)
      S2H=SIN(CH)
      DHY0=S2H/(8.0*XX)
      DHY1=S1H*3.0/(8.0*XX)
      HH=XX-(3.0*AH)
      S3H=SIN(HH)
      EH=2.0/(3.1415926536*XX)
      FH=DSQRT(EH)
      XMY0=S1H-DHY0
```

```
      XMY1=S3H+DHY1
      Y0X=XMY0*FH
      Y1X=-XMY1*FH
  100 RETURN
      END
C
C     ----------------------------------------------------------------
C
      SUBROUTINE ADDFBC(H,G,HOLD,AFR,AFI,AKR,AKI,N0,N1,N2,
     1 DFI,FID,FIB,DF)
C     ****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      REAL AFR(ND,ND),AFI(ND,ND),AKR(ND),AKI(ND)
      DOUBLE PRECISION H(NU,NU),G(NU,NU),T(100,20)
      DOUBLE PRECISION FID(NU),FIB(NU),DFI(NU),HOLD(NU,NU)
      REAL UL(20,20),VB(20,20),VL(20,20),UB(20,20)
      COMPLEX XKAP(20),PSI(20,20),TEMP1(20,20),TEMP2(20,20)
      COMPLEX SUM3,DX(20)
      REAL DXR(20),DXI(20)
      REAL ULT(20,20),VLT(20,20),UBT(20,20),VBT(20,20)
      REAL AFRT(20,20),AFIT(20,20),TP(20,20)
C
C REARRANGES H AND G MATRICES TO TAKE THE BASE AND FAR BOUNDARY
C CONDITIONS INTO ACCOUNT
C
C ARRANGE H MATRIX
C REAL PART
C
      IF(ME.GT.0)THEN
      DO 20 J=1,ME
      DO 10 I=1,ME
      UL(I,J)=AFR(I,J)*AKR(J)
      VB(I,J)=AFI(I,J)*AKI(J)
      VL(I,J)=AFI(I,J)*AKR(J)
      UB(I,J)=AFR(I,J)*AKI(J)
   10 CONTINUE
   20 CONTINUE
C
C CREATE MATRIX TP WHICH TRANSFORMS VECTORS OF PRESSURES AND THEIR
C DERIVATIVES AT FAR END FROM LINEAR FORMULATION INTO CONSTANT ONE
C ( FROM NODAL POINTS INTO MIDPOINTS )
C
      CALL TRANSP(TP)
C
C
      DO 21 I=1,ME
      DO 22 J=1,ME
      TEM1=0.0
      TEM2=0.0
      TEM3=0.0
      TEM4=0.0
      DO 23 K=1,ME
```

```
      TEM1=TEM1+TP(I,K)*UL(K,J)
      TEM2=TEM2+TP(I,K)*VB(K,J)
      TEM3=TEM3+TP(I,K)*VL(K,J)
      TEM4=TEM4+TP(I,K)*UB(K,J)
   23 CONTINUE
      ULT(I,J)=TEM1
      VBT(I,J)=TEM2
      VLT(I,J)=TEM3
      UBT(I,J)=TEM4
   22 CONTINUE
   21 CONTINUE
C
      DO 41 I=1,ME
      DO 31 J=1,ME
      ULT(I,J)=ULT(I,J)-VBT(I,J)
      VLT(I,J)=VLT(I,J)+UBT(I,J)
   31 CONTINUE
   41 CONTINUE
C
      DO 61 I=1,ME
      DO 62 J=1,ME
      TEM1=0.0
      TEM2=0.0
      DO 63 K=1,ME
      TEM1=TEM1+TP(I,K)*AFR(K,J)
      TEM2=TEM2+TP(I,K)*AFI(K,J)
   63 CONTINUE
      AFRT(I,J)=TEM1
      AFIT(I,J)=TEM2
   62 CONTINUE
   61 CONTINUE
C
      DO 70 I=1,N
      DO 60 J=1,ME
      TEM1=0.0
      TEM2=0.0
      DO 50 K=1,ME
      KK=K+N2
      TEM1=TEM1+H(I,KK)*DBLE(AFRT(K,J))
      TEM2=TEM2+G(I,KK)*DBLE(ULT(K,J))
   50 CONTINUE
      T(I,J)=TEM1+TEM2
   60 CONTINUE
   70 CONTINUE
C
      DO 75 I=1,N
      DO 72 J=1,ME
      JJ=J+N2
      H(I,JJ)=T(I,J)
   72 CONTINUE
   75 CONTINUE
C
C IMAGINARY PART
C
C
```

```
      DO 120 I=1,N
      DO 110 J=1,ME
      TEM1=0.0
      TEM2=0.0
      DO 100 K=1,ME
      KK=K+N2
      TEM1=TEM1+G(I,KK)*DBLE(VLT(K,J))
      TEM2=TEM2+H(I,KK)*DBLE(AFIT(K,J))
  100 CONTINUE
      T(I,J)=TEM1+TEM2
  110 CONTINUE
  120 CONTINUE
C
      DO 140 I=1,N
      DO 130 J=1,ME
      H(N+I,J+N2)=T(I,J)
  130 CONTINUE
  140 CONTINUE
      END IF
C
      IF(CONST2.GT.1.E-08) THEN
      TEMP=CONST1*CONST2
      DO 90 I=1,N
      DO 80 J=N1+1,N2
      H(N+I,J)=G(I,J)*TEMP
   80 CONTINUE
   90 CONTINUE
      END IF
C
C FILL IN THE REST OF H MATRIX
C
C
      DO 160 I=1,N
      DO 150 J=1,N
      H(I,J+N)=-H(I+N,J)
      H(I+N,J+N)=H(I,J)
  150 CONTINUE
  160 CONTINUE
C
C FORM RIGHT HAND VECTOR
C
      DO 180 I=1,N
      DO 170 J=N0+1,N1
      DFI(I)=DFI(I)+G(I,J)*FID(J-N0)
  170 CONTINUE
  180 CONTINUE
C
      DO 200 I=1,N
      DO 190 J=N1+1,N2
      DFI(I)=DFI(I)+G(I,J)*FIB(J-N1)
  190 CONTINUE
  200 CONTINUE
C
      IF(DF.GT.0.0) THEN
      DO 211 I=1,ME
```

```
      XKAP(I)=CMPLX(AKR(I),AKI(I))
      DO 212 J=1,ME
      PSI(I,J)=CMPLX(AFRT(I,J),AFIT(I,J))
  212 CONTINUE
  211 CONTINUE
      DO 210 I=1,ME
      DO 220 J=1,ME
      TEMP1(I,J)=PSI(I,J)/(XKAP(J)*XKAP(J))
  220 CONTINUE
  210 CONTINUE
      DO 230 I=1,ME
      DO 240 J=1,ME
      SUM3=(0.0,0.0)
      DO 250 K=1,ME
      SUM3=SUM3+TEMP1(J,K)*PSI(I,K)
  250 CONTINUE
      TEMP2(J,I)=SUM3
  240 CONTINUE
  230 CONTINUE
      DO 260 I=1,ME
      DX(I)=-TEMP2(I,1)*SNGL(DF)
  260 CONTINUE
      DO 270 I=1,ME
      DXR(I)=REAL(DX(I))
      DXI(I)=AIMAG(DX(I))
  270 CONTINUE
      WRITE(IMP,101)(DXR(I),DXI(I),I=1,ME)
  101 FORMAT(2(4X,E14.5))
      DO 280 I=1,N
      DO 290 J=1,ME
      K1=J-ME+N
      DFI(I)=DFI(I)+HOLD(I,K1)*DBLE(DXR(J))
      DFI(I+N)=DFI(I+N)+HOLD(I,K1)*DBLE(DXI(J))
  290 CONTINUE
  280 CONTINUE
      END IF
      RETURN
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE SLNPD(A,B,D,NX,NU,IMP)
C     *****************************************************************
C
C
C     SOLUTION OF LINEAR SYSTEMS OF EQUATIONS
C     BY THE GAUSS ELIMINATION METHOD PROVIDING
C     FOR INTERCHANGING ROWS WHEN ENCOUNTERING
C     A  ZERO DIAGONAL COEFFICIENT
C
C     A  :SYSTEM MATRIX
C     B  :ORIGINALLY IT CONTAINS THE INDEPENDENT
C     COEFFICIENTS.  AFTER SOLUTION IT CONTAINS THE
C     VALUES OF THE SYSTEM UNKNOWNS
C
```

```
C      NX  : ACTUAL NUMBER OF UNKNOWNS
C
C      NU  : ROW AND COLUMN DIMENSION OF A
C
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       DOUBLE PRECISION A(NU,NU),B(NU)
       N1=NX-1
       DO 100 K=1,N1
       K1=K+1
       C=A(K,K)
       IF(DABS(C)-.000001)1,1,3
     1 DO 7 J=K1,NX
C
C      TRY TO INTERCHANGE ROWS TO GET NON ZERO
C      DIAGONAL COEFFICIENT
C
       IF (DABS(A(J,K))-0.000001)7,7,5
     5 DO 6 LL=K,NX
       C=A(K,LL)
       A(K,LL)=A(J,LL)
     6 A(J,LL)=C
       C=B(K)
       B(K)=B(J)
       B(J)= C
       C=A(K,K)
       GO TO 3
     7 CONTINUE
     8 WRITE(IMP,2)K
       D=0.0
       GO TO 300
C
C      DIVIDE ROW BY DIAGONAL COEFFICIENT
C
     3 C=A(K,K)
       DO 4 J=K1,NX
     4 A(K,J)=A(K,J)/C
       B(K)=B(K)/C
C
C      ELIMINATE UNKNOWN  X(K) FROM ROW I
C
       DO 10 I=K1,NX
       C=A(I,K)
       DO 9 J=K1,NX
     9 A(I,J)=A(I,J)-C*A(K,J)
    10 B(I)=B(I)-C*B(K)
   100 CONTINUE
C
C      COMPUTE LAST UNKNOWN
C
       IF (DABS(A(NX,NX))-0.000001)101,101,102
   101 WRITE(IMP,2)NX
       D=0.
       GO TO 300
   102 B(NX)=B(NX)/A(NX,NX)
C
```

```
C     APPLY BACKSUBSTITUTION PROCESS TO COMPUTE REMAINING
C     UNKNOWNS
C
      DO 200 LL=1,N1
      K=NX-LL
      K1=K+1
      DO 200 J=K1,NX
  200 B(K)=B(K)-A(K,J)*B(J)
C
C     COMPUTE VALUE OF DETERMINATE
      D=1.0
      DO 250 I=1,NX
  250 D=D*A(I,I)
      WRITE(IMP,333)D
  333 FORMAT('THE DETERMINANT =',E14.8)
    2 FORMAT('****  SINGULARITY  IN  ROW  ****',I5)
  300 RETURN
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE JACOBI(AS,BM,EIGV,X,ME,ND)
C     *****************************************************************
C
      REAL EIGV(ND),X(ND,ND), BM(ND,ND)
      REAL AS(ND,ND),D(150)
C
C     AS=A : STIFNESS MATRIX
C     BM=F : MASS MATRIX
C     NT=ME : TOTAL DOF
C     X  : FINAL EIGENVECTOR
C    EIGV: FINAL EIGENVALUE
C    IFPR=0 :FOR PRINTOUT OF INTERM. RESULTS
C    IFPR=1 :FOR PRINTOUT OF INTERM. RESULTS
C
c     CONSTANSTS
C
      NSMAX = 15
      RTOL = 10.**(-12)
      IFPR = 0
C
C
      NT= ME
C
C     INITIALIZE EIGENVALUE AND EIGENVECTOR MATRICES
C
      DO 10 I = 1,NT
      IF (AS(I,I) .GT. 0. .AND. BM(I,I) .GT. 0.) GOTO 9
      WRITE (*,2020)
      STOP
9     D(I) = AS(I,I)/BM(I,I)
      EIGV(I) = D(I)
 10   CONTINUE
      DO 30 I = 1,NT
      DO 20 J = 1,NT
```

```
 20   X(I,J) = 0.
 30   X(I,I) = 1.
      IF (NT.EQ. 1)THEN
C       WRITE (*,*)' EIGENVALUE :  ',EIGV(1)
        RETURN
        ELSE
      ENDIF

C     INITIALIZE SWEEP COUNTER AND EIGEN ITERATION
C
      NSWEEP = 0
      NR = NT - 1
 40   NSWEEP = NSWEEP + 1
      IF (IFPR .EQ. 1) WRITE (*, 2000) NSWEEP
C
C     CHECK IF PRESENT OFF-DIAGONAL ELEMENTS ARE LARGE ENOUGH TO
C     REQUIRE ZEROING
C
      EPS = (.01**NSWEEP)**2
      DO 210 J = 1,NR
      JJ = J + 1
      DO 210 K = JJ,NT
      EPTOLA = (AS(J,K)*AS(J,K))/(AS(J,J)*AS(K,K))
      EPTOLB = (BM(J,K)*BM(J,K))/(BM(J,J)*BM(K,K))
      IF((EPTOLA .LT.EPS) .AND. (EPTOLB .LT. EPS)) GOTO 210
C
C       IF ZEROING IS REQUIRED CALCULATE THE ROT MAT ELM CA ,CG
C
      AKK = AS(K,K) * BM(J,K) - BM(K,K) * AS(J,K)
      AJJ = AS(J,J) * BM(J,K) - BM(J,J) * AS(J,K)
      AB = AS(J,J) * BM(K,K) - AS(K,K) * BM(J,J)
      CHECK = (AB*AB + 4.* AKK * AJJ)/4.
      IF (CHECK .LT. 0.) THEN
        WRITE (*,2020)
        STOP
       ELSE
      ENDIF
 60   SQCH = SQRT(CHECK)
      D1 = AB/2. + SQCH
      D2 = AB/2. - SQCH
      DEN = D1
      IF (ABS(D2) .GT. ABS(D1))THEN
      DEN = D2
      ELSE
      ENDIF
      IF(DEN .EQ. 0.) THEN
 70   CA = 0.
      CG = -AS(J,K)/AS(K,K)
      ELSE
 80   CA = (AKK)/(DEN)
      CG = - (AJJ)/(DEN)
      ENDIF
C
C     PERFORM GENERALIZED ROT TO ZERO THE PRESENT OFF  DG. ELM
C
```

```
   90 IF (NT .EQ. 2) GOTO 190
  100 JP1 = J + 1
      JM1 = J - 1
      KP1 = K + 1
      KM1 = K - 1
      IF (JM1 .LT. 1) GOTO 130
  110 DO 120 I =1,JM1
      AJ = AS(I,J)
      BJ = BM(I,J)
      AK = AS(I,K)
      BK = BM(I,K)
      AS(I,J) = AJ + CG * AK
      BM(I,J) = BJ + CG * BK
      AS(I,K) = AK + CA * AJ
  120 BM(I,K) = BK + CA * BJ
  130 IF (KP1 .GT. NT) GOTO 160
  140 DO 150 I = KP1,NT
      AJ = AS(J,I)
      BJ = BM(J,I)
      AK = AS(K,I)
      BK = BM(K,I)
      AS(J,I) = AJ + CG * AK
      BM(J,I) = BJ + CG * BK
      AS(K,I) = AK + CA * AJ
  150 BM(K,I) = BK + CA * BJ
  160 IF(JP1 .GT. KM1) GOTO 190
  170 DO 180 I = JP1,KM1
      AJ = AS(J,I)
      BJ = BM(J,I)
      AK = AS(I,K)
      BK = BM(I,K)
      AS(J,I) = AJ + CG * AK
      BM(J,I) = BJ + CG * BK
      AS(I,K) = AK + CA * AJ
  180 BM(I,K) = BK + CA * BJ
  190 AK = AS(K,K)
      BK = BM(K,K)
      AS(K,K) = AK + 2.* CA * AS(J,K) + CA * CA * AS(J,J)
      BM(K,K) = BK + 2.* CA * BM(J,K) + CA * CA * BM(J,J)
      AS(J,J) = AS(J,J) + 2.* CG * AS(J,K) + CG * CG * AK
      BM(J,J) = BM(J,J) + 2.* CG * BM(J,K) + CG * CG * BK
      AS(J,K) = 0.
      BM(J,K) = 0.
C
C     UPDATE EIGENVALUE MATRIX AFTER EACH ROTAITON
C
      DO 200 I = 1,NT
      XJ = X(I,J)
      XK = X(I,K)
      X(I,J) = XJ + CG * XK
      X(I,K) = XK + CA * XJ
  200 CONTINUE
  210 CONTINUE
C
C  UPDATE EIGENVALUE AFTER EACH SWEEP
```

```
C                                                                         BEM20360
      DO 220 I = 1,NT                                                     BEM20370
      IF (AS(I,I) .GT. 0. .AND. BM(I,I) .GT. 0.) GOTO 220                 BEM20380
      WRITE (*,2020)                                                      BEM20390
      STOP                                                                BEM20400
  220 EIGV(I) = AS(I,I)/BM(I,I)                                           BEM20410
      IF (IFPR .EQ. 0) GOTO 230                                           BEM20420
      WRITE (*,2030)                                                      BEM20430
      WRITE (*,710) (EIGV(I), I = 1,NT)                                   BEM20440
C                                                                         BEM20450
C       CHECK FOR CONVERGENCE                                             BEM20460
C                                                                         BEM20470
  230 DO 240 I = 1,NT                                                     BEM20480
      TOL = RTOL * D(I)                                                   BEM20490
      DIF = ABS(EIGV(I) - D(I))                                           BEM20500
      IF (DIF .GT. TOL) GOTO 280                                          BEM20510
  240 CONTINUE                                                            BEM20520
C                                                                         BEM20530
C       CHECK ALL OFF DIAG, TO SEE IF ANOTHER SWEEP IS REQD.              BEM20540
C                                                                         BEM20550
      EPS = RTOL ** 2                                                     BEM20560
      DO 250 J = 1,NR                                                     BEM20570
      JJ = J + 1                                                          BEM20580
      DO 253 K = JJ,NT                                                    BEM20590
      EPSA = (AS(J,K)*AS(J,K))/(AS(J,J)*AS(K,K))                          BEM20600
      EPSB = (BM(J,K)*BM(J,K))/(BM(J,J)*BM(K,K))                          BEM20610
      IF((EPSA .LT. EPS) .AND. (EPSB .LT. EPS)) GOTO 253                  BEM20620
      GOTO 280                                                            BEM20630
  253 CONTINUE                                                            BEM20640
  250 CONTINUE                                                            BEM20650
C                                                                         BEM20660
C       FILL  BOTTOM  TRAIANG OF RESLT MATRICES AND SCALE EG. VECT.       BEM20670
C                                                                         BEM20680
  255 DO 260 I = 1,NT                                                     BEM20690
      DO 260 J = 1,NT                                                     BEM20700
      AS(J,I) = AS(I,J)                                                   BEM20710
  260 BM(J,I) = BM(I,J)                                                   BEM20720
      DO 270 J = 1,NT                                                     BEM20730
      BB = SQRT(BM(J,J))                                                  BEM20740
      DO 270 K = 1,NT                                                     BEM20750
  270 X(K,J) = X(K,J)/BB                                                  BEM20760
C                                                                         BEM20770
C       SORT    EIGENVALUE AND VECTOR                                     BEM20780
C                                                                         BEM20790
  510 KOUNT = 0                                                           BEM20800
      DO 520 I = 1,NT-1                                                   BEM20810
      IF (EIGV(I) .GT. EIGV(I+1)) THEN                                    BEM20820
      KOUNT = 1                                                           BEM20830
      CHNG = EIGV(I)                                                      BEM20840
      EIGV(I) = EIGV(I+1)                                                 BEM20850
      EIGV(I+1) = CHNG                                                    BEM20860
      DO 530 J = 1,NT                                                     BEM20870
      CHNG = X(J,I)                                                       BEM20880
      X(J,I) = X(J,I+1)                                                   BEM20890
      X(J,I+1) = CHNG                                                     BEM20900
```

```
  530 CONTINUE
      ENDIF
  520 CONTINUE
      IF (KOUNT .EQ. 1) GOTO 510
      IF (IFPR .EQ. 0) GOTO 276
      WRITE (*,960)
      DO 275 K = 1,NT
      WRITE (*,710) (X(K,J), J = 1,NT)
  275 CONTINUE
  276 RETURN
C
C     UPDATE D MATRIX AND START NEW SWEEP, IF ALLOWED
C
  280 DO 290 I = 1,NT
  290 D(I) = EIGV(I)
      IF (NSWEEP .LT. NSMAX) GOTO 40
      GOTO 255
  710 FORMAT (10(E12.5,1X))

  910 FORMAT (//, ' **  STIFNESS MATRIX **',/)
  920 FORMAT (9(G10.4,1X)/
     .           5X,9(G10.4,1X))
  930 FORMAT (//, ' **  MASS MATRIX **',/)
  960 FORMAT (//, '    FINAL EIGEN VECTOR OF MODAL SYSTEM',/)
 2000 FORMAT (//,'  SWEEP CYCLE IN JACOBI   ' ,I4,/)
 2020 FORMAT (10X,'ERROR SOLUTION STOP',/
     .5X,' MATRICES NOT POSITIVE DEFINITE')
 2030 FORMAT (' CURRENT EIGENVALUE IN *JACOBI* ARE',/)
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE OUTPUT(XM,YM,FID,FIB,DFI,AFR,AFI,AKR,AKI,
     * N0,N1,N2,NOUT,FLAG,DF,NC,TPR,TPI,HXR,HXI,YR,YI)
C     *****************************************************************
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      DOUBLE PRECISION XM(1),YM(1),FID(1),FIB(1),DFI(1)
      REAL AFR(ND,ND),AFI(ND,ND),AKR(ND),AKI(ND)
      REAL UL(20,20),UB(20,20),VL(20,20),VB(20,20)
      DOUBLE PRECISION PRA(20),PIA(20),QRA(20),QIA(20)
      COMPLEX XKAP(20),PSI(20,20),TEMP1(20,20),TEMP2(20,20)
      COMPLEX SUM3,DX(20),PSIT(20,20)
      REAL TEMP1R(20,20),TEMP2R(20,20)
      REAL DXR(20),DXI(20),SUM3R
      REAL ULT(20,20),UBT(20,20),VLT(20,20),VBT(20,20)
      REAL AFRT(20,20),AFIT(20,20),TP(20,20)
      REAL YR(20,500),YI(20,500)
      LOGICAL FLAG
C
C
      RES=0.
      WRITE(IMP,500)
```

```
C
C     WRITE RESULTS FOR RESERVOIR SURFACE
C
      IF(NOUT.EQ.1)THEN
      WRITE(IMP,600)
      WRITE(IMP,330)
      WRITE(IMP,600)
      WRITE(IMP,100)
      WRITE(IMP,600)
      WRITE(IMP,300)
      WRITE(IMP,600)
      DO 10 I=1,NO
      WRITE(IMP,200)XM(I),YM(I),RES,DFI(I)
   10 CONTINUE
      IF(FLAG)THEN
      WRITE(IMP,600)
      WRITE(IMP,320)
      WRITE(IMP,600)
      DO 20 I=1,NO
      WRITE(IMP,200)XM(I),YM(I),RES,DFI(I+N)
   20 CONTINUE
      END IF
      END IF
C
C     WRITE RESULTS FOR DAM FACE
C
      WRITE(IMP,600)
      WRITE(IMP,340)
      WRITE(IMP,600)
      WRITE(IMP,100)
      WRITE(IMP,600)
      WRITE(IMP,300)
      WRITE(IMP,600)
      J1=0
      DO 30 I=NO+1,N1
      J1=J1+1
      WRITE(IMP,200)XM(I),YM(I),DFI(I),FID(I-NO)
      YR(J1,NC)=DFI(I)
   30 CONTINUE
      IF(FLAG)THEN
      WRITE(IMP,600)
      WRITE(IMP,320)
      WRITE(IMP,600)
      J1=0
      DO 40 I=NO+1,N1
      J1=J1+1
      WRITE(IMP,200)XM(I),YM(I),DFI(N+I),RES
      YI(J1,NC)=DFI(N+I)
   40 CONTINUE
      END IF
C
C     WRITE RESULTS FOR RESERVOIR BOTTOM
C
      IF(NOUT.EQ.1)THEN
      WRITE(IMP,600)
```

```
      WRITE(IMP,350)
      WRITE(IMP,600)
      WRITE(IMP,100)
      WRITE(IMP,600)
      WRITE(IMP,300)
      WRITE(IMP,600)
      DO 50 I=N1+1,N2
      QR=FIB(I-N1)
      IF(CONST2.GT.1.0E-08)THEN
      QR=QR+CONST1*CONST2*DFI(I+N)
      END IF
      WRITE(IMP,200)XM(I),YM(I),DFI(I),QR
   50 CONTINUE
      IF(FLAG)THEN
      WRITE(IMP,600)
      WRITE(IMP,320)
      WRITE(IMP,600)
      DO 60 I=N1+1,N2
      QI=0
      IF(CONST2.GT.1.0E-08)THEN
      QI=QI-CONST1*CONST2*DFI(I)
      END IF
      WRITE(IMP,200)XM(I),YM(I),DFI(I+N),QI
   60 CONTINUE
      END IF
C
C     WRITE RESULTS FOR THE FAR END
C
      IF(ME.GT.0)THEN
      IF(CONST2.GT.1.0E-08)THEN
      DO 80 J=1,ME
      DO 70 I=1,ME
      UL(I,J)=AFR(I,J)*AKR(J)
      VB(I,J)=AFI(I,J)*AKI(J)
      VL(I,J)=AFI(I,J)*AKR(J)
      UB(I,J)=AFR(I,J)*AKI(J)
   70 CONTINUE
   80 CONTINUE
C
      CALL TRANSP(TP)
C
      DO 81 I=1,ME
      DO 82 J=1,ME
      TEM1=0.0
      TEM2=0.0
      TEM3=0.0
      TEM4=0.0
      DO 83 K=1,ME
      TEM1=TEM1+TP(I,K)*UL(K,J)
      TEM2=TEM2+TP(I,K)*VB(K,J)
      TEM3=TEM3+TP(I,K)*VL(K,J)
      TEM4=TEM4+TP(I,K)*UB(K,J)
   83 CONTINUE
      ULT(I,J)=TEM1
      VBT(I,J)=TEM2
```

```
      VLT(I,J)=TEM3
      UBT(I,J)=TEM4
   82 CONTINUE
   81 CONTINUE
C
      DO 105 I=1,ME
      DO 90 J=1,ME
      ULT(I,J)=ULT(I,J)-VBT(I,J)
      VLT(I,J)=VLT(I,J)+UBT(I,J)
   90 CONTINUE
  105 CONTINUE
      ELSE IF(FLAG.AND.CONST2.LE.1.0E-08)THEN
C
      CALL TRANSP(TP)
C
      DO 120 J=1,ME
      DO 110 I=1,ME
      UL(I,J)=AFR(I,J)*AKR(J)
      VL(I,J)=AFR(I,J)*AKI(J)
  110 CONTINUE
  120 CONTINUE
      DO 121 I=1,ME
      DO 122 J=1,ME
      TEM1=0.0
      TEM2=0.0
      DO 123 K=1,ME
      TEM1=TEM1+TP(I,K)*UL(K,J)
      TEM2=TEM2+TP(I,K)*VL(K,J)
  123 CONTINUE
      ULT(I,J)=TEM1
      VLT(I,J)=TEM2
  122 CONTINUE
  121 CONTINUE
      END IF
C
      IF(FLAG)THEN
      DO 125 I=1,ME
      SUM1=0
      SUM2=0
      DO 124 J=1,ME
      SUM1=SUM1+ULT(I,J)*DFI(N2+J)-VLT(I,J)*DFI(N2+J+N)
      SUM2=SUM2+VLT(I,J)*DFI(N2+J)+ULT(I,J)*DFI(N2+J+N)
  124 CONTINUE
      QRA(I)=-SUM1
      QIA(I)=-SUM2
  125 CONTINUE
      END IF
C
C
      CALL TRANSP(TP)
C
      DO 126 I=1,ME
      DO 127 J=1,ME
      TEM1=0.0
      TEM2=0.0
```

```
      DO 128 K=1,ME                                                     BEM23110
      TEM1=TEM1+TP(I,K)*AFR(K,J)                                        BEM23120
      TEM2=TEM2+TP(I,K)*AFI(K,J)                                        BEM23130
  128 CONTINUE                                                          BEM23140
      AFRT(I,J)=TEM1                                                    BEM23150
      AFIT(I,J)=TEM2                                                    BEM23160
  127 CONTINUE                                                          BEM23170
  126 CONTINUE                                                          BEM23180
C                                                                       BEM23190
      IF(CONST2.GT.1.0E-08)THEN                                         BEM23200
      IF(DF.GT.0.0) THEN                                                BEM23210
      DO 111 I=1,ME                                                     BEM23220
      XKAP(I)=CMPLX(AKR(I),AKI(I))                                      BEM23230
      DO 112 J=1,ME                                                     BEM23240
      PSIT(I,J)=CMPLX(AFRT(I,J),AFIT(I,J))                              BEM23250
      PSI(I,J)=CMPLX(AFR(I,J),AFI(I,J))                                 BEM23260
  112 CONTINUE                                                          BEM23270
  111 CONTINUE                                                          BEM23280
      DO 113 I=1,ME                                                     BEM23290
      DO 114 J=1,ME                                                     BEM23300
      TEMP1(I,J)=PSIT(I,J)/(XKAP(J)*XKAP(J))                            BEM23310
  114 CONTINUE                                                          BEM23320
  113 CONTINUE                                                          BEM23330
      DO 115 I=1,ME                                                     BEM23340
      DO 116 J=1,ME                                                     BEM23350
      SUM3=(0.0,0.0)                                                    BEM23360
      DO 117 K=1,ME                                                     BEM23370
      SUM3=SUM3+TEMP1(J,K)*PSI(I,K)                                     BEM23380
  117 CONTINUE                                                          BEM23390
      TEMP2(J,I)=SUM3                                                   BEM23400
  116 CONTINUE                                                          BEM23410
  115 CONTINUE                                                          BEM23420
      DO 118 I=1,ME                                                     BEM23430
      DX(I)=TEMP2(I,1)*SNGL(DF)                                         BEM23440
  118 CONTINUE                                                          BEM23450
      DO 119 I=1,ME                                                     BEM23460
      DXR(I)=REAL(DX(I))                                                BEM23470
      DXI(I)=AIMAG(DX(I))                                               BEM23480
  119 CONTINUE                                                          BEM23490
      END IF                                                            BEM23500
      DO 140 I=1,ME                                                     BEM23510
      SUM1=0                                                            BEM23520
      SUM2=0                                                            BEM23530
      DO 130 J=1,ME                                                     BEM23540
      SUM1=SUM1+AFRT(I,J)*DFI(N2+J)-AFIT(I,J)*DFI(N2+J+N)               BEM23550
      SUM2=SUM2+AFIT(I,J)*DFI(N2+J)+AFRT(I,J)*DFI(N2+J+N)               BEM23560
  130 CONTINUE                                                          BEM23570
      IF(DF.GT.0.0) THEN                                                BEM23580
      PRA(I)=SUM1+DXR(I)                                                BEM23590
      PIA(I)=SUM2+DXI(I)                                                BEM23600
      ELSE                                                              BEM23610
      PRA(I)=SUM1                                                       BEM23620
      PIA(I)=SUM2                                                       BEM23630
      END IF                                                            BEM23640
  140 CONTINUE                                                          BEM23650
```

```
      ELSE IF(CONST2.LE.1.0E-08.AND.FLAG)THEN
      IF(DF.GT.0.0) THEN
      DO 141 I=1,ME
      DO 142 J=1,ME
      IF(AKI(J).GT.0.0) THEN
      TEMP1R(I,J)=-AFRT(I,J)/(AKI(J)*AKI(J))
      ELSE
      TEMP1R(I,J)=AFRT(I,J)/(AKR(J)*AKR(J))
      END IF
  142 CONTINUE
  141 CONTINUE
      DO 145 I=1,ME
      DO 146 J=1,ME
      SUM3R=0.0
      DO 147 K=1,ME
      SUM3R=SUM3R+TEMP1R(J,K)*AFR(I,K)
  147 CONTINUE
      TEMP2R(J,I)=SUM3R
  146 CONTINUE
  145 CONTINUE
      DO 148 I=1,ME
      DXR(I)=TEMP2R(I,1)*SNGL(DF)
  148 CONTINUE
      END IF
      DO 160 I=1,ME
      SUM1=0
      SUM2=0
      DO 150 J=1,ME
      SUM1=SUM1+AFRT(I,J)*DFI(N2+J)
      SUM2=SUM2+AFRT(I,J)*DFI(N2+J+N)
  150 CONTINUE
      IF(DF.GT.0.0) THEN
       IF(FLAG) THEN
       PRA(I)=SUM1+DXR(I)
       PIA(I)=SUM2
       ELSE
       PRA(I)=SUM1+DXR(I)
       PIA(I)=SUM2+DXI(I)
       END IF
      ELSE
      PRA(I)=SUM1
      PIA(I)=SUM2
      END IF
  160 CONTINUE
      END IF
C
C
      IF(.NOT.FLAG)THEN
      IF(DF.GT.0.0) THEN
      DO 311 I=1,ME
      DO 312 J=1,ME
      TEMP1R(I,J)=AFRT(I,J)/(AKR(J)*AKR(J))
  312 CONTINUE
  311 CONTINUE
      DO 313 I=1,ME
```

```
      DO 314 J=1,ME
      SUM3R=0.0
      DO 315 K=1,ME
      SUM3R=SUM3R+TEMP1R(J,K)*AFR(I,J)
  315 CONTINUE
      TEMP2R(J,I)=SUM3R
  314 CONTINUE
  313 CONTINUE
      DO 316 I=1,ME
      DXR(I)=TEMP2R(I,1)*SNGL(DF)
  316 CONTINUE
      END IF
      DO 180 I=1,ME
      SUM1=0.
      DO 170 J=1,ME
      SUM1=SUM1+AFRT(I,J)*DFI(N2+J)
  170 CONTINUE
      IF(DF.GT.0.0) THEN
      PRA(I)=SUM1+DXR(I)
      ELSE
      PRA(I)=SUM1
      END IF
  180 CONTINUE
C
C
      DO 205 J=1,ME
      DO 190 I=1,ME
      UL(I,J)=AFRT(I,J)*AKR(J)
  190 CONTINUE
  205 CONTINUE
      DO 220 I=1,ME
      SUM1=0
      DO 210 J=1,ME
      SUM1=SUM1+UL(I,J)*DFI(N2+J)
  210 CONTINUE
      QRA(I)=-SUM1
  220 CONTINUE
      END IF
C
C
      WRITE(IMP,600)
      WRITE(IMP,360)
      WRITE(IMP,600)
      WRITE(IMP,100)
      WRITE(IMP,600)
      WRITE(IMP,300)
      WRITE(IMP,600)
      DO 230 I=1,ME
      K1=N2+I
      WRITE(IMP,200)XM(K1),YM(K1),PRA(I),QRA(I)
  230 CONTINUE
      IF(FLAG)THEN
      WRITE(IMP,600)
      WRITE(IMP,320)
      WRITE(IMP,600)
```

```
      DO 240 I=1,ME
      K1=N2+I
      WRITE(IMP,200)XM(K1),YM(K1),PIA(I),QIA(I)
  240 CONTINUE
      END IF
      END IF
      END IF
C
C      COMPUTE TOTAL PRESSURE ON DAM
C
      NE=N1-N0
      DY=HY/NE
      TPR=DFI(N1)
      DO 250 I=N0+1,N1-1
      TPR=TPR+DFI(I)
  250 CONTINUE
      NX=0
      TPRR=DFI(N1)*0.5
      DO 251 I=N0+1,N1-1
      NX=NX+1
      RR=REAL(NX-1)
      TPRR=TPRR+DFI(I)*((N1-N0)-RR-0.5)
  251 CONTINUE
      TPR=DY*TPR
      TPRR=DY*TPRR*DY
      HXR=TPRR/TPR
      WRITE(IMP,600)
      WRITE(IMP,370)TPR
      WRITE(IMP,371)HXR
C
C
      IF(FLAG)THEN
      TPI=DFI(N1+N)
      DO 260 I=N0+1,N1-1
      TPI=TPI+DFI(I+N)
  260 CONTINUE
      NX=0
      TPRI=DFI(N1+N)*0.5
      DO 261 I=N0+1,N1-1
      NX=NX+1
      RR=REAL(NX-1)
      TPRI=TPRI+DFI(I+N)*((N1-N0)-RR-0.5)
  261 CONTINUE
      TPI=TPI*DY
      TPRI=TPRI*DY*DY
      HXI=TPRI/TPI
      WRITE(IMP,600)
      WRITE(IMP,380)TPI
      WRITE(IMP,381)HXI
      TPRT=DSQRT(TPR**2+TPI**2)
      WRITE(IMP,600)
      WRITE(IMP,390)TPRT
      WRITE(IMP,600)
      END IF
C
```

```
C
  320 FORMAT(1X,'IMAGINARY PART')
  200 FORMAT(4(4X,E14.5))
  100 FORMAT(1X,'BOUNDARY  ELEMENTS',//
     *,10X,'X',17X,'Y',15X,'PRESSURE',5X,'PRESSURE DERIVATIVE')
  300 FORMAT(1X,'REAL PART')
  330 FORMAT(1X,'PRESSURE AND PRESSURE DERIVATIVE ON RESERVOIR SURFACE'
     *)
  340 FORMAT(1X,'PRESSURE AND PRESSURE DERIVATIVE ON DAM FACE')
  350 FORMAT(1X,'PRESSURE AND PRESSURE DERIVATIVE ON BOTTOM')
  360 FORMAT(1X,'PRESSURE AND PRESSURE DERIVATIVE ON FAR END')
  370 FORMAT(1X,'TOTAL PRESSURE ON DAM FACE, REAL PART',E14.5)
  371 FORMAT(1X,'HXR =',E14.5)
  380 FORMAT(1X,'TOTAL PRESSURE ON DAM FACE, IMAGINARY PART',E14.5)
  381 FORMAT(1X,'HXI =',E14.5)
  390 FORMAT(1X,'TOTAL PRESSURE ON DAM FACE, ABSOLUTE VALUE',E14.5)
  500 FORMAT(1X,70('*'))
  600 FORMAT(1X,' ')
      RETURN
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE ADDUND(H,G,HOLD,AFR,AKR,AKI,N0,N1,N2,DFI,FID,FIB,
     *FLAG,DF)
C     *****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      REAL AFR(ND,ND),AKR(ND),AKI(ND)
      DOUBLE PRECISION H(NU,NU),G(NU,NU),T(100,20)
      DOUBLE PRECISION FID(NU),FIB(NU),DFI(NU),HOLD(NU,NU)
      REAL U(20,20),V(20,20)
      REAL TEMP1(20,20),TEMP2(20,20)
      REAL SUM3,DXR(20)
      REAL UT(20,20),VT(20,20),AFRT(20,20),TP(20,20)
      LOGICAL FLAG
C
C REARRANGES H AND G MATRICES TO TAKE THE BASE AND FAR BOUNDARY
C CONDITIONS INTO ACCOUNT
C
C ARRANGE H MATRIX
C REAL PART
C
      IF(ME.GT.0)THEN
      DO 20 J=1,ME
      DO 10 I=1,ME
      U(I,J)=AFR(I,J)*AKR(J)
   10 CONTINUE
   20 CONTINUE
C
      CALL TRANSP(TP)
C
      DO 21 I=1,ME
```

```
      DO 22 J=1,ME
      TEM1=0.0
      DO 23 K=1,ME
      TEM1=TEM1+TP(I,K)*U(K,J)
   23 CONTINUE
      UT(I,J)=TEM1
   22 CONTINUE
   21 CONTINUE
C
      IF(FLAG)THEN
      DO 40 J=1,ME
      DO 30 I=1,ME
      V(I,J)=AFR(I,J)*AKI(J)
   30 CONTINUE
   40 CONTINUE
C
      DO 41 I=1,ME
      DO 42 J=1,ME
      TEM1=0.0
      DO 43 K=1,ME
      TEM1=TEM1+TP(I,K)*V(K,J)
   43 CONTINUE
      VT(I,J)=TEM1
   42 CONTINUE
   41 CONTINUE
      END IF
C
C
      DO 51 I=1,ME
      DO 52 J=1,ME
      TEM1=0.0
      DO 53 K=1,ME
      TEM1=TEM1+TP(I,K)*AFR(K,J)
   53 CONTINUE
      AFRT(I,J)=TEM1
   52 CONTINUE
   51 CONTINUE
C
      DO 70 I=1,N
      DO 60 J=1,ME
      SUM1=0
      SUM2=0
      DO 50 K=1,ME
      KK=K+N2
      SUM1=SUM1+H(I,KK)*DBLE(AFRT(K,J))
      SUM2=SUM2+G(I,KK)*DBLE(UT(K,J))
   50 CONTINUE
      T(I,J)=SUM1+SUM2
   60 CONTINUE
   70 CONTINUE
C
      DO 75 I=1,N
      DO 72 J=1,ME
      JJ=J+N2
      H(I,JJ)=T(I,J)
```

```
   72 CONTINUE
   75 CONTINUE
C
C IMAGINARY PART
C
      IF(FLAG)THEN
      DO 120 I=1,N
      DO 110 J=1,ME
      SUM1=0
      DO 100 K=1,ME
      KK=K+N2
      SUM1=SUM1+G(I,KK)*DBLE(VT(K,J))
  100 CONTINUE
      T(I,J)=SUM1
  110 CONTINUE
  120 CONTINUE
C
      DO 140 I=1,N
      DO 130 J=1,ME
      H(N+I,J+N2)=T(I,J)
  130 CONTINUE
  140 CONTINUE
C
C FILL IN THE REST OF H MATRIX
C
C
      DO 160 I=1,N
      DO 150 J=1,N
      H(I,J+N)=-H(I+N,J)
      H(I+N,J+N)=H(I,J)
  150 CONTINUE
  160 CONTINUE
      END IF
      END IF
C
C FORM RIGHT HAND VECTOR
C
      DO 180 I=1,N
      DO 170 J=N0+1,N1
      DFI(I)=DFI(I)+G(I,J)*FID(J-N0)
  170 CONTINUE
  180 CONTINUE
C
      DO 200 I=1,N
      DO 190 J=N1+1,N2
      DFI(I)=DFI(I)+G(I,J)*FIB(J-N1)
  190 CONTINUE
  200 CONTINUE
C
      IF(DF.GT.0.0) THEN
      DO 210 I=1,ME
      DO 220 J=1,ME
      IF(AKI(J).GT.0.0) THEN
      TEMP1(I,J)=-AFRT(I,J)/(AKI(J)*AKI(J))
      ELSE
```

```
      TEMP1(I,J)=AFRT(I,J)/(AKR(J)*AKR(J))
      END IF
  220 CONTINUE
  210 CONTINUE
      DO 230 I=1,ME
      DO 240 J=1,ME
      SUM3=0.0
      DO 250 K=1,ME
      SUM3=SUM3+TEMP1(J,K)*AFR(I,K)
  250 CONTINUE
      TEMP2(J,I)=SUM3
  240 CONTINUE
  230 CONTINUE
      DO 260 I=1,ME
      DXR(I)=-TEMP2(I,1)*SNGL(DF)
  260 CONTINUE
      DO 290 I=1,N
      DO 300 J=1,ME
      K1=J-ME+N
      DFI(I)=DFI(I)+HOLD(I,K1)*DBLE(DXR(J))
  300 CONTINUE
  290 CONTINUE
      END IF
      RETURN
      END
C
C     -----------------------------------------------------------------
C
      SUBROUTINE TRANSP(TP)
C     *****************************************************************
C
C TRANSP COMPUTES THE MATRIX OF TRANSFOERMATION TP
C FOR ADJUSTMENT OF LINEARLY FORMED FAR END CONDITION
C INTO CONSTANT ELEMENT FORMULATION
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/MAIN/ CONST,N,LEC,IMP,NU,ND
      COMMON/ADD/ HY,PI,ME,MF,CONST1,CONST2
      REAL TP(20,20)
C
      DO 10 I=1,ME
      DO 20 J=1,ME
      TP(I,J)=0.0
   20 CONTINUE
   10 CONTINUE
C
      DO 30 I=1,ME-1
      N1=I+1
      DO 40 J=I,N1
      TP(I,J)=0.5
   40 CONTINUE
   30 CONTINUE
      TP(ME,ME)=0.5
*     DO 50 I=1,ME
*     WRITE(6,*)(TP(I,J),J=1,ME)
```

```
*  50 CONTINUE                                                          BEM27510
      RETURN                                                            BEM27520
      END                                                               BEM27530
C                                                                       BEM27540
C                                                                       BEM27550
C****** THIS IS THE LAST CARD OF THE CONSTANT BEM PROGRAM****************BEM27560
                                                                        BEM27570
                                                                        BEM27580
                                                                        BEM27590
                                                                        BEM27600
                                                                        BEM27610
                                                                        BEM27620
                                                                        BEM27630
                                                                        BEM27640
```

```
RRRRRRRRRR    SSSSSSSSS    CCCCCCCCC    SSSSSSSSS        444         444    2222222222    666666666
RRRRRRRRRRR  SSSSSSSSSSS  CCCCCCCCCCC  SSSSSSSSSSS      4444        4444   22222222222  66666666666
RR       RR  SS       SS  CC       CC  SS       SS     44 44       44 44   22       22  66       66
RR       RR  SS           CC           SS             44  44      44  44            22  66
RR       RR  SSS          CC           SSS           44   44     44   44            22  66
RRRRRRRRRRR   SSSSSSSSS   CC            SSSSSSSSS   4444444444  4444444444         22   666666666
RRRRRRRRRR     SSSSSSSS   CC             SSSSSSSS  44444444444 44444444444        22    66666666666
RR   RR             SSS   CC                  SSS        44          44         22      66       66
RR    RR             SS   CC                   SS        44          44        22       66       66
RR     RR    SS      SS   CC       CC  SS      SS        44          44       22        66       66
RR      RR   SSSSSSSSSSS  CCCCCCCCCCC  SSSSSSSSSSS       44          44    22222222222  66666666666
RR       RR   SSSSSSSSS    CCCCCCCCC    SSSSSSSSS        44          44    22222222222   666666666


EEEEEEEEEEE  NN       NN  DDDDDDDDD                     444         444    2222222222    666666666
EEEEEEEEEEE  NNN      NN  DDDDDDDDDD                   4444        4444   22222222222  66666666666
EE           NNNN     NN  DD      DD                  44 44       44 44   22       22  66       66
EE           NN NN    NN  DD       DD                44  44      44  44            22  66
EE           NN  NN   NN  DD       DD               44   44     44   44            22  66
EEEEEEE      NN   NN  NN  DD       DD              4444444444  4444444444         22   666666666
EEEEEEE      NN    NN NN  DD       DD             44444444444 44444444444        22    66666666666
EE           NN     NNNN  DD       DD                   44          44         22      66       66
EE           NN      NNN  DD       DD                   44          44        22       66       66
EE           NN       NN  DD      DD                    44          44       22        66       66
EEEEEEEEEEE  NN       NN  DDDDDDDDDD                    44          44    22222222222  66666666666
EEEEEEEEEEE  NN        N  DDDDDDDDD                     44          44    22222222222   666666666

********************************************************************************************************************
********************************************************************************************************************
********************************************************************************************************************
********************************************************************************************************************
```

B.3 Program FOURTN FORTRAN A1

RM 151

M 20

JABLONSKI

RSCS1011

DATE/TIME= SEP 18, 1990 16:15:46 | PRINT: CLASS= E | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP | FORMS= S1PT | RM. 151
NODES O\E\C=NRCVM01 \NRCVM01 \NRCVM01 | FCB= STD8 | M-20
JOB = J1011 START | UCS= GT15 |

```
C     PROGRAM FOURTN
C     FOR CALCULATION OF THE RESPONSE OF THE LINEAR
C     DAM-RESERVOIR-FOUNDATION SYSTEM
C     TO THE EXCITATION HISTORY GIVEN IN THE DISCRETE
C     FORM A(NT*DT)
C     BY MEANS OF THE FFT TECHNIQUE
C
C     PROGRAMMER: ALEXANDER M. JABLONSKI
C
C     DATE: APRIL 25, 1989
C           SUBSTANTIALLY MODIFIED  JULY 11, 1989
C
C     CENTER: STRUCTURES SECTION
C             INSTITUTE FOR RESEARCH IN CONSTRUCTION
C             NATIONAL RESEARCH COUNCIL, CANADA
C
C     NOTE: THIS PROGRAM IS BASED ON THE THREE PROGRAMS:
C           FOUR21 - INITIAL VERSION OF THE FFT PROGRAM;
C           ACCEL  - READING A PROCESSED EARTHQUAKE DATA AND CONVERTING
C                    IT INTO INPUT DATA (ACCELERATION VECTOR)
C           INPUTF - READING AN OUTPUT DATA FILE FROM THE CONSTANT
C                    BOUNDARY ELEMENT PROGRAM BEMC2DN AND CONVERTING
C                    IT INTO INPUT DATA (RESPONSE TO THE HARMONIC
C                    EXCITATION)
C     PROGRAM FOURTN EMPLOYS THE FFT ROUTINES ORIGINALLY DEVELOPED
C     BY PAUL SWARZTRAUBER, NATIONAL CENTER FOR ATMOSPHERIC RESEARCH,
C     DENVER, COLORADO, USA, AND IN THE FORM PROVIDED BY THE NEW VERSION
C     1.0 OF THE MATH LIBRARY, IMSL.
C
      PROGRAM FOURTN
      PARAMETER (NT=4096)
      PARAMETER (NC=200)
      PARAMETER (ND=10)
C     NT = NUMBER OF STEPS TO BE EMPLOYED IN THE FFT
C     NC = INITIAL NUMBER OF FREQUENCY STEPS EMPLOYED TO CALCULATE
C          HARMONIC RESPONSE
C     ND = NUMBER OF NODES AT DAM FACE INTRODUCED IN BEMC2DN PROGRAM
C     AC(I) = PROCESSED ACCELERATION VECTOR FROM EARTHQUAKE RECORDS
C     A(I)  = MODIFIED ACCELERATION VECTOR - INPUT TO THE FORWARD FFT
C             ROUTINE
C     TX(I) = VECTOR OF TOTAL HYDRODYNAMIC FORCE (I=1,NC)
C     Y1(I) = MODIFIED VECTOR OF TOTAL HYDRODYNAMIC FORCE (I=1,NT)
C     YC(I,J) = HYDRODYNAMIC PRESSURE MATRIX (I=1,ND, J=1,NC)
C     YC1(I,J)= MODIFIED PRESSURE MATRIX (I=1,ND, J=1,NT)
C
      COMPLEX AC(NT), YC(ND,NC), TX(NC), HX(NC)
      COMPLEX YC1(ND,NT), Y1(NT)
      COMPLEX Y2(NT), Y2N(NT), AN(NT)
      REAL A(NT), RWSKP(24611)
      REAL Y1R(NT),Y1I(NT)
C
C
      CHARACTER*22 INPS1,INFT1,OUTFFT
C
      COMMON/WORKSP/ RWSKP
```

```
      DATA INPS1/'/INPS1 INPUT A1'/
      DATA INFT1/'/INFT1 INPUT A1'/
      DATA OUTFFT/'/OUTFFT OUTPUT A1'/
C
      OPEN(2,FORM='FORMATTED',STATUS='OLD',FILE=INPS1)
      OPEN(3,FORM='FORMATTED',STATUS='OLD',FILE=INFT1)
      OPEN(5,FORM='FORMATTED',STATUS='NEW',FILE=OUTFFT)
C
C     READ INPUT DATA FILE CREATED BY PROGRAM ACCEL
C
      N=1+(NT/6)
      DO 10 J=1,N
      NX=(J-1)*6
      READ(2,100)(A(I+NX),I=1,6)
   10 CONTINUE
  100 FORMAT(6F12.3)
C
C     DIVIDE THE ACCELERATION VECTOR BY 100 TO HAVE IT IN M/SEC**2
C
      DO 20 I=1,NT
      A(I)=A(I)/100
   20 CONTINUE
C
      DO 30 I=1,NT
      AC(I)=CMPLX(A(I),0.0)
   30 CONTINUE
C
C     CARRY OUT FORWARD FFT
C
      CALL IWKIN(24611)
      CALL FFTCF(NT,AC,AN)
C
C
C     READ REQUIRED HARMONIC RESPONES TX, HX, YC
C
C
      DO 31 I=1,NT
      WRITE(5,*)I,AN(I)
   31 CONTINUE
      WRITE(5,300)
  300 FORMAT(//,1X,70('*'),//)
C
      DO 40 I=1,NC
      READ(3,*)I,TX(I),HX(I)
      READ(3,300)
      DO 50 J=1,ND
      READ(3,*)YC(J,I)
   50 CONTINUE
   40 CONTINUE
C
C     MODIFY HARMONIC RESPONSE TO HAVE IT IN THE FORM READY
C     FOR FFT ROUTINE
C
      DO 60 I=1,NT
      IF(I.GT.NC) THEN
```

```
      Y1(I)=(0.0,0.0)
      GO TO 60
      END IF
C
C     NORMALIZATION OF THE VECTOR Y1(I) W.R.T. HYDROSTATIC FORCE
C     IN THIS CASE FOR H=100 M (HEIGHT OG THE DAM)
C
      Y1(I)=TX(I)/5000
   60 CONTINUE
C
      DO 61 I=1,NT
      Y1R(I)=REAL(Y1(I))
      Y1I(I)=AIMAG(Y1(I))
   61 CONTINUE
C
      DO 62 I=1,NT/2
      Y1R(I+NT/2)=Y1R(NT/2-I+1)
   62 CONTINUE
C
      DO 63 I=1,NT/2
      Y1I(I+NT/2)=-Y1I(NT/2-I+1)
   63 CONTINUE
C
      DO 64 I=1,NT
      Y1(I)=CMPLX(Y1R(I),Y1I(I))
   64 CONTINUE
C
      DO 65 I=1,NT
      WRITE(5,*)I,Y1(I)
   65 CONTINUE
      WRITE(5,300)
C
      DO 70 I=1,NT
      DO 80 J=1,ND
      IF(I.GT.NC) THEN
      YC1(J,I)=(0.0,0.0)
      GO TO 80
      END IF
      YC1(J,I)=YC(J,I)/100
   80 CONTINUE
   70 CONTINUE
C
C     CARRY OUT INVERSE FFT FOR A CHOSEN HARMONIC RESPONSE VECTOR
C
      DO 90 I=1,NT
      Y2(I)=AN(I)*Y1(I)
   90 CONTINUE
C
C     CARRY INVERSE FFT
C
      CALL FFTCB(NT,Y2,Y2N)
C
      DO 101 I=1,NT
      Y2N(I)=Y2N(I)/NT
  101 CONTINUE
```

```
C
      DO 110 I=1,NT
      WRITE(5,*)Y2N(I)
  110 CONTINUE
C
      STOP
      END
C
C     END OF THE FOURTN PROGRAM
```

```
RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS       11          00000000        11            11
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS     111         0000000000      111           111
RR        RR  SS        SS  CC        CC  SS        SS    1111        00      0000    1111          1111
RR        RR  SS            CC            SS                11        00     00 00      11            11
RR        RR  SSS           CC            SSS               11        00    00  00      11            11
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS        11        00   00   00      11            11
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS       11        00  00    00      11            11
RR    RR               SSS  CC                     SSS      11        00 00     00      11            11
RR     RR               SS  CC                      SS      11        0000      00      11            11
RR      RR    SS        SS  CC        CC  SS        SS      11        000       00      11            11
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  1111111111     0000000000   1111111111    1111111111
RR        RR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS   1111111111      00000000    1111111111    1111111111
```

```
EEEEEEEEEEEE  NN        NN  DDDDDDDDD                       11          00000000        11            11
EEEEEEEEEEEE  NNN       NN  DDDDDDDDDD                     111         0000000000      111           111
EE            NNNN      NN  DD       DD                   1111        00      0000    1111          1111
EE            NN NN     NN  DD        DD                    11        00     00 00      11            11
EE            NN  NN    NN  DD        DD                    11        00    00  00      11            11
EEEEEEEE      NN   NN   NN  DD        DD                    11        00   00   00      11            11
EEEEEEEE      NN    NN  NN  DD        DD                    11        00  00    00      11            11
EE            NN     NN NN  DD        DD                    11        00 00     00      11            11
EE            NN      NNNN  DD        DD                    11        0000      00      11            11
EE            NN       NNN  DD       DD                     11        000       00      11            11
EEEEEEEEEEEE  NN        NN  DDDDDDDDDD                  1111111111     0000000000   1111111111    1111111111
EEEEEEEEEEEE  NN         N  DDDDDDDDD                   1111111111      00000000    1111111111    1111111111
```

```
**********************************************************************************************************************************
**********************************************************************************************************************************
**********************************************************************************************************************************
**********************************************************************************************************************************
```

B.4 Program SPLOTN2 FORTRA A1

```
RRRRRRRRRRR   MM        MM                      11     555555555555      11
RRRRRRRRRRRR  MMM      MMM                     111     555555555555     111
RR        RR  MMMM    MMMM                    1111     55              1111
RR        RR  MM MM  MM MM                      11     55                11
RR        RR  MM  MMMM  MM                      11     55                11
RRRRRRRRRRRR  MM   MM   MM                      11     555555555         11
RRRRRRRRRRR   MM        MM                      11     5555555555        11
RR    RR      MM        MM                      11              55       11
RR     RR     MM        MM                      11               55      11
RR      RR    MM        MM                      11               55      11
RR       RR   MM        MM                  1111111111 555555555555  1111111111
RR        RR  MM        MM                  1111111111 55555555555   1111111111

                     MM        MM           2222222222    00000000
                     MMM      MMM          222222222222  0000000000
                     MMMM    MMMM          22        22 00      0000
                     MM MM  MM MM                    22 00     00 00
                     MM  MMMM  MM                    22 00    00  00
                     MM   MM   MM                   22  00   00   00
                     MM        MM                 22    00  00    00
                     MM        MM               22      00 00     00
                     MM        MM             22        0000      00
                     MM        MM           22          000       00
                     MM        MM          222222222222  0000000000
                     MM        MM          222222222222   00000000

  JJJJJJJJJJ  AAAAAAAAAA   BBBBBBBBBBB  LL           000000000000 NN        NN  SSSSSSSSSS  KK        KK
  JJJJJJJJJJ AAAAAAAAAAAA  BBBBBBBBBBBB LL           000000000000 NNN       NN SSSSSSSSSSSS KK       KK
      JJ     AA        AA  BB        BB LL           00        00 NNNN      NN SS        SS KK      KK
      JJ     AA        AA  BB        BB LL           00        00 NN NN     NN SS           KK     KK
      JJ     AA        AA  BB        BB LL           00        00 NN  NN    NN SSS          KK    KK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB   LL           00        00 NN   NN   NN  SSSSSSSSS   KKKKKKK
      JJ     AAAAAAAAAAAA  BBBBBBBBBB   LL           00        00 NN    NN  NN   SSSSSSSSS  KKKKKKK
      JJ     AA        AA  BB        BB LL           00        00 NN     NN NN         SSS  KK    KK
JJ    JJ     AA        AA  BB        BB LL           00        00 NN      NNNN          SS  KK     KK
JJ    JJ     AA        AA  BB        BB LL           00        00 NN       NNN SS        SS KK      KK
JJJJJJJJ     AA        AA  BBBBBBBBBBBB LLLLLLLLLLLL 000000000000 NN        NN SSSSSSSSSSSS KK       KK
 JJJJJJ      AA        AA  BBBBBBBBBBB  LLLLLLLLLLLL 000000000000 NN         N  SSSSSSSSSS  KK        KK

RRRRRRRRRRR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS   777777777777   00000000   777777777777  9999999999
RRRRRRRRRRRR SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  77777777777   0000000000  77777777777  999999999999
RR        RR SS        SS  CC        CC  SS        SS  77        77  00      0000 77        77 99        99
RR        RR SS            CC            SS                      77  00     00 00          77  99        99
RR        RR SSS           CC            SSS                    77   00    00  00         77   99        99
RRRRRRRRRRRR  SSSSSSSSS    CC             SSSSSSSSS            77    00   00   00        77    999999999999
RRRRRRRRRRR    SSSSSSSSS   CC              SSSSSSSSS           77    00  00    00        77    999999999999
RR    RR             SSS   CC                     SSS          77    00 00     00        77              99
RR     RR             SS   CC                      SS          77    0000      00        77              99
RR      RR   SS        SS  CC        CC  SS        SS          77    000       00        77    99        99
RR       RR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS          77     0000000000         77    999999999999
RR        RR  SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS           77      00000000          77     9999999999
```

DATE/TIME= AUG 09, 1990 16:20:04 | PRINT: CLASS= E | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP | FORMS= S1PT | RM. 151
NODES O\E\C=NRCVM01 \NRCVM01 \NRCVM01 | FCB= STD8 | M-20
JOB = J7079 START | UCS= GT15 |

```
C     PROGRAM SPLOTN
C.    CALLS SET OF DISSPLA ROUTINES TO CREATE SET OF PLOTS
C
C     PROGRAMMER: ALEXANDER M. JABLONSKI
C
C     DATE: JULY 12, 1989
C     MODIFIED: JANUARY 31, 1990
C     MODIFIED: APRIL 18, 1990
C
C     CENTER: STRUCTURES SECTION
C             INSTITUTE FOR RESEARCH IN CONSTRUCTION
C             NATIONAL RESEARCH COUNCIL, CANADA
C
      PROGRAM SPLOTN
C
C     IT PRODUCES THE FOLLOWING PLOTS:
C     1. ACCELERATION RECORD
C     2. REAL PART OF DFT OF ACCEL.
C     3. IMAG PART OF DFT OF ACCEL.
C     4. REAL PART OF HARMONIC RESPONSE
C     5. IMAG PART OF HARMONIC RESPONSE
C     6. FINAL EARTHQUAKE RESPONSE
C
      PARAMETER (NT=4096)
      PARAMETER (NE=4096)
      COMPLEX AN(NE),Y2N(NE),Y1(NE)
      REAL X(NE),Y(NE),A(NE),AA(NE)
      REAL X1(NE),X2(NE),X3(NE)
      REAL ANR(NE),ANI(NE),Y2NR(NE),Y2NI(NE)
      REAL Y1R(NE),Y1I(NE),Z(NE)
      CHARACTER*22 INPS1,OUTFFT,SUM1
C
      DATA INPS1/'/INPS1 INPUT A1'/
      DATA OUTFFT/'/OUTFFT OUTPUT A1'/
      DATA SUM1/'/SUM1 OUTPUT A1'/
      OPEN(2,FORM='FORMATTED',STATUS='OLD',FILE=INPS1)
      OPEN(3,FORM='FORMATTED',STATUS='OLD',FILE=OUTFFT)
      OPEN(4,FORM='FORMATTED',STATUS='NEW',FILE=SUM1)
C
      CALL NOMDEV(1)
C
      PI=3.141592653
      DO 10 I=1,NE
      X(I)=REAL(I)
   10 CONTINUE
C
      N=1+(NT/6)
      DO 20 J=1,N
      NX=(J-1)*6
      READ(2,100)(A(I+NX),I=1,6)
   20 CONTINUE
  100 FORMAT(6F12.3)
C
      DO 30 I=1,NE
      IF(I.GT.NT) THEN
```

```
      Y(I)=0.0
      END IF
      Y(I)=(REAL(A(I)))/100.
   30 CONTINUE
C
      DO 399 I=1,NE
      X1(I)=0.02*X(I)
  399 CONTINUE
C
      CALL PLOT1(NE,X1,Y)
C
      DO 31 I=1,NE
      READ(3,*)I,AN(I)
   31 CONTINUE
      READ(3,300)
  300 FORMAT(//,1X,70('*'),//)
C
      DO 32 I=1,NE
      ANR(I)=REAL(AN(I))
      ANI(I)=IMAG(AN(I))
   32 CONTINUE
C
C
      CALL PLOT2(NE,X,ANR)
C
      CALL PLOT3(NE,X,ANI)
C
C
      DO 40 I=1,NE
      READ(3,*)I,Y1(I)
   40 CONTINUE
      READ(3,300)
C
      DO 41 I=1,NE
      Y1R(I)=REAL(Y1(I))
      Y1I(I)=IMAG(Y1(I))
      Z(I)=SQRT(Y1I(I)**2+Y1R(I)**2)
   41 CONTINUE
C
      NX=200
      DO 410 I=1,NX
      X2(I)=(0.3068*X(I))/(2*PI)
  410 CONTINUE
C
      CALL PLOT4(NX,X2,Y1R)
C
      CALL PLOT5(NX,X2,Y1I)
C
      CALL PLOT51(NX,X2,Z)
C
C
      DO 33 I=1,NE
      AA(I)=SQRT(ANR(I)**2+ANI(I)**2)
   33 CONTINUE
C
```

```
      CALL PLOT31(NE,X,AA)
C
      DO 50 I=1,NE
      READ(3,*)Y2N(I)
   50 CONTINUE
C
      DO 51 I=1,NE
      Y2NR(I)=REAL(Y2N(I))
      Y2NI(I)=IMAG(Y2N(I))
      WRITE(4,*)Y2NR(I)
   51 CONTINUE
C
      DO 510 I=1,NE
      X3(I)=0.02*X(I)
  510 CONTINUE
C
      CALL PLOT6(NE,X3,Y2NR)
C
      CALL DONEPL
      STOP
      END
C
      SUBROUTINE PLOT1(N,X,Y)
      REAL X(N),Y(N)
      RMAX=0.0
      RMIN=0.0
      DO 5 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
    5 CONTINUE
      DO 6 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      R1=I*0.02
      ELSE IF(RMIN.EQ.Y(I)) THEN
      R2=I*0.02
      END IF
    6 CONTINUE
      YMAX=+2.5
      YMIN=-2.5
      XMIN=0
      XMAX=90.0
      XSTP=10.0
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=0.5
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TRIPLX
      CALL TITLE('TAFT SCHOOL HOR.',16,'TIME SEC',8,
     *'ACCELERATION M/SEC/SEC',22,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
```

```
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+2,4.8,6)
      CALL REALNO(RMIN,+2,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL REALNO(R1,+2,6.6,6)
      CALL REALNO(R2,+2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
C
      SUBROUTINE PLOT2(N,X,Y)
      REAL X(N),Y(N)
      YMAX=+200.0
      YMIN=-200.0
      XMIN=0
      XMAX=4500
      XSTP=500
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=50.0
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('DFT OF ACCEL. - REAL PART',25,
     *'STEPS, 1 STEP=0.3068 RAD/S',26,'AMPLITUDE OF ACCELERATION',25,
     *9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
      SUBROUTINE PLOT3(N,X,Y)
      REAL X(N),Y(N)
      YMAX=+200.0
      YMIN=-200.0
      XMIN=0
      XMAX=4500
      XSTP=500
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=50.0
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('DFT OF ACCEL. - IMAG. PART',26,
```

```
     *'STEPS, 1 STEP=0.3068 RAD/S',26,'AMPLITUDE OF ACCELERATION',25,
     *9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
      SUBROUTINE PLOT4(N,X,Y)
      REAL X(N),Y(N)
      PI=3.141592653
      YMAX=+0.2
      YMIN=-0.2
      XMIN=0
      XMAX=10
      XSTP=1
      DO 5 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
    5 CONTINUE
      DO 6 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      R1=(I*0.3068)/(2*PI)
      ELSE IF(RMIN.EQ.Y(I)) THEN
      R2=(I*0.3068)/(2*PI)
      END IF
    6 CONTINUE
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      DO 11 I=2,N
      IF(Y(I).GE.Y(I-1)) GO TO 11
      YM=Y(I)
   11 CONTINUE
      YSTP=0.05
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('HARM. RESPONSE - REAL PART',26,
     *'FREQUENCY HZ',12,'TOTAL HYDR. FORCE/STAT. FORCE',
     *29,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+2,4.8,6)
      CALL REALNO(RMIN,+2,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL REALNO(R1,+2,6.6,6)
      CALL REALNO(R2,+2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
```

```
      CALL ENDPL(0)
      RETURN
      END
C
      SUBROUTINE PLOT5(N,X,Y)
      REAL X(N),Y(N)
      PI=3.141592653
      YMAX=+0.2
      YMIN=-0.2
      XMIN=0
      XMAX=10
      DO 5 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
    5 CONTINUE
      DO 6 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      R1=(I*0.3068)/(2*PI)
      ELSE IF(RMIN.EQ.Y(I)) THEN
      R2=(I*0.3068)/(2*PI)
      END IF
    6 CONTINUE
      XSTP=1
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=0.05
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('HARM. RESPONSE - IMAG. PART',27,
     *'FREQUENCY HZ',12,'TOTAL HYDR. FORCE/STAT. FORCE',
     *29,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+2,4.8,6)
      CALL REALNO(RMIN,+2,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL REALNO(R1,+2,6.6,6)
      CALL REALNO(R2,+2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
      SUBROUTINE PLOT6(N,X,Y)
      REAL X(N),Y(N)
      YMAX=+0.1
      YMIN=-0.1
      XMIN=0
      XMAX=90.0
```

```
      XSTP=10.0
      DO 5 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
    5 CONTINUE
      DO 6 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      R1=I*0.02
      ELSE IF(RMIN.EQ.Y(I)) THEN
      R2=I*0.02
      END IF
    6 CONTINUE
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=0.025
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('TAFT SCHOOL RESPONSE',20,'TIME SEC'
     *,8,'TOTAL HYDR. FORCE/STAT. FORCE',29,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+3,4.8,6)
      CALL REALNO(RMIN,+3,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL REALNO(R1,+2,6.6,6)
      CALL REALNO(R2,+2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
      SUBROUTINE PLOT31(N,X,Y)
      REAL X(N),Y(N)
      YMAX=+200.0
      YMIN=-200.0
      XMIN=0
      XMAX=4500
      XSTP=500
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=50.0
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('DFT OF ACCEL. - ABS. VALUE',26,
     *'STEPS, 1 STEP=0.3068 RAD/S',26,'AMPLITUDE OF ACCELERATION',
     *25,9.0,6.5)
```

```
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
      SUBROUTINE PLOT51(N,X,Y)
      REAL X(N),Y(N)
      PI=3.141592653
      YMAX=+0.2
      YMIN=-0.2
      XMIN=0
      XMAX=10
      XSTP=1
      DO 5 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
    5 CONTINUE
      DO 6 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      R1=(I*0.3068)/(2*PI)
      ELSE IF(RMIN.EQ.Y(I)) THEN
      R2=(I*0.3068)/(2*PI)
      END IF
    6 CONTINUE
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=0.05
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TITLE('HARM. RESPONSE - ABS. VALUE',27,
     *'FREQUENCY HZ',12,'TOTAL HYDR. FORCE/STAT. FORCE',
     *29,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+2,4.8,6)
      CALL REALNO(RMIN,+2,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL REALNO(R1,+2,6.6,6)
      CALL REALNO(R2,+2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
```

```
RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS    777777777777    00000000    777777777777    9999999999
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS   77777777777    0000000000   77777777777    999999999999
RR        RR  SS        SS  CC        CC  SS        SS   77        77   00      0000  77        77   99        99
RR        RR  SS            CC            SS                      77    00     00 00           77    99        99
RR        RR  SSS           CC            SSS                    77     00    00  00          77     99        99
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS            77      00   00   00         77      999999999999
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS           77      00  00    00         77      999999999999
RR    RR              SSS   CC                    SSS           77      00 00     00         77                99
RR     RR              SS   CC                     SS           77      0000      00         77                99
RR      RR    SS        SS  CC        CC  SS        SS          77      000       00         77      99        99
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS          77       0000000000          77      999999999999
RR        RR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS           77        00000000           77       9999999999


EEEEEEEEEEEE  NN        NN  DDDDDDDDD                    777777777777    00000000    777777777777    9999999999
EEEEEEEEEEEE  NNN       NN  DDDDDDDDDD                   77777777777    0000000000   77777777777    999999999999
EE            NNNN      NN  DD       DD                  77        77   00      0000  77        77   99        99
EE            NN NN     NN  DD        DD                          77    00     00 00           77    99        99
EE            NN  NN    NN  DD        DD                         77     00    00  00          77     99        99
EEEEEEEE      NN   NN   NN  DD        DD                        77      00   00   00         77      999999999999
EEEEEEEE      NN    NN  NN  DD        DD                        77      00  00    00         77      999999999999
EE            NN     NN NN  DD        DD                        77      00 00     00         77                99
EE            NN      NNNN  DD        DD                        77      0000      00         77                99
EE            NN       NNN  DD       DD                         77      000       00         77      99        99
EEEEEEEEEEEE  NN        NN  DDDDDDDDDD                          77       0000000000          77      999999999999
EEEEEEEEEEEE  NN         N  DDDDDDDDD                           77        00000000           77       9999999999


**************************************************************************************************************************************
**************************************************************************************************************************************
**************************************************************************************************************************************
**************************************************************************************************************************************
```

B.5 Program SUM FORTRA A1

```
RRRRRRRRRRR   MM        MM                 11      555555555555      11
RRRRRRRRRRRR  MMM      MMM                111      555555555555     111
RR        RR  MMMM    MMMM               1111      55              1111
RR        RR  MM MM  MM MM                 11      55                11
RR        RR  MM  MMMM  MM                 11      55                11
RRRRRRRRRRRR  MM   MM   MM                 11      555555555         11
RRRRRRRRRRR   MM        MM                 11      5555555555        11
RR    RR      MM        MM                 11               55       11
RR     RR     MM        MM                 11                55      11
RR      RR    MM        MM                 11                55      11
RR       RR   MM        MM             1111111111  555555555555  1111111111
RR        RR  MM        MM             1111111111  55555555555   1111111111

MM        MM          2222222222    00000000
MMM      MMM         222222222222  0000000000
MMMM    MMMM         22        22  00      0000
MM MM  MM MM                   22  00     00 00
MM  MMMM  MM                   22  00    00  00
MM   MM   MM                  22   00   00   00
MM        MM                 22    00  00    00
MM        MM               22      00 00     00
MM        MM             22        0000      00
MM        MM           22          000       00
MM        MM         222222222222  0000000000
MM        MM         222222222222   00000000

  JJJJJJJJJJ   AAAAAAAAAA   BBBBBBBBBBB   LL            000000000000  NN        NN   SSSSSSSSSS   KK        KK
  JJJJJJJJJJ  AAAAAAAAAAAA  BBBBBBBBBBBB  LL            000000000000  NNN       NN  SSSSSSSSSSSS  KK       KK
      JJ      AA        AA  BB        BB  LL            00        00  NNNN      NN  SS        SS  KK      KK
      JJ      AA        AA  BB        BB  LL            00        00  NN NN     NN  SS            KK     KK
      JJ      AA        AA  BB       BB   LL            00        00  NN  NN    NN  SSS           KK    KK
      JJ      AAAAAAAAAAAA  BBBBBBBBBB    LL            00        00  NN   NN   NN   SSSSSSSSS    KKKKKKK
      JJ      AAAAAAAAAAAA  BBBBBBBBBB    LL            00        00  NN    NN  NN    SSSSSSSSS   KKKKKKK
      JJ      AA        AA  BB       BB   LL            00        00  NN     NN NN          SSS   KK    KK
JJ    JJ      AA        AA  BB        BB  LL            00        00  NN      NNNN           SS   KK     KK
JJ    JJ      AA        AA  BB        BB  LL            00        00  NN       NNN  SS        SS  KK      KK
JJJJJJJJ      AA        AA  BBBBBBBBBBBB  LLLLLLLLLLLL  000000000000  NN        NN  SSSSSSSSSSSS  KK       KK
 JJJJJJ       AA        AA  BBBBBBBBBBB   LLLLLLLLLLLL  000000000000  NN         N   SSSSSSSSSS   KK        KK

RRRRRRRRRRR    SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS   777777777777   3333333333        11         00000000
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  77777777777   333333333333      111        0000000000
RR        RR  SS        SS  CC        CC  SS        SS  77        77  33          33   1111        00      0000
RR        RR  SS            CC            SS                     77               33     11        00     00 00
RR        RR  SSS           CC            SSS                   77                33     11        00    00  00
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS           77               3333     11        00   00   00
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS          77               3333     11        00  00    00
RR    RR              SSS   CC                    SSS          77                 33     11        00 00     00
RR     RR              SS   CC                     SS          77                 33     11        0000      00
RR      RR    SS        SS  CC        CC  SS        SS         77         33      33     11        000       00
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS         77        333333333333  1111111111  0000000000
RR        RR   SSSSSSSSSS    CCCCCCCCCC    SSSSSSSSSS          77         3333333333   1111111111   00000000
```

```
DATE/TIME= AUG 13, 1990 12:10:54      | PRINT: CLASS= E      | DR. A.JABLONSKI
USER ACCOUNT= ALEXNMP                 |    FORMS= S1PT       | RM. 151
NODES O\E\C=NRCVM01 \NRCVM01 \NRCVM01 |    FCB=  STD8        | M-20
JOB = J7310 START                     |    UCS=  GT15        |
```

```
C
C     PROGRAM SUM
C     CALCULATES THE SUM OF EARTHQUAKE RESPONSES
C     TO VERTICAL AND HORIZONTAL COMPONENTS
C     OF GROUND MOTION
C
C     PROGRAMMER: ALEXANDER M. JABLONSKI
C
C     DATE: AUGUST 2, 1989
C     MODIFIED: FEBRUARY 1, 1990
C
C     CENTER: STRUCTURAL SECTION
C             INSTITUTE FOR RESEARCH IN CONSTRUCTION
C             NATIONAL RESEARCH COUNCIL, CANADA
C
      PROGRAM SUM
C
C
      PARAMETER(NT=4096)
      REAL EH(NT), EV(NT), E(NT), X(NT)
C
      CHARACTER*22 SUM1H,SUM1V
      DATA SUM1H/'/SUM1H OUTPUT A1'/
      DATA SUM1V/'/SUM1V OUTPUT A1'/
      OPEN(2,FORM='FORMATTED',STATUS='OLD',FILE=SUM1H)
      OPEN(3,FORM='FORMATTED',STATUS='OLD',FILE=SUM1V)
C
      CALL NOMDEV(1)
C
      READ(2,*)(EH(I),I=1,NT)
      READ(3,*)(EV(I),I=1,NT)
C
      DO 10 I=1,NT
      E(I)=EH(I)+EV(I)
   10 CONTINUE
C
      DO 20 I=1,NT
      X(I)= 0.005*REAL(I)
   20 CONTINUE
      CALL PLOT1(NT,X,E)
      CALL DONEPL
      STOP
      END
C
      SUBROUTINE PLOT1(N,X,Y)
      REAL X(N),Y(N)
      RMAX=0.0
      RMIN=0.0
      DO 15 I=1,N
      RMAX=MAX(RMAX,Y(I))
      RMIN=MIN(RMIN,Y(I))
   15 CONTINUE
      DO 16 I=1,N
      IF(RMAX.EQ.Y(I)) THEN
      I1=I*0.005
```

```
      ELSE IF(RMIN.EQ.Y(I)) THEN
      I2=I*0.005
      END IF
   16 CONTINUE
      YMAX=+0.5
      YMIN=-0.5
      XMIN=0
      XMAX=25.0
      XSTP=2.5
      DO 10 I=1,N
      IF(Y(I).GT.YMAX) YMAX=Y(I)
      IF(Y(I).LT.YMIN) YMIN=Y(I)
   10 CONTINUE
      YSTP=0.10
      CALL PAGE(11.0,8.5)
      CALL NOBRDR
      CALL PHYSOR(1.0,1.0)
      CALL TRIPLX
      CALL TITLE('MANIC3.1 COMB. SAGUENAY CHICOUTIMI RESPONSE',43,'
     &TIME SEC',8,'TOTAL HYDR. FORCE/STAT. FORCE',29,9.0,6.5)
      CALL GRAF(XMIN,XSTP,XMAX,YMIN,YSTP,YMAX)
      CALL MESSAG('MAX = ',6,4.0,6)
      CALL MESSAG('MIN = ',6,4.0,5.5)
      CALL REALNO(RMAX,+3,4.8,6)
      CALL REALNO(RMIN,+3,4.8,5.5)
      CALL MESSAG('AT X = ',7,5.8,6)
      CALL MESSAG('AT X = ',7,5.8,5.5)
      CALL INTNO(I1,6.6,6)
      CALL INTNO(I2,6.6,5.5)
      CALL STRTPT(0.0,3.25)
      CALL CONNPT(9.0,3.25)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      RETURN
      END
C
C     THIS IS THE END OF PROGRAM SUM
```

```
RRRRRRRRRRR    SSSSSSSSSS     CCCCCCCCCC     SSSSSSSSSS    777777777777   3333333333        11         00000000
RRRRRRRRRRRR  SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS  77777777777   333333333333     111        0000000000
RR        RR  SS        SS  CC        CC  SS        SS  77      77    33        33    1111       00      0000
RR        RR  SS            CC            SS                   77             33      11       00     00 00
RR        RR  SSS           CC            SSS                 77              33      11       00    00  00
RRRRRRRRRRRR   SSSSSSSSS    CC             SSSSSSSSS         77             3333      11       00   00   00
RRRRRRRRRRR     SSSSSSSSS   CC              SSSSSSSSS        77             3333      11       00  00    00
RR    RR               SSS  CC                     SSS       77               33      11       00 00     00
RR     RR               SS  CC                      SS       77               33      11       0000      00
RR      RR    SS        SS  CC        CC  SS        SS       77       33        33      11       000       00
RR       RR   SSSSSSSSSSSS  CCCCCCCCCCCC  SSSSSSSSSSSS       77       333333333333  1111111111    0000000000
RR        RR   SSSSSSSSSS     CCCCCCCCCC     SSSSSSSSSS        77        3333333333   1111111111     00000000
```

```
EEEEEEEEEEEE  NN        NN  DDDDDDDDD                   777777777777   3333333333        11         00000000
EEEEEEEEEEEE  NNN       NN  DDDDDDDDDD                  77777777777   333333333333     111        0000000000
EE            NNNN      NN  DD       DD                 77      77    33        33    1111       00      0000
EE            NN NN     NN  DD        DD                       77             33      11       00     00 00
EE            NN  NN    NN  DD        DD                      77              33      11       00    00  00
EEEEEEEE      NN   NN   NN  DD        DD                     77             3333      11       00   00   00
EEEEEEEE      NN    NN  NN  DD        DD                     77             3333      11       00  00    00
EE            NN     NN NN  DD        DD                     77               33      11       00 00     00
EE            NN      NNNN  DD        DD                     77               33      11       0000      00
EE            NN       NNN  DD       DD                      77       33        33      11       000       00
EEEEEEEEEEEE  NN        NN  DDDDDDDDDD                       77       333333333333  1111111111    0000000000
EEEEEEEEEEEE  NN         N  DDDDDDDDD                        77        3333333333   1111111111     00000000
```

****
****
****
****