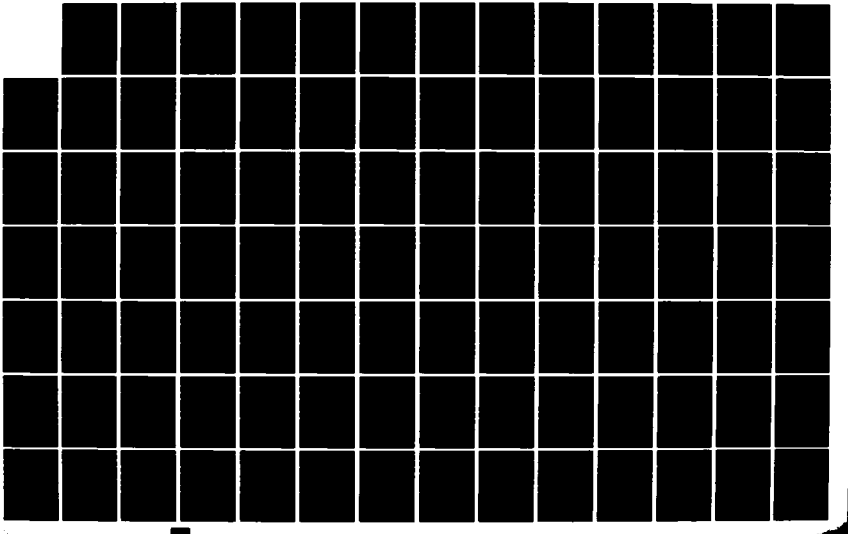
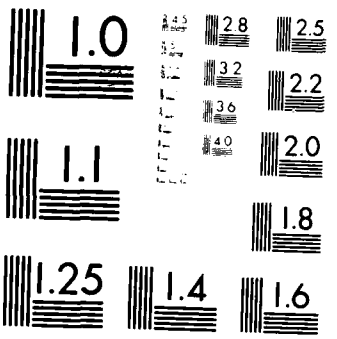
AD-A151 922 A COMPUTER PROGRAM FOR CONSOLIDATION AND DYNAMIC RESPONSE ANALYSIS OF FLU.. (U) OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS B L ABOUSTIT ET AL. JUN 83 1/2

UNCLASSIFIED OSURF-715107-84-5 AFOSR-TR-85-0266 F/G 8/13 NL





RF Project 763420/715107
Annual Report

A COMPUTER PROGRAM FOR CONSOLIDATION AND DYNAMIC RESPONSE ANALYSIS OF FLUID-SATURATED MEDIA

Ranbir S. Sandhu, B. Aboustit, S. J. Hong and M. S. Hiremath
Department of Civil Engineering

DEPARTMENT OF THE AIR FORCE
Air Force Office of Scientific Research
Bolling Air Force Base, D.C. 20332

Grant No. AFOSR-83-0055

June, 1984

The Ohio State University Research Foundation
1314 Kinnear Road
Columbus, Ohio 43212

UNCLASSIFIED

AD-A151 922

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION: Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT: Approved for Public Release Distribution Limited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S): OSURF - 715107 - 84 - 5	5 MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR
6a NAME OF PERFORMING ORGANIZATION: The Ohio State Research Foundation; 6b OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State and ZIP Code): Department of Civil Engineering, 131 Hitchcock Road, Columbus, Ohio 43210	7b ADDRESS (City, State and ZIP Code)
8a NAME OF FUNDING/SPONSORING ORGANIZATION: Air Force Office of Scientific Research; 8b OFFICE SYMBOL (If applicable): AFOSR/NA	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: AFOSR-83-0055
8c ADDRESS (City, State and ZIP Code): Bolling AFB, DC 20332-6448	10 SOURCE OF FUNDING NOS: PROGRAM ELEMENT NO 61102F; PROJECT NO 2307; TASK NO C1; WORK UNIT NO
11 TITLE (Include Security Classification): A Computer Program for Consolidation and Dynamic Response Analysis of Fluid-Saturated Media (UNCLASSIFIED)	
12 PERSONAL AUTHOR(S): Baher L. Aboustit, Ranbir S. Sandhu, Soon J. Hong and Mahantesh S. Hiremath	
13a TYPE OF REPORT: INTERIM; 13b TIME COVERED: FROM 1 Feb. '83 TO 31 Jan 84; 14 DATE OF REPORT (Yr, Mo, Day): June, 1984; 15 PAGE COUNT: 112	
16 SUPPLEMENTARY NOTATION	
17 COSATI CODES: FIELD, GROUP, SUB GR	18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Computer Simulation, Finite Element Method Flow through porous media seepage Seismic Response, Consolidation, Dynamic Response,

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A computer program was developed for evaluation of finite element models for soil consolidation and study of dynamic response of fluid-saturated soils. One- and two-dimensional consolidation problems were analyzed using different finite elements. Transient response of saturated porous elastic media for dynamic as well as quasi-static problems was studied. Results were compared with the numerical and analytical solutions available.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED X SAME AS RPT DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL: Lt Col Lawrence D Hokanson	22b TELEPHONE NUMBER (Include Area Code): 202/767-4935; 22c OFFICE SYMBOL: AFOSR/NA

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

OSURF-715107-84-5

A COMPUTER PROGRAM FOR CONSOLIDATION AND DYNAMIC RESPONSE ANALYSIS OF FLUID-SATURATED MEDIA

By

Ranbir S. Sandhu, B. Aboustit, S. J. Hong and M. S. Hiremath
Department of Civil Engineering

June 1984

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
Grant: AFOSR-83-0055

Geotechnical Engineering Report No. 14

The Ohio State University Research Foundation
1314 Kinnear Road, Columbus, Ohio 43212

FOREWORD

The investigation reported herein is part of the research project at The Ohio State University, Columbus, Ohio supported by the Air Force Office of Scientific Research Grant 83-00-55. Lt. Col. John J. Allen was the Program Manager at the commencement of the Project. Lt. Col. Lawrence D. Hokanson was the Program Manager from July 1,1983. The Research project was started in February 1, 1983 and is continuing. The present report documents part of the work done up to January 31, 1984. At The Ohio State University, the Project is being supervised by Dr. Ranbir S. Sandhu, Professor, Department of Civil Engineering. The computer program modification reported herein were carried out by Dr. Baher L. Aboustit, Post-doctoral Research Associate. The documentation of this report was done by Graduate Research Associates Soon-jo Hong and Mahantesh S. Hiremath. The Instruction and Research Computer Center at The Ohio State University provided the computational facilities.

ABSTRACT

A computer program was developed for evaluation of finite element models for soil consolidation and study of dynamic response of fluid-saturated soils. One- and two-dimensional consolidation problems were analysed using different finite elements. Transient response of saturated porous elastic media for dynamic as well as quasi-static problems was studied Results were compared with the numerical and analytical solutions available.

TABLE OF CONTENTS

LIST OF FIGURES

SECTION I

INTRODUCTION

Numerical performance of Ghaboussi and Wilson's [1] isoparametric bilinear quadrilateral element was compared with that of Sandhu [2,3,4] for analysis of quasi-static flow of an incompressible fluid through a linear elastic saturated soil in reference [5]. Both these elements were used to solve Terzaghi's problem of one-dimensional consolidation and Gibson's [6] analysis of two-dimen§ional plain strain consolidation. The results from the computer program developed for this purpose were compared with the well-known theoretical solutions.

Reference [7] used one-dimensional linear element and 4-node and 8-node isoparametric two-dimensional elements for numerical solution of several one-dimensional problems for quasi-static as well as dynamic loading conditions. Here again, the results of the computer analysis were checked with the exact solutions available in literature.

This report forms the continuation of the investigation carried out in [5] and [7]. The computer program developed for implementaion of the theory presented in [5] and [7] is described here. The program is capable of handling one- and two-dimensional problems for steady-state and transient solutions for soil consolidation. Quasi-static and dynamic loading conditions have been included. Compressible as well as incom-

pressible fluids are permitted. Time domain integration is carried out by using Wilson's [1] β-γ-θ method. However, the program is limited for linear analysis of soil consolidation i.e. small deformations, linear elastic soil behavior, constant permeability and satuzated soils.

The computer program described, is writen in FORTRAN IV language, with which engineers are well versed. The program is modular and versatile enough to classify as a research tool, in which 'modules' can be changed or added as desired. In order to have a computer program, which can solve many types of finite element problems, a macro programming language is introduced. The macro programming language ia associated with a set of compact subprograms, each desired to compute one or at most a few basic steps in a finite element solution process. This concept also permits inclusion of different solution algorithms. Dynamic storage allocation has been used to economise on storage. Thus, each array is variably dimensioned by using a set of pointers established in the control program.

Section II discusses the program capability including types of problems analysed, elements used, field equations adopted and solution process incorporated. Section III gives program structure with flow charts for individual problems, nesting of subroutines along with brief discussion on each subroutine. Section IV includes the instructions for program usage, particularly data input module. A complete listing of the program is placed in the Appendix.

SECTION II

PROGRAM NARRATIVE

2.1 PURPOSE OF THE COMPUTER PROGRAM

The computer program obtains numerical solution using finite element method for one- and two-dimensional consolidation problems. The details are available in [5]. Displacements and pore water pressure are the field variables used in these problems. The program also analyses the dynamic response of fluid saturated porous media. Quasi-static analysis is also included. Solid displacements and relative displacement of fluid with respect to soil skeleton are used as unknown quantities in these problems. The time domain integration is carried out by β-γ-θ method [7].

2.2 METHOD OF ANALYSIS

The finite element is used for obtaining numerical solutions. The method is well-documented in literature and details of these are not included in this report. Variational principle leading to matrix fomulation used in case of consolidation analysis is represented in [5], while Galerkin finite element method used in response study of saturated soils is presented in detail in [7]. The theoretical background for time domain integration using Wilson's [1] β-γ-θ method is also given in [7].

2.3 FINITE ELEMENTS USED

For soil consolidation analysis [5], two types of elements are used viz. Ghaboussi and Wilson's [1] 6-4 element and Sandhu's [3] 8-4 composite element. Ghaboussi's element is more like 4-4 isoparametric element, but uses additional 'local' nodes that give it a character of a 'higher order scheme'. Sandhu's element uses 8-point biquadratic interpolation for dispalcements and 4-point isoparametric quadrilateral for fluid pressure. In response analysis of fluid-saturated soil in [7], three types of elements have been used. These are one-dimensional linear element, 4-node isoparametric element and 8-node isoparametric element. Details about finite element formulations for these two analyses and element development is given in [5] and [7].

2.4 ASSUMTIONS MADE BY THE PROGRAMMER

In consolidation analysis [5], the fluid was assumed to be incompressible in solving both the one- and two-dimensional problems. For other material properties, refer to [5]. In 2-dimensional analysis, Poisson's ratio was set equal to zero. Also, the finite domain was modeled by a finite one in the numerical schemes. Following alternative boundary conditions were specified at the boundary, where half space was cut off.

1. Pore fluid pressure and horizontal displacements prescribed.
2. Fluid flux and horizontal displacements precribed.

3. Pore presure and traction prescribed.

In temporal discretization, piecewise linear variation of displacement and pore fluid pressure was assumed.

In the report on response analysis of fluid-saturated porous medium [7], following problems were solved using the computer program :

1. Quasi-static soil consolidation.
2. Dynamic response of an elastic layer of single material.
3. Response of fluid-saturated soil layer.

All the problems deal with the compressional wave propagation in an initially undisturbed, homogeneous, isotropic, elastic, porous or non-porous system due to dynamic loading. Solutions for non-porous media were accomplished by prescribing relative displacement between soil and fluid to be zero. Quasi-static consolidation problems were solved by setting density and damping to zero. For Ghaboussi's [1] problem, damping was assumed to be linear combination of stiffness and mass matrix as given in [1]. The time domain integration scheme using Wilson's [1] β-δ-θ method utilized various values of θ and θ for stability analysis. In analysis of Garg's [7] problems with 'strong' and 'weak' coupling different values of permeability and time-steps were used. Further theoretical details are given in [7].

2.5 FIELD EQUATIONS OF ANALYSES

2.5.1 Equations Governing Linear Elastic Soil Consolidation

Assuming pore water to be compressible, the equations of force equilibrium of elementary volumes and mass continuity, over the spatial region of interest R, may be written in standard indicial notation as

$$[E_{klij}u_{k,l}]_{,i} + \alpha\pi_{,j} + f_j = 0 \tag{1}$$

$$[K_{ij}(\pi_{,i} + \rho_2 f_i)]_{,j} + u_{j,j} = \frac{1}{M}\dot{\pi} \tag{2}$$

where u_i, f_i, E_{klij}, K_{ij}, denote the cartesian components, respectively, the displacement vector, the body force vector per unit mass, the isothermal elasticity tensor and the permeability tensor. ρ is the mass density of the saturated soil and ρ_2 that of water. π is the pore water pressure, α is the solid compressibility and M is a measure of fluid compressibility. With these field equations we associate the following boundary conditions;

$$\begin{aligned} u_i &= \hat{u}_i && \text{on } S_{1i} \\ t_i &= \tau_{ij}n_j = \hat{t}_i && \text{on } S_{2i} \\ \pi &= \hat{\pi} && \text{on } S_3 \\ Q &= q_i n_i = \hat{Q} && \text{on } S_4 \end{aligned} \tag{3}$$

Here, t_i, q_i are components of the traction and fluid flux vectors associated with surfaces embedded in the closure of R. τ_{ij} are components of the total stress tensor. S_{1i}, S_{2i} are complementary subsets of the boundary of the spatial region of interest and so are S_3, S_4. Even though the equations given above apply to compressible fluids, the applications reported herein assumed incompressible fluid i.e. $M \to \infty$. The initial conditions for the problem are

$$u_i(x_j, 0) = u_i(0) \quad \text{on R}$$

$$\pi(x_j, 0) = (0) \quad \text{on R}$$

2.5.2 Equations Governing Dynamics of Fluid-Saturated Media

Biot's [8] equations of motion for an elastic porous medium saturated with a compressible fluid may be written in standard indicial notation as;

$$[E_{ijkl}u_{k,l} + \alpha M(\alpha u_{k,k} + w_{k,k})]_{,j} + \rho f_i = \rho \ddot{u}_i + \frac{1}{f}\rho_2 \ddot{w}_i \qquad (4)$$

$$[M(\alpha u_{k,k} + w_{k,k})]_{,i} + \frac{1}{f}\rho_2 f_i = \frac{1}{f}\rho_2 \ddot{u}_i + \frac{1}{f^2}\rho_2 \ddot{w}_i + \frac{1}{K}\dot{w}_i \qquad (5)$$

where u_i, w_i, f_i, E_{ijkl} denote the cartesian components, respectively, of the solid displacement vector, the relative fluid displacement vector, the body force vector per unit mass and the isothermal elasticity tensor. ρ is mass density of the saturated soil and ρ_2 that of water per unit bulk volume. f, K, α, M are, respectively, the porosity, the perme-

ability, the solid compressibility and the fluid compressibility. The superposed dot implies a time derivative. All the functions are defined over the cartesian product $R \times [0,\infty)$ where R is the spatial region of interest and $[0,\infty)$ is the positive interval of time. With these field equations, we associate the following boundary conditions.

$$u_i(t) = \hat{u}_i(t) \quad \text{on } S_{1i} \tag{6}$$

$$t_i = \tau_{ij} n_j = (E_{ijkl} u_{k,l} + \alpha \pi \delta_{ij}) n_j = \hat{t}_i \quad \text{on } S_{2i} \tag{7}$$

$$\pi(t) = M(\alpha u_{k,k} + w_{k,k}) = \hat{\pi}(t) \quad \text{on } S_3 \tag{8}$$

$$w_i(t) = \hat{w}_i(t) \quad \text{on } S_4 \tag{9}$$

where S_{1i}, S_{2i} are complementary subsets of the boundary S of the spatial region of interest and so are S_3, S_4. The initial conditions for the problem are given by:

$$u(\underset{\sim}{x}, 0) = u_0(x)$$

$$\dot{u}(\underset{\sim}{x}, 0) = \dot{u}_0(x)$$

$$w(\underset{\sim}{x}, 0) = w_0(x)$$

$$\dot{w}(\underset{\sim}{x}, 0) = \dot{w}_0(x)$$

2.6 SOLUTION PROCESS

2.6.1 Consolidation Analysis

Once the finite element discretization is done, the solution process is initiated by setting up the 'spatial stiffness matrix', 'spatial flow matrix' and 'coupling matrix' for each element. For further details refer to [5]. The element load vectors are also set up as given in [5]. After assembly is done as usual, geometric boundary conditions are applied and solutions for displacements and pore fluid pressures are obtained for the load under consideration. The load vector is updated for subsequent time intervals and the process is continued till all time-increments are completed.

2.6.2 Dynamic Response Analysis

Following the finite element formulation presented in [7], the first step in dynamic response analysis is setting up of element stiffness, mass and damping matrices and load vectors. The standard assembly process follows. The time domain integration is used to solve for solid displacements and relative fluid displacements, which are the unknown quantities for this analysis. This is used for calculation of stresses. The process is repeated for successive time intervals, while values of load vector are updated at the end of each time interval.

SECTION III

PROGRAM STRUCTURE

3.1 INTRODUCTION

The computer program developed for soil consolidation and dynamic response analysis is general in nature and modular in structure. each module is designated to carry out one or at most a few basic steps in finite element solution process. The use of macro programming language is ideally suited for a such a comprehensive program, which is capable of solving many types of finite element problems. Individual modules can be easily modified or added to further the capabilities of this program. The use of dynamic storage allocation permits economic use of storage capacity. The program is user friendly and easy to follow once macro programming concept is understood.

3.2 FLOW CHARTS

The flow charts for soil consolidation and dynamic response analysis are shown in figures (1) and (2).

For Quasistatic Consolidation Analysis

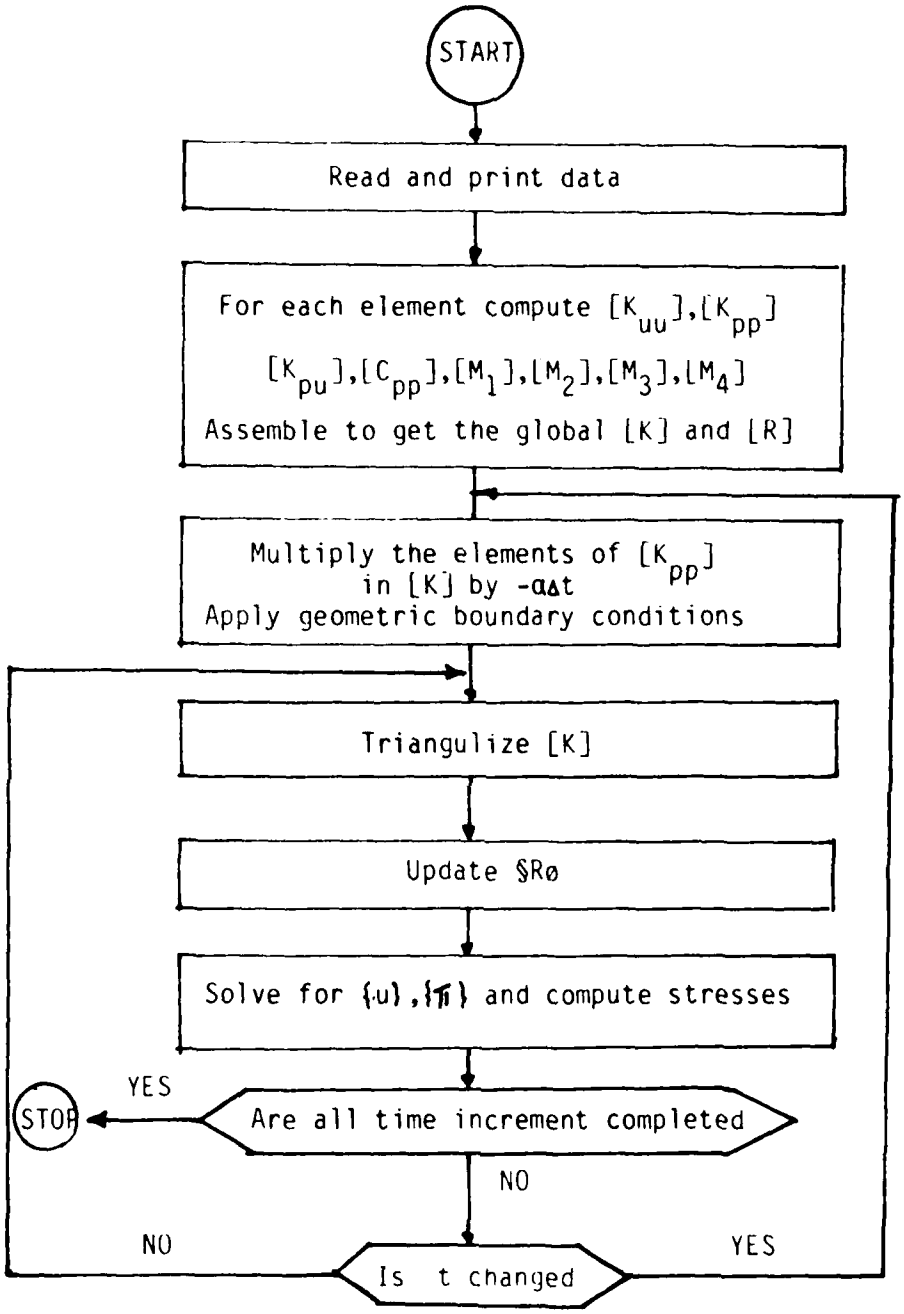


Figure 1: Flow Chart For Consolidation Analysis

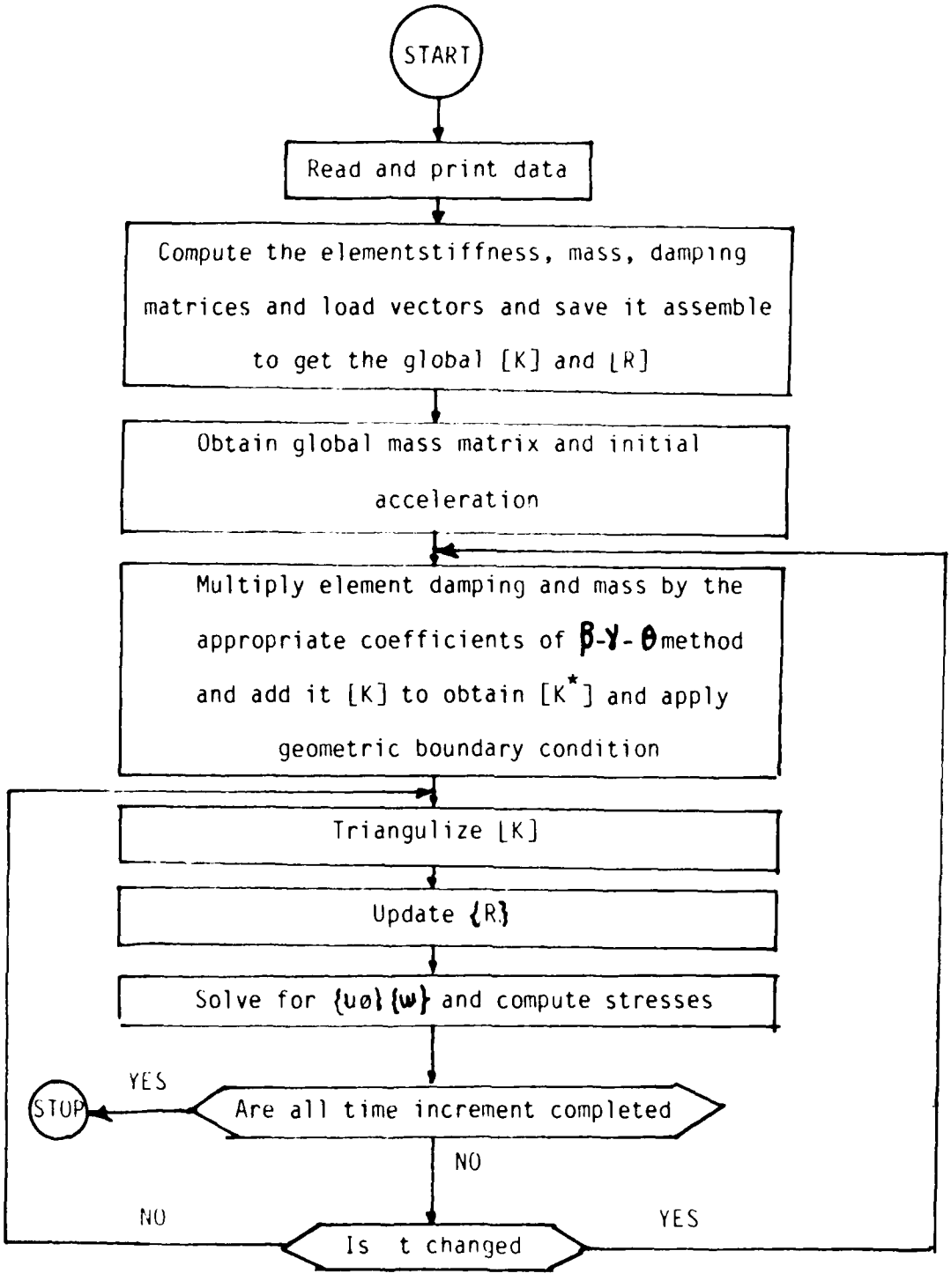


Figure 2: Flow Chart For Dynamic Response Analysis

3.3 NESTING OF SUBROUTINE

The various subroutines used and their structural relationship is shown in figure (3).

3.4 SUBROUTINES

Following paragraphs outline the operations performed in each program unit.

3.4.1 MAIN

In this unit, the program capacity is set up. It calls the subroutine ACNTRL, which is the principal control unit of the entire program.

3.4.2 Subroutine ACNTRL

This subroutine is called by the unit MAIN. It forms the central core unit of the program and calls several other subroutines to carry out specific operations. It also reads in the control data and sets the pointers for dynamic storage allocation.

3.4.3 Subroutine ACTCOL

This subroutine is called by ACNTRL and is in active column profile symmetric equation solver. The columns above the diagonal or the rows below the diagonal are stored in a single subscript array and a pointer is used to locate the diagonal elements.

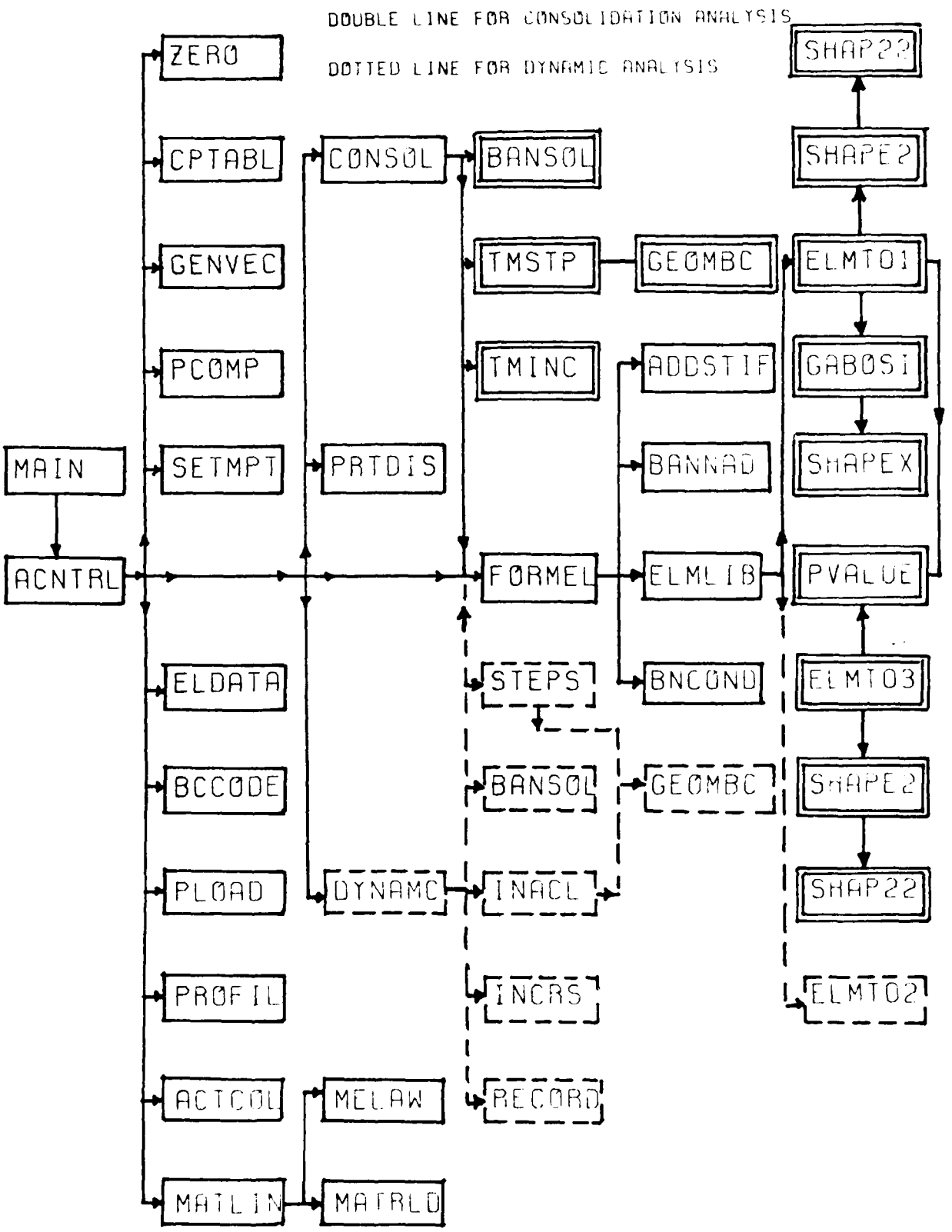


Figure 3: Nesting Of Subroutines

3.4.4 Subroutine ADDSTF

This subroutine, called by subroutine FORMEL, assembles the global arrays by direct stiffness method. This assembly is for stiffness, mass and forces. Logical variables are set to define which array or arrays are to be assembled.

3.4.5 Subroutine BANADD

Subroutine FORMEL calls this unit to assemble the element arrays (EK) and (EF) to form the global arrays viz. (GK) and (GF). The global arrays are stored in compacted form.

3.4.6 Subroutine BANSOL

This subroutine is called by ACNTRL and CONSOL subroutines. It triangularises a banded and symmetzic coefficient matrix and then solves the system of linear equations.

3.4.7 Subroutine BCCODE

This subroutine is called by ACNTRL and reads and generates boundary constraints code for each node.

3.4.8 Subroutine BNCOND

This subroutine is called by FORMEL, to modify force vector according to element boundary conditions.

3.4.9 Subroutine CPTABL

This subroutine is called by ACNTRL to estimate time required for setting up stiffness matrix for each element.

3.4.10 Subroutine CONSOL

This is prime control unit for the consolidation analysis. It is called by ACNTRL and sets up the finite element scheme by calling several other subroutines.

3.4.11 Subroutine DOT

This function subprogram is used to carry out the vector dot product and is used by subroutines ACTCOL.

3.4.12 Subroutine DYNAMC

This is a prime unit for dynamic analysis part of the program and calls a series of subroutines to complete the scheme of dynamic analysis.

3.4.13 Subroutine ELDATA

This subroutine is called by ACNTRL to read and/or generate element data.

3.4.14 Subroutine ELMLIB

This subroutine is called by FORMEL and is essentially a element library generation unit. It calls in element subroutines marked for particular problems.

3.4.15 Subroutine ELMT01

This subroutine is for Ghaboussi's element called as 6-4 quadrilateral element used in one-dimensional consolidation problem.

3.4.16 Subroutine ELMT02

This element subroutine sets up the stiffness, mass and damping matrices in the dynamic analysis problem. It also evaluates the stresses at the center of each element. It is used for one-dimensional dynamic response analysis problems.

3.4.17 Subroutine ELMT03

This subroutine is similar to ELMT10 but uses 8-noded quadrilateral element. It is used for plane strain dynamic response analysis.

3.4.18 Subroutine FORMEL

This subroutine called by ACNTRL and loops over each element to set up element matrices and later assemble them into global arrays.

3.4.19 Subroutine GENVEC

This subroutine called by ACNTRL, formulates the nodal geometry data either by reading it or generating it. Linear interpolation is used for this purpose.

3.4.20 Subroutine GEOMBC

This subroutine called by TMSTP, STEPS and INACL and involves applying kinematic constraints for each degree of freedom.

3.4.21 Subroutine GABOSI

This is a part of the scheme to set up Ghaboussi's element and is called by ELMT09 for tthis purpose.

3.4.22 Subroutine INACL

This subroutine is called by DYNAMC and assemles mass matrix and solves the dynamic equation of motion for initial values of acceleration.

3.4.23 Subroutine INCRS

This subroutine is called by the unit DYNAMC to set up Wilson's o-o-O method and updates the load vectors for successive cycles.

3.4.24 Subroutine MATLIN

This subroutine is called by ACNTRL to read in all material data and set up strain-stress relations for isothermal material.

3.4.25 Subroutine MATRLD

This subroutine is called by MATLIN for setting up material properties for stress analysis problem.

3.4.26 Subroutine MELAW

This subroutine is called by MATLIN for formulating stress strain relationships for isothermal element.

3.4.27 Subroutine PCOMP

This logical function subprogram is called by ACNTRL, GENVEC, PRTDIS to check for overflow during computations.

3.4.28 Subroutine PLOAD

This subroutine called by ACNTRL formulates the load vector.

3.4.29 Subroutine PROFIL

This unit is called by ACNTRL to compute the profile for the global arrays so that they could be stored into compact form. The equation numbers are also set up.

3.4.30 Subroutine PRTDIS

This subroutine is called by ACNTRL and CONSOL to print out the nodal values of the unknown quantities for each problem.

3.4.31 Subroutine PVALUE

This subroutine, called by ELMT01 and ELMT03 to calculate the values of principal strains and stresses.

3.4.32 Subroutine SETMPT

This subroutine, called by ACNTRL, checks the storage capacity.

3.4.33 Subroutines SHAPE2, SHAPE22, SHAPEX

These subroutines are meant for setting up the shape functions for various elements, calculating the derivatives and transformation to natural coordinates.

3.4.34 Subroutines TMINC

This subroutine forms the part of the dynamic analysis problem and updates the load vectors for successive time increments.

3.4.35 Subroutine TMSTP

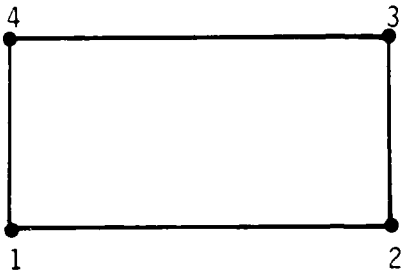
This subroutine does a part of setting up of stiffness matrices in consolidation problem.

SECTION IV

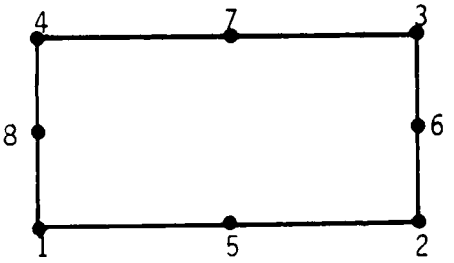
COMPUTER PROGRAM USAGE

4.1 PROGRAM CAPABILITY

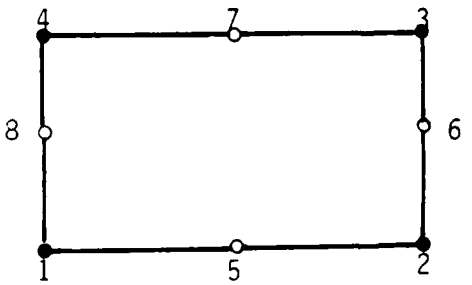
The code described in this report is a modified version of program "LAMP" originally developed by the author in the course of his doctoral research. The code was used in [5,7] to predict the quasi-static response and dynamic response of saturated elastic media to any boundary conditions or application of external forces. The code can handle mechanical as well as hydraulic anisotrophy, and fluid compressibility. The program allows for a choice of several element types. These include the 8-8, the 8-4 and 6-4 elements for quasi-static consolidation (Figure 5), and the 1-1, the 4-4 and the 8-8 elements for dynamic response analysis (Figure 6). Plane strain, plane stress as well as axisymmetric cases can be handled. Linear elastic as well as elasto-plastic behavior is allowed for. The program will solve one- and two-dimensional mass and heat transfer, convective transport problems as well as coupled deformation and mass transport. The example problems stated in [5,7] are a special case of this latter class.



The 6-4 Element



The 8-8 Element

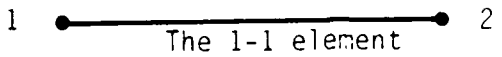


The 8-4 Element

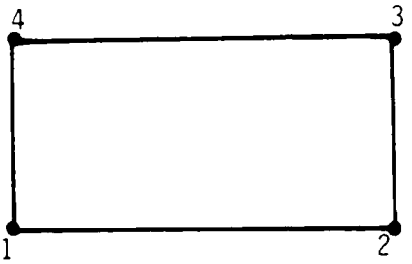
● DOF; $\underset{\sim}{u}$, π

o DOF; $\underset{\sim}{u}$

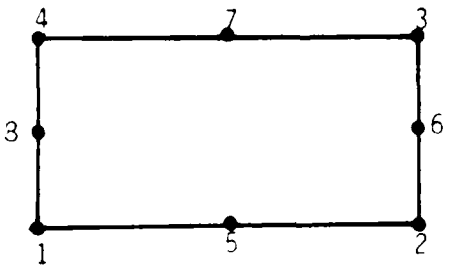
Figure 5: Element used for Quasi-static Consolidation of Soils



The 1-1 element



The 4-4 element



The 8-8 element

Figure 6: Element Used for Dynamic Response Analysis

4.2 PROGRAM LIMITATIONS

For analysis of soil consolidation, the program capability is limited to the case of small deformations, linear elastic soil bahavior, constant permeability and continued saturation. The program assumes the applied boundary conditions and loads constant throughout.

4.3 INSTRUCTIONS FOR USE

The program is basically composed of two modules, namely the data input module and the solution and output module. They are controlled by a set of easily recognized macro commands. This feature allows a user to program input data mode and the corresponding flow of solution process. Followings are the sequential order of input data cards in which distinctions are made between quasi-static consolidation and dynamic response analysis when needed.

4.3.1 Title Card (20A4)

Columns	Variable	Description
1-4	TITL (1)	Must contain LAMP to start the program.
5-80	(TITL (I), I=2,20)	Alpha numeric title of the problem.

4.3.2 Control Data Card-NAMELIST Input with the Name CONTROL

Variable	Description
NUMNP	Total number of nodal points

NUMEL	Total number of elements
NUMMAT	Number of different material sets
NEN	Number of nodes per element
NDM	Spatial dimension
NDF	Number of unknowns per nodes
NPN	Number of pressure nodes
NSTEPS	Number of time steps

Note 1; (a) If NAMELIST input is to be continued on to the next cards, a blank on the first column serves as a signal for continuation.

(b) For quasi-static consolidation and dynamic response analysis the following elements can be selected.

For Quasi-static Consolidation Analysis

The problem is solved only for plane strain state. The noal values are (u_x, u_y, π), i.e. NDF=3 and three types of elements can be used independently (Figure 5).

The 8-4 element in which NEN=8, NPN=4

The 8-8 element in which NEN=8, NPN=8

The 6-4 element in which NEN=4, NPN=4

For Dynamic Response Analysis

The problem is solved for one-dimensional and plane strain. For one-dimensional analysis (NDM=1), the 1-1 element where the linear interpolation is employed is used. The nodal values are (u_x, w_y), i.e. NDF=2, and NEN=2. For plane

strain analysis (NDM=2), the nodal values are (u_x, u_y, w_x, w_y) i.e. NDF=4, and two types of elements can be selected independently (Figure 6).

The 4-4 element which is defined dy NEN=4

The 8-8 element which is defined by NEN=8

Example; the below is an example of NAMELIST input for the Terzaghi's consolidation problem.

b&CONTROLbNUMNP=48,NUMEL=9,NUMMAT=1,NDF=3,NEN=8,NPN=4,NSTEPS=5

4.3.3 Node Geometry Data

4.3.3.1 Macro-Control Card

The macro instruction NODE(A4) which causes node input mode must be supplied at the first card.

4.3.3.2 Coordinate Data Card (2I5,6F10.0)

Repeat as many times as necessary after NODE card.

Columns	Variable	Description
1 - 5	N	Node number
6 - 10	NG	Generation increment (see Note 2)
11 -	(X(I,N),I=1,NDM)	I-th coordinate value of node N
	(FACTOR(I),I=1,NDM)	Spacing increment factor (see Note 2)

Terminate with a blank card.

Note 2: Nodal coordinates can be generated based on input values on two successive cards. The node number is computed sequentially (i.e.

N,N+NG,N+2NG,---. etc.) until the value of N on the second card is reached. NG=0 or blank causes no generation. Nodal spacing in I-th coordinate of generated nodes will be increased (decreased) successively by the FACTOR(I) prescribed on the first card of the pair if positive (negative). FACTOR(I)=0 will result in equal spacing in I-direction. Nodal number N needs not to be in in order. NG can be negative.

4.3.4 Element Data

4.3.4.1 Macro Control Card

The macro instruction ELEM(A4) at the first card causes element connectivity input mode.

4.3.4.2 Element Data Card (3I5,NENI5)

Repeat as many times as necessary after ELEM card.

Columns	Variable	Description
1-5	L	Element number
6-10	LK--IX(NEN1,L)	Material set number (<NUMMAT)
11-15	LX	Generation increment of node
16-	(IX(J,L),J=1,NEN)	J-th nodal number

Note 3: Elements must be in order and the nodal numbers must be read in counter clockwise. If element cards are omitted, the element data will be generated from the previous element with the same material number and all nodes are incremented by LX on the previous element. Generation up to the NUmel occurs when a blank

card is encountered. No blank card is needed if the last data is given.

Note 4: The consolidation elements in Note 1 can be generated by the common shape function routines, SHAPE2, SHAPE22 and SHAPEX according to number of nodes/elements.

4.3.5 Boundary Restraint Code Data

4.3.5.1 Macro Control Card

The macro instruction BOUN(A4) at the first card causes boundary restraint data input mode.

4.3.5.2 Boundary restrain code data cards (6I5)

Repeat as many times as necessary after BOUN card.

Columns	Variable	Description
1-5	N	Node number
6-10	NX	Generation increment of node
11-10+5*NDF	(ID(I), I=1,NDF)	Restraint code in I-direction

Terminate with a blank card.

Note 5: Nodal numbers are incremented by NX up to, but not including , N on the next card. NX=0 causes no generation. ID(I)=1 if I-th degree of freedom is prescribed, ID(I)=0 if not. Only nonzero codes need to be specified.

4.3.6 Nodal Boundary Value Data

4.3.6.1 Macro Control Card

The macro instruction BVAL(A4) at the first card causes boundary data input mode.

4.3.6.2 Macro Nonzero Boundary Data Card (2I5,7F10.0)

Repeat as many times as there are boundary values.

Columns	Variable	Description
1-5	N	node number
6-10	NG	generation increment
11-20+10*NDF	(F(I,N),I=1,NDF)	DOF(I) force (displ.)
	(FACTOR(I),I=1,NDF)	increment factor for I-force (displ.)

Terminate with a blank card. Boundary data generation algorithm is identical to the node geometry generation (see Note 2).

4.3.7 Material and Element Property Card

4.3.7.1 Macro Control Card

The macro instruction MATE(A4) AT THE FIRST CARD causes material data input mode.

4.3.7.2 Material Set Heading Card (20A4)

Alpha-numeric heading indicating the material, element type and so on is given. One card must be provided for each material set.

4.3.7.3 Identification Cards

NAMELIST input with the name MATID at commencement.

Variable	Description
MATNO	Material set number
NGAUSS	Number of Gauss integartion points (1 to 4). Need not to be provided if recommended default value 2 is to be used.
NSTRES	Number of stress points (1 to 4). Need not to be provided if recommended defeult value of 1 is used.
LTYPE	Element type for this material set; 9 for plane strain consolidation, 10 for one-dimensional dynamic analysis, 11 for dynamic analysis of plane strain case.

Example of identification card;

```
bATIDb MATNO=1,NGAUSS=1,LTYPE=10
```

4.3.7.4 Material Property Card

NAMELIST input with the name MATRL(A7) at commencement.

(a) Material properties required for quasi-static consolidation

Variable	Description
E1	Modulus of elasticity 1-axis.
E2	2-axis.
XNU1	Poisson's ratio in 1- and 2-axis.
XNU2	1- and 3-axis.
G2	Shear modulus (to be computed if not given).
RHO	Mixture mass density.
PHI	Reciprocal of fluid compressibility.

K1	Permeability in 1-direction.
K2	2-direction.
C1	Fluid mass density.

(b) Material properties required for dynamic analysis

Variable	Description
E1	Modulus of elasticity 1-axis
E2	2-axis
XNU1	Poisson's ratio 1- and 2-axis
XNU2	1- and 3-axis
G2	Shear modulus (to be computed if not given)
RHO	Mixture mass density (ρ)
PHI	Porosity
K1	Permeability in 1-direction
K2	2-direction
C1	Fluid mass density (ρ_2)
ALPA1	Fluid compressibility
ALPA2	Solid compressibility
SC	Stiffner damping coefficient for solid
ST	Mass damping coefficient for solid

4.3.8 Macro Commands for Solution Phase

For solution process, the following cards are needed.

(a) For Quasi-static Consolidation Analysis

Col. 1-4	Col. 6-9	Description
TANG	BAN	Symmetric tangent matrix

FORM*	BFO	Form body force vector
CONS***	SAVET**	Perform consolidation analysis
STOP		Terminate the run

* If body force is not considered, do not include this card. However, if this option is used, 1- and 2-components of the gravitational constant with correct signs must be provided in (2F10.0) format immediately after 'FORM BFO' card.

** If this option is used, the output will be saved on NTAPE (NTAPE=11 for displacement, 12 for stresses).

*** This card must be followed by the following cards.

1st card (3F10.0)

Columns	Variable	Description
1-10	Uo	Initial horizontal displacement
11-20	Vo	Initial vertical displacement
21-30	Po	Initial pore pressure

2nd Card Set (F10.0,4I5)

Repeat as many NSTEPS as defined in C.4.2.

1-10	(TIMES(I),I=1,NSTEPS)	Time at the end of a step
11-15	(NDT(I), I=1,NSTEPS)	Number of time increments/step
16-20	(NOPT(I),I=1,NSTEPS)	Time domain integration factor
21-25	(NPRINT(I),I=1,NSTEPS)	Printout parameter
26-30	(KPRS(I), I=1,NSTEPS)	=0/1 do not print/print stress

where NOPT = 1 for α=0.5

= 2 for α=0.67

= 3 for α=0.871

= 4 for α=1.0

= 5 for α=1+1/Δt -1/ln(1+Δt)

(b) Dynamic Response Analysis

Col. 1-4	Col.6-9	Description
TANG	BAN	Symmetric tangent matrix
FORM*	BFO	Form body force vector
DYNM***	SAVET**	Perform dynamic response analysis
STOP		Terminate the run

* Same as in Part a

** Same as in Part a

*** This card must be followed by the following cards

1st Card (4F10.0)

Col.	Variable	Description
1-10	USO	Initial solid displacement
11-20	VSO	Initial solid velocity
21-30	UFO	Initial fluid relative displacement
31-40	VFO	Initial fluid relative velocity

2nd Card (I5,4F10.0)

The dynamic load is input by the coefficients of loading function defined as

P(t) = AZERO + BZERO*COS(OMEGA*T) + CZERO*SIN(OMEGA*T)

Col.	Variable	Description
1-5	LOAD	Loading criterion+

6-15	OMEGA	Loading Frequency (radius/sec)
16-25	AZERO	Coefficient
26-35	BZERO	Coefficient
36-45	CZERO	Coefficient

+ LOAD = 0 ; Loading by traction only

= 1 ; Loading by support accelleration only

= 2 ; Loading by traction and acceleration

If LOAD = 0, skip the following two cards and go to 5th card.

3rd Card (4I5)

Col.	Variable	Description
1-5	LACEL	Acceleration with time ++=0/1
6- 5+NDF*5	IACEL	=0/1 for which degree of freedom has acceleration

++ LACEL=0 Use the loading function in Eq.(B-1)

LACEL=1 Use the Accelegram recorded data

If LACEL =0, skip the following card and go to 5th card.

4th card (2I5)

Col.	Variable	Description
1-5	NTAPE	The tape No. on which accelegram data is given
6-10	NDTACL	Number of acceleration records

Then, the following data are read from the tape;

(TRECORD(I),ACEL(I),I=1,NDTACL)

where TRECORD is the time and ACEL is the acceleration value.

5th Card (4F10.0,3I5)

Repeat as many as NSTEPS given in Control Data Card.

Col.	Variable	Description
1-10	(Times(I),I=1,NSTEPS)	The time at the end of a step
11-20	(θ(I),I=1,NSTEPS)	Wilson's coefficient
21-30	(β(I),I=1,NSTEPS)	Newmark's coefficient
31-40	(γ(I),I=1,NSTEPS)	Newmark's coefficient
41-45	(NDT(I),I=1,NSTEPS)	Number of increments/steps
46-50	(NPRINT(I),I+1,NSTEPS)	=J; Print Jth increment
51-55	(KPRS(I),I=1,NSTEPS)	=0/1 Print/do not print stresses

REFERENCES

1. Ghaboussi, J., and Wilson, E. L.: "Flow of Compressible Fluid in Porous Elastic Media", Int. J. Num. Methods in Engrg., 5, pp.419-442, 1973.

2. Sandhu, R. S.: Fluid Flow in Saturated Porous Elastic Media, Ph.D. Thesis, University of California at Berkeley, Berkeley, California, 1968.

3. Sandhu, R. S., Liu, H.,and Sing, K. J.: "Numerical Performance of Some Finite Element Schemes for Analysis of Seepage in Porous Elastic Media", Int. J. Num. Anal. Methods in Geomech., 1, pp.177-194, 1977.

4. Sandhu, R. S., and Liu, H.: "Analysis of Consolidation of Viscoelastic Soils", in Numerical Methods in Geomechanics, Proc., 3rd. Int. Conf. Numer. Methods in Geomech., Ed. W. Wittke, A.A Balkema, pp.1255-1263, 1979.

5. Aboustit, B. L., and Sandlu, R. S.: An Evaluation of Finite Element Models for Soil Consolidation, Report OSURF-715107-84-2 to Air Force Office of Scientific Research, Dept. of Civil Engineering, The Ohio State University, Columbus, Ohio, 184.

6. Gibson, R. E., Schiffman, R. L., and Pu., S. L.: "Plane Strain and Axially Symmetric Problems of Consolidation of a Semi-infinite Stratum", Quar. J. Mech. Appl. Math., 16, pp.34-50, 1970.

7. Sandhu, R. S., Aboustit, B. L., and Hong, S. J.: Response of Saturated Soils to Dynamic Loading, Report OSURF-715107-84-4 to Air Force Office of Scientific Research, Dept. of Civil Engineering, The Ohio State University, Columbus, Ohio, 184.

8. Boit, M. A.: "Mechanics of Deformation and Acoustic Propagation in Porous Media", J. Appl. Phys., 33, No.4, 1483-1498, 1961.

Appendix A

PROGRAM LISTING

The complete listing of the computer program is given below.

```
C     ....PROGRAMMED BY PROF. J. K. LEE, B.L. ABOUSTIT
C     DEPT OF ENGR. MECH.OHIO STATE UNIVERSITY, COLUMBUS, OHIO.
C     ....MAIN PROGRAM : SET PROGRAM CAPACITY.
C     ---------------------------------------
C     ....PROBLEM SIZE IS CONTROLLED BY THE DIMENSION IN BLANK COMMON AN
C     THE VALUE OF MAX.
C     ....ALL ARRAYS RESIDE IN CENTRAL MEMORY.
C     ....DOUBLE PRECISION IS ASSUMED. FOR SINGLE PRECISION SET IPR = 1
C     AND SUPPRESS ALL IMPLICIT REAL STATEMENTS.
C              CALLED BY -
C              CALLS     - ACNTRL
C
C     .... FILE ASSIGNMENT  :
C      FILE NO. 10 = MESH DATA IF NODE SAVET, ETC.
C      FILE NO. 11 = SOLUTION (DISP) IF DISP SAVET.
C      FILE NO. 12 = STRESSES IF STRE SAVET
C      FILE NO. 13 = SCRATCH FILE IF NTEMP .GE. 2.
C      FILE NO. 14 = SCRATCH FILE FOR STRAINS
C      FILE NO. 15 = SCRATCH FILE FOR ELEMENT STIFFNESS
C      FILE NO. 16 = SCRATCH FILE FOR GLOBAL STIFFNESS
C      FILE NO. 17 = SCRATCH FILE FOR GLOBAL FORCE VECTOR
C      FILES 10 TO 12 USE FORMATTED I/O ( 5 FIELD INTEGERS AND 12 FIELD
C      REALS ).
C
      COMMON M(280000)
      COMMON /SIZE/ MAX
      MAX=280000
      IPR=2
C
      CALL ACNTRL (IPR)
C
      STOP
      END
```

```
      BLOCK DATA
C     ---------------
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      COMMON /LABL/ PD(9),A(9),BC(2),DI(6),CD(3),TE(3),FD(3),FOL(3),FON(
     14)
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,TWR
      COMMON /MACRO/ WD(15),NWD,NV,NC,NRD
      COMMON /MLIST/ IEL,NTEMP,NGAUSS,NSTRES,DUMY(23)
      COMMON /ELABL/ ITYPE(6,10)
      DATA FOL/3H(I5,2H  ,4HI5) /,FON/3H(I5,2H  ,4HF12.,2H4)/
      DATA ANG/57.29578,6.2831853,32.174,9.81/,ST/3HESS,3HAIN/
      DATA A/2H,1,2H,2,2H,3,2H,4,2H,5,2H,6,2H,7,2H,8,2H,9/,CD/4H COO,4HR
     1DIN,4HATES/
      DATA TE/4H TEM,4HPERA,4HTURE/,FD/4H BOU,4HND V,4HALUE/
      DATA BC/4H B.C,2H. /,DI/4H DIS,2HPL,4H VEL,2HOC,4H ACC,2HEL/
      DATA O/1HO/
      DATA PD/6H(I4,2X,2H  ,2HI1,2H  ,8HD12.5   ,2H  ,8HD12.5,2X,2H  ,6H
     1D12.5)/
      DATA GPT/4*0.D0,-.57735026919896D0,.57735026919896D0,2*0.D0,-.774596
     16692415D0,0.D0,.7745966692415D0,0.D0,-.86113631159410D0,-.339981043
     25849D0,.33998104358490D0,.86113631159410D0/
      DATA WT/2.D0,3*0.D0,2*1.D0,2*0.D0,.55555555555556D0,.8888888888889D
     10,.5555555555556D0,0.D0,.34785484513750D0,.65214515486250D0,.6521451
     2548625D0,.34785484513750D0/
      DATA WD/4HNODE,4HELEM,4HBOUN,4HMATE,4HBVAL,4HTEMP,4HUTAN,4HTANG,4H
     1FORM,4HSOLV,4HDISP,4HSTRE,4HCONS,4HDYNM,4HSTOP/,NWD/15/,NV/1/,NC/1
     2/,NRD/26/
      DATA LX/-1,1,1,-1/,LE/-1,-1,1,1/
      DATA NTEMP/1/,NGAUSS/2/,NSTRES/1/,DUMY/23*0.0D0/
      DATA ITYPE/4HPLAN,4HE ST,4HRESS,3*4H    ,4HPLAN,4HE ST,4HRAIN,3*4H
     1    ,4HAXIS,4HYMME,4HTRIC,4H STR,4HESS ,4H    ,4H1-D ,4HCART,4HESI
     2A,4HN TR,4HANSP,4HORT ,4H2-D ,4HCART,4HESIA,4HN TR,4HANSP,4HORT ,4
     3H1-D ,4HAXIS,4HYM. ,4HTRAN,4HSPOR,4HT   ,4H2-D ,4HAXIS,4HYM. ,4HTR
     4AN,4HSPOR,4HT   ,4H2-D ,4HADVE,4HCTIO,4HN-DI,4HFUSS,4HION ,4HP.ST,
     54HRAIN,4H-CON,4HSOLI,4HDATI,4HON  ,6*4H    /
      END
```

```
      SUBROUTINE ACNTRL (IPR)
C     ----------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NP
C
C     ....MAIN CONTROL ROUTINE
C     ....READ HEADING AND THE CONTROL DATA.
C     ....SET STORAGE POINTERS
C
C              CALLED BY - MAIN
C              CALLS     - PCOMP,ZERO,GENVEC,CPTABL,BCCODE,PROFIL,
C                          SETMPT,ELDATA,ELMLIB,PLOAD,FORMEL,ACTCOL
C                          UACTCL,PRTDIS,CONSOL
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /LABL/ PD(9),A(9),BC(2),DI(6),CD(3),TE(3),FD(3),FOL(3),FON(
     14)
      COMMON /PRLOD/ PROP
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,TWR
      COMMON /TIMECK/ CPTIME(20),COMPUT(20),NCHK
      COMMON /MACRO/ WD(15),NWD,NV,NC,NRD
      NAMELIST /CONTRL/ NUMNP,NUMEL,NUMMAT,NEN,NDM,NDF,NPN,NSTEPS
      LOGICAL PCOMP,ERR,PRT,UNSYM,AFAC,BACK,AFL,BFL,CFL,TRD,TWR,TLD,BND
      COMMON M(1)
      DIMENSION TITL(20), SAVEFL(10), NFILE(10)
      DATA START/4HLAMP/,COMNT/4HC.../,TL/4H TLO/,BF/4H BFO/,NP/4H NOP/,
     1RD/4H REA/,WR/4H SAV/,PR/4H PRI/,BN/4H BAN/
C     ....FILE NO. 10 : MESH DATA
C     ....FILE NO. 11 : SOLUTION ( IN SUB PRTDIS )
C     ....FILE NO. 12 : P.STRESSES ( IN SUB PVALUE )
      ERR=.FALSE.
      PROP=1.0D0
C     ....READ AND WRITE INPUT DATA AS GIVEN
      LINE=0
      WRITE (6,38)
 1    READ (5,34,END=2) TITL
      LINE=LINE+1
      WRITE (6,39) LINE,TITL
      GO TO 1
 2    REWIND 5
      NSAVE=0
 3    IF(ERR) STOP
      READ (5,34) TITL
      WRITE (6,40) TITL
      IF(PCOMP(TITL(1),COMNT)) GO TO 3
      DO 4 IM=1,NWD
      IF(PCOMP(TITL(1),WD(IM))) GO TO 6
 4    CONTINUE
      IF(.NOT.PCOMP(TITL(1),START)) GO TO 31
      DO 5 I=1,20
 5    HEAD(I)=TITL(I)
      IF(NCHK.NE.0) CALL CPTABL (TITL(1),1)
      CALL ZERO (CPTIME,20)
```

```
      NCHK=0
C
      READ (5,CONTRL)
      WRITE (6,37) HEAD,NUMNP,NUMEL,NUMMAT,NEN,NDM,NDF
C     ..................................................................
C     NUMNP            NUMBER OF NODAL POINTS
C     NUMEL            NUMBER OF ELEMENTS
C     NEN              MAX. NUMBER OF NODES PER ELEMENT
C     NDM              NUMBER OF DIMENSION
C     NDF              NO. OF DEGREE OF FREEDOM PER NODE
C     ..................................................................
      MEQ=NUMNP*NDF
      NST=NDF*NEN
      NEN1=NEN+1
      MEQR=MEQ*IPR
      AFAC=.TRUE.
      BACK=.TRUE.
      TLD=.FALSE.
C
C     ....SET STORAGE POINTERS FOR ALL ARRAYS EXCEPT FOR THE GLOBAL STIF
C     ....M'S ARE FOR INTEGER ARRAYS AND N'S ARE FOR REAL ARRAYS.
C
      NO=1
      NT=NO+NUMNP*NDM*IPR
      MO=NT+NUMNP*IPR
      M1=MO+NEN1*NUMEL
      M2=M1+NUMMAT*4
      M3=M2+NDF*NUMNP
      M4=M3+NST
      M5=M4+NEN
      LAST=M5
      CALL SETMPT (N1,M5,MEQ,LAST,TITL(1))
      N2=N1+NDM*NEN*IPR
      N3=N2+NRD*IPR
      N4=N3+NST*NST*IPR
      N5=N4+NST*IPR
      N6=N5+NST*IPR
      N7=N6+NEN*IPR
      N8=N7+MEQR
      N9=N8+MEQR
      N10=N9+MEQR
      CALL ZERO (M(N7),MEQ*3)
      GO TO 3
C     CHECK MACRO COMMAND CARD AND TRANSFER TO APPROPRIATE PROCEDURE
 6    TRD=PCOMP(TITL(2),RD)
      TWR=PCOMP(TITL(2),WR)
      PRT=.TRUE.
      IF(IM.LE.7) NTAPE=10
      IF(IM.EQ.11.OR.IM.EQ.13) NTAPE=11
      IF(IM.EQ.12) NTAPE=12
      IF(PCOMP(TITL(2),NP)) PRT=.FALSE.
      IF(.NOT.TRD) GO TO 8
```

```
      IF(NCHK.NE.0) GO TO 7
      READ (NTAPE,36) TITL,NODES,NELS,NUMS,NENS,NDMS,NDFS
      IF(NODES.EQ.NUMNP.AND.NELS.EQ.NUMEL.AND.NENS.EQ.NEN.AND.NDMS.EQ.ND
     1M.AND.NDFS.EQ.NDF) GO TO 7
      WRITE (6,46) NTAPE,NODES,NELS,NUMS,NENS,NDMS,NDFS
      STOP
 7    READ (NTAPE,34) TITL(1)
      IF(PCOMP(TITL(1),WD(IM))) GO TO 9
      WRITE (6,47) TITL,WD(IM),NTAPE
      STOP
 8    IF(.NOT.TWR) GO TO 9
      IF(NCHK.EQ.0) WRITE (NTAPE,36) HEAD,NUMNP,NUMEL,NUMMAT,NEN,NDM,NDF
      WRITE (NTAPE,34) TITL(1)
      NSAVE=NSAVE+1
      SAVEFL(NSAVE)=TITL(1)
      NFILE(NSAVE)=NTAPE
 9    CALL SCLOK1
      GO TO (10,11,12,13,14,15,16,17,22,24,27,28,29,30,33), IM
C     .............READ AND /OR GENERATE NODAL GEOMETRY DATA.............
C             TITL(1) = NODE
 10   CONTINUE
      CALL GENVEC (NDM,M(NO),CD,PRT,ERR,TRD,TWR)
      GO TO 32
C     .............READ AND/OR GENERATE ELEMENT DATA.............
C           TITL(1) = ELEM
 11   CONTINUE
      CALL ELDATA (M(MO),M(M4),NEN1,NDF,PRT,ERR,TRD,TWR,MDIF)
      GO TO 32
C     READ AND/OR GENERATE BOUNDARY CONSTRAINT CODES.
C      TITL(1) = BOUN
 12   CALL BCCODE (M(M2),M(M3),NDF,NUMNP,PRT,TRD,TWR,HEAD,NTAPE)
      GO TO 32
C
C     ....READ THE MATERIAL DATA.
C
 13   WRITE (6,41) HEAD
      NOW=0
      CALL MATLIN (M(N10),NADD,M(M1),NUMMAT,NRD,M(N2))
      LAST=N10
      CALL SETMPT (N11,N10,NADD*IPR,LAST,TITL(1))
      GO TO 3
C
C     .... GET NODAL BOUNDARY VALUES ( NATURAL AND/OR ESSENTIAL ).
C
 14   CALL GENVEC (NDF,M(N8),FD,PRT,ERR,TRD,TWR)
      GO TO 3
C
C
 15   CALL GENVEC (1,M(NT),TE,PRT,ERR,TRD,TWR)
      GO TO 3
C
C     .... FORM TANGENT MATRIX
```

```
C
 16   UNSYM=.TRUE.
 17   IF(IM.EQ.8) UNSYM=.FALSE.
      CFL=UNSYM
      AFL=.TRUE.
      BFL=.TRUE.
      AFAC=.TRUE.
      BND=PCOMP(TITL(2),BN)
      IF(.NOT.BND) GO TO 18
      IBAND=(MDIF+1)*NDF
      WRITE (6,50) IBAND
      NCOEF=MEQ*IBAND
      GO TO 19
 18   CALL PROFIL (M(M5),M(M2),M(MO),NDF,NEN1,NCOEF,TRD,TWR)
 19   CALL SETMPT (N12,N11,NCOEF*IPR,LAST,TITL(1))
      NA=N11
      N13=N12
      CALL ZERO (M(NA),NCOEF)
      IF(.NOT.UNSYM) GO TO 20
      CALL SETMPT (N13,N12,NCOEF*IPR,LAST,TITL(1))
      NC=N12
      CALL ZERO (M(NC),NCOEF)
 20   WRITE (6,44) NO,NT,MO,M1,M2,M3,M4,M5,N1,N2,N3,N4,N5,N6,N7,N8,N9,N1
     10,NA,NC,N13
      WRITE (6,45) MEQ,NEQ,NCOEF,LAST,NRD
      ISW=2
 21   NOW=NOW+1
      PRT=PCOMP(TITL(3),PR)
      CALL FORMEL (M(NO),M(N1),M(N10),M(N2),M(N3),M(N4),M(N5),M(N6),M(N7
     1),M(N8),M(NT),M(MO),M(M1),M(M2),M(M3),M(M4),M(M5),NDF,NDM,NEN1,NST
     2,ISW,M(NA),M(NC),M(N9),AFL,BFL,CFL,PRT,TLD,NRD,NOW,IBAND,BND)
      IF(ISW.NE.5) GO TO 32
      TLD=.FALSE.
      CALL ZERO (M(N7),MEQ)
      GO TO 32
C
C     .... FORM CONSISTANT LOADING DUE TO TEMPERATURE OR/AND BODY FORCE.
C
 22   ISW=0
      IF(.NOT.PCOMP(TITL(2),TL)) GO TO 23
      ISW=3
      TLD=.TRUE.
 23   IF(PCOMP(TITL(2),BF)) ISW=4
      IF(ISW.EQ.0) GO TO 3
      IF(ISW.EQ.4) READ (5,35) (DC(I),I=1,NDM)
C
C     .... DC(I) = I COMPONENT OF THE GRAVITATIONAL CONSTANT G.
C
      AFL=.FALSE.
      CFL=.FALSE.
      BFL=.TRUE.
      PRT=.FALSE.
```

```
      GO TO 21
C
C     .... SOLVE THE EQUATIONS.
C
 24   CALL PLOAD (M(M2),M(N8),M(N7),M(N9),M(NT),MEQ,1.DO,0,BND)
      IF(UNSYM) GO TO 25
      IF(BND) CALL BANSOL (M(NA),M(N7),M(N7),MEQ,IBAND,AFAC,BACK)
      IF(.NOT.BND) CALL ACTCOL (M(NA),M(N7),M(M5),NEQ,AFAC,BACK)
      GO TO 26
 25   CONTINUE
 26   AFAC=.FALSE.
      IF(PCOMP(TITL(2),TE(1))) CALL PLOAD (M(M2),M(N8),M(N7),M(N9),M(NT)
     1,MEQ,1.0DO,1,BND)
      CALL ZERO (M(N9),MEQ)
      GO TO 32
C
C     .... PRINT THE SOLUTION VECTOR
C
 27   CALL PRTDIS (M(M2),M(NO),M(N7),M(N8),NDM,NDF,TRD,TWR,PRT,0,0,0.DO,
     1BND)
      GO TO 3
C
C     .... COMPUTE AND PRINT GRADIENT OF SOLUTION ( OR STRESS )
C      AT GAUSS PTS.
C
 28   ISW=5
      AFL=.FALSE.
      BFL=.FALSE.
      CFL=.FALSE.
      LINE=0
      GO TO 21
C
C     .... CONSOLIDATION ANALYSIS .... TITL(1) = CONS ....
C
 29   CONTINUE
C
C     M(N13) = TIMES(1)
C     M(M6)  = NDT(1), M(M7) = NPRINT(1), M(M8) = NOPT(1)
C
      M6=N13+NSTEPS*IPR
      M7=M6+NSTEPS
      M8=M7+NSTEPS
      CALL SETMPT (N14,M8,NSTEPS,LAST,TITL(1))
      WRITE (6,48) N13,M6,M7,M8,N14,LAST
C
      CALL CONSOL (M(NO),M(NT),M(MO),M(M1),M(M2),M(M3),M(M4),M(M5),M(N1)
     1,N(N2),M(N3),M(N4),M(N5),M(N6),M(N7),M(N8),M(N9),M(N10),M(NA),M(N1
     23),M(M6),M(M7),M(M8),NEN,NEN1,NPN,NDF,NDM,MEQ,IBAND,NUMEL,NUMNP,NS
     3TEPS,NST,NRD,BND)
C
      GO TO 32
C
```

```
C      .... DYNAMIC CONSOLIDATION ANALYSIS
C
 30    CONTINUE
C      M(N13) = TIMES(1), M(M6) = NDT(1), M(M7)= NPRINT(1)
C      M(N14) = THETA(1), M(N15) = GAMA(1), M(N16) = BETA(1)
C      M(N17) = VO(NDF,1), M(N18) = AO(NDF,1)
C      M(N19) = U(NDF,1), M(N20) = V(NDF,1),M(N21) = A(NDF,1)
C
C
       M6=N13+NSTEPS*IPR
       M7=M6+NSTEPS
       LAST=M7
       CALL SETMPT (N14,M7,NSTEPS,LAST,TITL(1))
       N15=N14+NSTEPS*IPR
       N16=N15+NSTEPS*IPR
       N17=N16+NSTEPS*IPR
       N18=N17+NDF*NUMNP*IPR
       N19=N18+NDF*NUMNP*IPR
       N20=N19+NDF*NUMNP*IPR
       N21=N20+NDF*NUMNP*IPR
       LAST=N21
       CALL SETMPT (N22,N21,NDF*NUMNP*IPR,LAST,TITL(1))
       PRINT 49, N13,M6,M7,N14,N15,N16,N17,N18,N19,N20,N21,N22,LAST
C
       CALL DYNAMC (M(NO),M(NT),M(MO),M(M1),M(M2),M(M3),M(M4),M(M5),M(N1)
      1,M(N2),M(N3),M(N4),M(N5),M(N6),M(N7),M(N8),M(N9),M(N10),M(NA),M(N1
      23),M(M6),M(M7),M(N14),M(N15),M(N16),M(N17),M(N18),M(N19),M(N20),M(
      3N21),NEN,NEN1,NDF,NDM,MEQ,IBAND,NUMEL,NUMNP,NSTEPS,NST,NRD,BND)
       GO TO 32
 31    WRITE (6,42) TITL
       STOP
C
 32    CALL CPTABL (TI  '1),0)
       GO TO 3
 33    CONTINUE
       CALL CPTABL (TITL(1),1)
       IF(NSAVE.NE.0) WRITE (6,43) (N,NFILE(N),SAVEFL(N),N=1,NSAVE)
C
       RETURN
C
 34    FORMAT (20A4)
 35    FORMAT (3F10.0)
 36    FORMAT (20A4,/,10I5)
 37    FORMAT (1H1,5X,20A4,//,5X,'NUMBER OF NODAL POINTS
      1        = ',I5,/,5X,'NUMBER OF ELEMENTS
      2  = ',I5,/,5X,'NUMBER OF MATERIAL SETS                              = ',
      3I5,/,5X,'MAXIMUM NUMBER OF NODES PER ELEMENT           = ',I5,/,5
      4X,'DIMENSION OF COORDINATE SPACE                  = ',I5,/,5X,'DEG
      5REE OF FREEDOM PER NODE                     = ',I5,//)
 38    FORMAT (1H1,//10X,'....LIST OF INPUT AS GIVEN....'//)
 39    FORMAT (5X,'CARD NO. =',I4,':',20A4)
 40    FORMAT (//5X,20A4)
```

```
41   FORMAT (1H1,5X,20A4)
42   FORMAT (///5X,'....FATAL ERROR....THE MACRO COMMAND'//5X,20A4//5X,
    1'IS NOT ALLOWED IN THIS PROGRAM.')
43   FORMAT (////10X,'D A T A  F I L E '//10X,'FILE NO.',5X,'MACRO'//10
    1(5X,2I5,5X,A4/))
44   FORMAT (///5X,'ARRAY POINTERS : NO,NT,M0-M5,N1-N10,NA,NC,N13'//5X,
    12(12I10//))
45   FORMAT (///5X,'NUMBER OF TOTAL EQUATIONS (MEQ) =',I8/5X,'NUMBER OF
    1 ACTIVE EQUATIONS(NEQ) =',I8/5X,'NUMBER OF COEFFICIENTS IN (A)   =
    2',I8/5X,'REQUIRED SIZE OF (M)            =',I8/5X,'NRD ( SET IN BL
    3OCK DATA )       =',I8)
46   FORMAT (///5X,'....FATAL ERROR.... CONTROL DATA FROM TAPE NO.',I5,
    1'  ARE :',//5X,'NUMBER OF NODAL POINTS                              = '
    2,I5,/,5X,'NUMBER OF ELEMENTS                                  = ',I5,/,
    35X,'NUMBER OF MATERIAL SETS                             = ',I5,/,5X,'MA
    4XIMUM NUMBER OF NODES PER ELEMENT                 = ',I5,/,5X,'DIMENSIO
    5N OF COORDINATE SPACE                       = ',I5,/,5X,'DEGREE OF FREE
    6DOM PER NODE                          = ',I5,/,5X,//,5X,'WHICH DO NOT M
    7ATCH WITH THE CONTROL DATA SPECIFIEDD      FOR THIS RUN..... EXC
    8UTION TERMINATED..... ')
47   FORMAT (///5X,'.....FATAL ERROR.. : MACRO MISMATCH.....CORRECT
    1  MACRO IS ',//5X,20A4,//5X,'INSTEAD OF *',A4,'* TO RETRIEVE
    2 DATA FROM THE TAPE NUMBER',I5,'....EXCUTION TERMINATED....')
48   FORMAT (///5X,'ADDITIONAL MEMORY POINTERS FOR THE CONSOLIDATION AN
    1ALYSIS'//5X,'TIMES(1) = M(N13)     :',I8/5X,'NDT(1) = M(M6)
    2 :',I8/5X,'NOPT(1) = M(M7)       :',I8/5X,'NPRINT(1) =M(M8)      :
    3',I8/5X,'TOTAL RESERVED M(N14) :',I8,5X,'LAST =',I8)
49   FORMAT (///5X,'ADDITIONAL MEMORY POINTERS FOR THE DYNAMIC ANA
    1 LYSIS'//5X,'TIMES(1) = M(N13)     :',I8/5X,'NDT(1) = M(M6)
    2 :',I8/5X,'NPRINT(1) =M(M7)      :',I8/5X,'THETA(1) = M(N14)     :
    3',I8/5X,'GAMA(1) = M(N15)      :',I8/5X,'BETA(1) = M(N16)      :',
    4I8/5X,'V0(1)   = M(N17)      :',I8/5X,'A0(1)   = M(N18)      :',I8
    5/5X,'U(1)    = M(N19)      :',I8/5X,'V(1)    = M(N20)      :',I8/5
    6X,'A(1)    = M(N21)      :',I8/5X,'TOTAL RESERVED M(N22) :',I8,5X,
    7'LAST =',I8)
50   FORMAT (//5X,'BAND SOLVER IS USED.....IBAND =',I8)
     END
```

```
      SUBROUTINE ACTCOL (A,B,JDIAG,NEQ,AFAC,BACK)
C     ------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL AFAC,BACK
      COMMON /ENGYS/ AENGY
      DIMENSION A(1), B(1), JDIAG(1)
C
C     .... ACTIVE COLUMN PROFILE SYMMETRIC EQUATION SOLVER
C
C               CALLED BY - ACNTRL
C               CALLS     - DOT
C
C     .... FACTOR A TO UT*D*U, REDUCE B
      AENGY=0.0D0
      JR=0
      DO 6 J=1,NEQ
      JD=JDIAG(J)
      JH=JD-JR
      IS=J-JH+2
      IF(JH-2) 6,3,1
 1    IF(.NOT.AFAC) GO TO 5
      IE=J-1
      K=JR+2
      ID=JDIAG(IS-1)
C     .... REDUCE ALL EQUATIONS EXCEPT DIAGONAL
      DO 2 I=IS,IE
      IR=ID
      ID=JDIAG(I)
      IH=MINO(ID-IR-1,I-IS+1)
      IF(IH.GT.0) A(K)=A(K)-DOT(A(K-IH),A(ID-IH),IH)
 2    K=K+1
C     .... REDUCE DIAGONAL TERM
 3    IF(.NOT.AFAC) GO TO 5
      IR=JR+1
      IE=JD-1
      K=J-JD
      DO 4 I=IR,IE
      ID=JDIAG(K+I)
      IF(A(ID).EQ.0.0D0) GO TO 4
      D=A(I)
      A(I)=A(I)/A(ID)
      A(JD)=A(JD)-D*A(I)
 4    CONTINUE
C     .... REDUCE RHS
 5    IF(BACK) B(J)=B(J)-DOT(A(JR+1),B(IS-1),JH-1)
 6    JR=JD
      IF(.NOT BACK) RETURN
C     .... DIVIDE BY DIAGONAL PIVOTS
      DO 7 I=1,NEQ
      ID=JDIAG(I)
      IF(A(ID).NE.0.0D0) B(I)=B(I)/A(ID)
 7    AENGY=AENGY+B(I)*B(I)*A(ID)
```

```
C       .... BACKSUBSTITUTE
        J=NEQ
        JD=JDIAG(J)
 8      D=B(J)
        J=J-1
        IF(J.LE.0) RETURN
        JR=JDIAG(J)
        IF(JD-JR.LE.1) GO TO 10
        IS=J-JD+JR+2
        K=JR-IS+1
        DO 9 I=IS,J
 9      B(I)=B(I)-A(I+K)*D
 10     JD=JR
        GO TO 8
        END
```

```
      SUBROUTINE ADDSTF (A,B,C,S,P,JDIAG,LD,NST,NEL,AFL,BFL,CFL)
C     ------------------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... ASSEMBLE GLOBAL ARRAYS
C
C               CALLED BY - FORMEL
C
      LOGICAL AFL,BFL,CFL
      DIMENSION A(1), B(1), JDIAG(1), P(1), S(NST,1), LD(1), C(1)
      DO 2 J=1,NEL
      K=LD(J)
      IF(K.EQ.0) GO TO 2
      IF(BFL) B(K)=B(K)+P(J)
      IF(.NOT.AFL.AND..NOT.CFL) GO TO 2
      L=JDIAG(K)-K
      DO 1 I=1,NEL
      M=LD(I)
      IF(M.GT.K.OR.M.EQ.0) GO TO 1
      M=L+M
      IF(AFL) A(M)=A(M)+S(I,J)
      IF(CFL) C(M)=C(M)+S(J,I)
 1    CONTINUE
 2    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BANADD (GK,MXR,MXC,GF,FE,EK,NK,NDFE,LD,STIF,FORC)
C     ------------------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL FORC,STIF
      DIMENSION GK(MXR,MXC), GF(1), FE(1), EK(NK,NK), LD(1)
C
C     THIS SUBROUTINE ASSEMBLES ELEMENT ARRAYS (EK) AND (FE) TO FORM
C     GLOBAL ARRAYS (GK) AND (GF), RESPECTIVELY. (EK) IS ASSUMED TO BE S
C
C     (GK) = THE UPPER HALF OF A SYMM. GLOBAL MATRIX STORED IN
C            RECTANGULAR ARRAY WITH DIAGONALS ON THE FIRST COLUMN.
C     MXR  = MAX. NO. OF ROWS OF (GK) AS DIMENSIONED IN THE MAIN.
C     MXC  = MAX. NO. OF COLS OF (GK) AS DIMENSIONED IN THE MAIN.
C     (EK) = SYMMETRIC ELEMENT MATRIX OF ACTUAL SIZE NDFE BY NDFE
C            SUPPLIED BY THE MAIN.
C
C
C     NDFE = DEGREE OF FREEDOM FOR THIS ELEMENT
C          = (NO. OF NODES) * (DEGREE OF FREEDOM) FOR THIS ELEMENT.
C
C     (LD) = ELEMENT ASSEMBLY POINTER ARRAY.
C
C     BOUNDARY CONDITIONS WILL BE IMPOSED ON LATER IN THE SUBROUTINE BNC
C
      DO 2 I=1,NDFE
      II=LD(I)
      IF(FORC) GF(II)=GF(II)+FE(I)
      IF(.NOT.STIF) GO TO 2
      DO 1 J=1,NDFE
      JJ=LD(J)
      IF(II.GT.JJ) GO TO 1
      K=JJ-II+1
      GK(II,K)=GK(II,K)+EK(I,J)
 1    CONTINUE
 2    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BANSOL (A,X,B,N,IB,BACK,FORW)
C     ----------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL BACK,FORW
      DIMENSION A(N,IB), X(1), B(1)
      IF(BACK) GO TO 3
C     ----     THIS SUBROUTINE TRIANGULARIZES A BANDED AND SYMMETRIC
C     COEFFICIENT MATRIX (A).
C     -----    ONLY THE UPPER HALF BAND PORTION OF THE COEFFICIENT MATRI
C     IS STORED AS A RECTANGULAR ARRAY
      DO 2 I=2,N
      M1=MINO(IB-1,N-I+1)
      DO 2 J=1,M1
      SUM=0.0
      K1=MINO(I-1,IB-J)
      DO 1 K=1,K1
    1 SUM=SUM+A(I-K,K+1)*A(I-K,J+K)/A(I-K,1)
    2 A(I,J)=A(I,J)-SUM
C     -----    THIS SUBROUTINE MULTIPLIES THE INVERSE OF LEFT TRIANGULAR
C     FORM WITH THE RIGHT HAND SIDE VECTOR, AND THEN SOLVES FOR THE
C     UNKNOWNS BY BACK SUBSTITUTION PROCESS
      IF(FORW) RETURN
    3 NP1=N+1
      DO 5 I=2,N
      SUM=0.0
      K1=MINO(IB-1,I-1)
      DO 4 K=1,K1
    4 SUM=SUM+A(I-K,K+1)/A(I-K,1)*B(I-K)
    5 B(I)=B(I)-SUM
C     -----    BEGIN BACK SUBSTITUTION
      X(N)=B(N)/A(N,1)
      DO 7 K=2,N
      I=NP1-K
      J1=I+1
      J2=MINO(N,I+IB-1)
      SUM=0.0
      DO 6 J=J1,J2
      MM=J-J1+2
    6 SUM=SUM+X(J)*A(I,MM)
    7 X(I)=(B(I)-SUM)/A(I,1)
      RETURN
      END
```

```
      SUBROUTINE BCCODE (ID,IDL,NDF,NUMNP,PRT,TRD,TWR,HEAD,NTAPE)
C     ------------------------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL PRT,TRD,TWR
      DIMENSION ID(NDF,1), IDL(1), HEAD(20)
      COMMON /LABL/ PD(9),A(9),BC(2),DUM(15),FOL(3),FON(4)
C
C               CALLS     -
C
      FOL(2)=A(NDF+1)
      IF(PRT) WRITE (6,12) HEAD,(I,BC,I=1,NDF)
      IF(TRD) READ (NTAPE,FOL) (N,NG,(ID(I,N),I=1,NDF),N=1,NUMNP)
      IF(TRD) GO TO 6
      DO 1 I=1,NDF
      DO 1 J=1,NUMNP
    1 ID(I,J)=0
      N=0
      NG=0
    2 L=N
      LG=NG
      READ (5,10) N,NG,(IDL(I),I=1,NDF)
      IF(N.LE.0.OR.N.GT.NUMNP) GO TO 6
      DO 3 I=1,NDF
    3 ID(I,N)=IDL(I)
      LG=ISIGN(LG,N-L)
    4 L=L+LG
      IF((N-L)*LG.LE.0) GO TO 2
      DO 5 I=1,NDF
    5 ID(I,L)=ID(I,L-LG)
      GO TO 4
    6 CONTINUE
      IF(.NOT.PRT) RETURN
      DO 9 N=1,NUMNP
      DO 7 I=1,NDF
      IF(ID(I,N).NE.0) GO TO 8
    7 CONTINUE
      GO TO 9
    8 WRITE (6,11) N,(ID(I,N),I=1,NDF)
    9 CONTINUE
      NG=0
      IF(TWR) WRITE (NTAPE,FOL) (J,NG,(ID(I,J),I=1,NDF),J=1,NUMNP)
      RETURN
C
   10 FORMAT (16I5)
   11 FORMAT (I10,4I13)
   12 FORMAT (1H1,5X,20A4,//5X,'NODAL B.C.'//6X,'NODE',9(I7,A4,A2)/)
      END
```

```
      SUBROUTINE BNCOND (S,FL,IDL,BVL,NST,NDFE)
C     ------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION IDL(1), BVL(1), S(NST,1), FL(1)
C
C     MODIFY THE R.H.S. ACCORDING TO ELEMENT B.C.
C     BVL    ; BOUNDARY VALUE
C     S      ; ELEMENT STIFFNESS MATRIX (NST,NST)
C     FL     ; RIGHT-HAND SIDE
C     NDFE   ; NUMBER OF EQUATIONS
C     NST    ; MAX. NUMBER OF EQUATION PER ELEMENT
C
C               CALLED BY - FORMEL
C
      DO 2 I=1,NDFE
      IBC=IDL(I)
C
C     CODE FOR DIRICHLET TYPE CONSTRAINT BOUNDARY CONDITIONS ARE GIVEN
C     AS '1' OR '-1', AND HAVE BEEN CHANGED TO '0' ALREADY.
C
      IF(IBC.NE.0) GO TO 2
C
C     MODIFY THE R.H.S. FOR NON HOM. DIRICHLET COND.
C
      DO 1 J=1,NDFE
      IF(I.NE.J) FL(J)=FL(J)-S(J,I)*BVL(I)
 1    CONTINUE
 2    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CPTABL (TITLE,IRITE)
C     ---------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /TIMECK/ CPTIME(20),COMPUT(20),NCHK
C
C               CALLED BY - ACNTRL
C
      NCHK=NCHK+1
      N=NCHK-1
      IF(IRITE.EQ.0) GO TO 1
      WRITE (6,4) (COMPUT(I),CPTIME(I),I=1,N)
      GO TO 3
 1    IF(NCHK.LE.20) GO TO 2
      WRITE (6,4) (COMPUT(I),CPTIME(I),I=1,N)
      NCHK=1
 2    CPTIME(NCHK)=RCLOK1(1.0)
      COMPUT(NCHK)=TITLE
 3    RETURN
C
 4    FORMAT (1H1,///10X,'TIME ELAPSED FOR EACH COMMAND'//20(10X,'MACRO'
     1,5X,A4,'  ;  ',F10.6,' SECONDS'/))
      END
```

```
      FUNCTION DOT (A,B,N)
C     ---------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... VECTOR DOT PRODUCT
C
C               CALLED BY - ACTCOL
C
      DIMENSION A(1), B(1)
      DOT=0.0D0
      DO 1 I=1,N
    1 DOT=DOT+A(I)*B(I)
      RETURN
      END
```

```
      SUBROUTINE ELDATA (IX,IDL,NEN1,NDF,PRT,ERR,TRD,TWR,IDIF)
C     ------------------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /LABL/ PD(9),A(9),BC(2),DI(6),CD(3),TE(3),FD(3),FOL(3),FON(
     14)
      LOGICAL ERR,PRT,TRD,TWR
      DIMENSION IX(NEN1,1), IDL(1)
C
C               CALLED BY - ACNTRL
C
      FOL(2)=A(NEN1)
      IF(TRD) GO TO 10
      L=0
      DO 7 N=1,NUMEL
      IF(L-N) 1,3,5
 1    READ (5,15) L,LK,LX,(IDL(K),K=1,NEN)
C        L  = ELEMENT NO.
C        LK = MATERIAL SET NO.
C        LX = GENERATION INCREMENT. ALL NODES ARE INCREMENTED BY LX
C             FOR EVERY ELEMENT BETWEEN L AND L ON THE NEXT CARD.
C        ELEMENTS MUST BE IN ORDER.
      NP1=N+1
      IF(L.EQ.0) L=NUMEL+1
      IF(LX.EQ.0) LX=1
      IF(L-N) 2,3,5
 2    WRITE (6,18) L,N
      ERR=.TRUE.
      GO TO 7
 3    NX=LX
      DO 4 K=1,NEN1
 4    IX(K,L)=IDL(K)
      IX(NEN1,L)=LK
      GO TO 7
 5    IX(NEN1,N)=IX(NEN1,N-1)
      DO 6 K=1,NEN
      IX(K,N)=IX(K,N-1)+NX
 6    IF(IX(K,N-1).EQ.0) IX(K,N)=0
 7    CONTINUE
C      .............COMPUTE THE HALF BAND WIDTH IBAND.............
      IDIF=0
      NENII=NEN-1
      DO 9 I=1,NUMEL
      DO 9 J=1,NENII
      IF(IX(J,I).LE.0) GO TO 9
      L=J+1
      DO 8 K=L,NEN
      IF(IX(K,I).LE.0) GO TO 8
      NDIF=IABS(IX(K,I)-IX(J,I))
      IF(IDIF.LT.NDIF) IDIF=NDIF
 8    CONTINUE
 9    CONTINUE
```

```
      IF(TWR) WRITE (NTAPE,FOL) (J,(IX(I,J),I=1,NEN1),J=1,NUMEL),IDIF
      GO TO 11
   10 READ (NTAPE,FOL) (J,(IX(I,J),I=1,NEN1),J=1,NUMEL),IDIF
   11 IF(.NOT.PRT) RETURN
      DO 12 I=1,NUMEL,50
      WRITE (6,16) O,HEAD,(K,K=1,NEN)
      J=MINO(NUMEL,I+49)
      DO 12 N=I,J
   12 WRITE (6,17) N,IX(NEN1,N),(IX(K,N),K=1,NEN)
      IF(TWR) WRITE (6,13) NTAPE
      IF(TRD) WRITE (6,14) NTAPE
      RETURN
C
   13 FORMAT (//5X,'.....SAVED ON FILE NO :',I4)
   14 FORMAT (//5X,'.....RTEAD FROM FILE NO :',I4)
   15 FORMAT (16I5)
   16 FORMAT (1H1,A1,5X,20A4//5X,8HELEMENTS//3X,7HELEMENT,2X,8HMATERIAL,
     114(I3,5H NODE)/(20X,14(I3,5H NODE)))
   17 FORMAT (2I10,14I8/(20X,14I8))
   18 FORMAT (5X,20H**ERROR 03** ELEMENT,I5,22H APPEARS AFTER ELEMENT,I5
     1)
      END
```

```
      SUBROUTINE ELMLIB (IXL,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,N,IEL,N
C     ----------------------------------------------------------------------
     1G)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION IXL(1), XL(1), D(1), FL(1), S(NST,1), UL(1), TL(1)
C
C               CALLED BY -  FORMEL
C               CALLS     - ZERO
C
C
 1    IF(ISW.EQ.2) CALL ZERO (S,NST*NST)
      IF(ISW.NE.5) CALL ZERO (FL,NST)
C
      IF(IEL.EQ.2) GO TO 2
      IF(IEL.EQ.3) GO TO 3
      CALL ELMT01 (IXL,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,IEL,N,NG)
      RETURN
 2    CALL ELMT02 (IXL,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,IEL,N,NG)
      RETURN
 3    CALL ELMT03 (IXL,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,IEL,N,NG)
      RETURN
      END
```

```
      SUBROUTINE FORMEL (X,XL,D,DL,S,UL,FL,TL,U,F,T,IX,IE,ID,LD,IXL,JDIA
C     ------------------
     1G,NDF,NM,NEN1,NST,ISW,A,C,B,AFL,BFL,CFL,PRT,TLD,NRD,
     2NOW,IBAND,BND)
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL AFL,BFL,CFL,DFL,EFL,PRT,TLD,BND
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /STRES/ EPS(6),SIG(6)
      DIMENSION X(NM,1), D(NRD,NUMMAT,1), F(NDF,1), XL(NM,1), FL(NDF,1),
     1 S(NST,1), T(1), UL(NDF,1), TL(1), IX(NEN1,1), LD(NDF,1), ID(NDF,1
     2), JDIAG(1), B(1), A(1), C(1), U(1), IXL(1), IE(4,1), DL(NRD)
C
C               CALLED BY - ACNTRL
C               CALLS     - ELMLIB,BNCOND,ADDSTF,BANNAD
C
      NDM=NM
      MA1=0
      IF(ISW.EQ.5) REWIND 14
      REWIND 13
C     .... LOOP ON ELEMENTS
      DO 11 N=1,NUMEL
      MA=IX(NEN1,N)
      IF(MA.EQ.MA1) GO TO 2
      IEL=IE(1,MA)
      NT=IE(2,MA)
      NGP=IE(3,MA)
      IF(ISW.EQ.5) NGP=IE(4,MA)
      DO 1 I=1,NRD
    1 DL(I)=D(I,MA,1)
      MA1=MA
    2 EFL=.FALSE.
C     .... SET UP LOCAL ARRAYS
      DO 8 I=1,NEN
      II=IX(I,N)
      IXL(I)=II
      TL(I)=0.0D0
      IF(II.NE.0) GO TO 5
      DO 3 J=1,NDM
    3 XL(J,I)=0.0D0
      DO 4 J=1,NDF
      UL(J,I)=0.0D0
    4 LD(J,I)=0
      GO TO 8
    5 NEL=I
      IID=II*NDF-NDF
      IF(TLD.OR.ISW.EQ.2) TL(I)=T(II)
      DO 6 J=1,NDM
    6 XL(J,I)=X(J,II)
      DO 7 J=1,NDF
      K=IABS(ID(J,II))
      UL(J,I)=F(J,II)
      IF(K.GT.0) UL(J,I)=U(K)
```

```
      IF(K.EQ.0.AND.F(J,II).NE.0) EFL=.TRUE.
      LD(J,I)=K
      IF(BND) UL(J,I)=U(IID+J)
 7    IF(BND) LD(J,I)=IID+J
 8    CONTINUE
C
      CALL ELMLIB (IXL,XL,DL,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,N,IEL,NGP)
C
      NDFE=NDF*NEL
      IF(.NOT.PRT) GO TO 9
      WRITE (6,14) N,((LD(I,J),I=1,NDF),J=1,NEL)
      IF(ISW.EQ.2) WRITE (6,15) ((S(I,J),J=1,NST),I=1,NST)
      IF(ISW.EQ.5) WRITE (6,16) ((UL(I,J),I=1,NDF),J=1,NEL)
      IF(ISW.EQ.3.OR.ISW.EQ.4) WRITE (6,17) ((FL(I,J),I=1,NDF),J=1,NEL)
      IF(ISW.EQ.3) WRITE (6,18) (TL(I),I=1,NEL)
 9    IF(BND) GO TO 10
      IF(AFL.AND.BFL.AND.EFL) CALL BNCOND (S,FL,LD,UL,NST,NDFE)
      IF(AFL.OR.BFL.OR.CFL) CALL ADDSTF (A,B,C,S,FL,JDIAG,LD,NST,NDFE,AF
     1L,BFL,CFL)
      GO TO 11
 10   IF(AFL.OR.BFL) CALL BANADD (A,MEQ,IBAND,B,FL,S,NST,NDFE,LD,AFL,BFL
     1)
 11   CONTINUE
      IF(ISW.NE.5.OR..NOT.PRT) GO TO 13
      IF(IEL.LE.2) GO TO 13
      REWIND 14
      DO 12 II=1,NUMEL,50
      WRITE (6,26) HEAD
      JJ=MINO(NUMEL,II+49)
      DO 12 J=II,JJ
      MA=IX(NEN1,J)
      NG=IE(4,MA)
      IEL=IE(1,MA)
      DO 12 K=1,NG
      READ (14) XX,YY,ZZ,EPS,P1,P2,P3,A1
 12   CONTINUE
 13   CONTINUE
      IF(.NOT.PRT.OR.ISW.NE.2) RETURN
      JKL=JDIAG(NEQ)
      IF(BND) JKL=MEQ*IBAND
      WRITE (6,19) ISW,NEQ,MEQ,JKL
      IF(.NOT.BND) WRITE (6,20) (JDIAG(I),I=1,NEQ)
      WRITE (6,21) (A(I),I=1,JKL)
      IF(CFL) WRITE (6,22) (C(I),I=1,JKL)
      WRITE (6,23) (B(I),I=1,NEQ)
      WRITE (6,24) (U(I),I=1,NEQ)
      WRITE (6,25) ((F(I,J),I=1,NDF),J=1,NUMNP)
      RETURN
C
 14   FORMAT (//2X,'STIFF. N = ',I5,5X,'LD  ',24I4//)
 15   FORMAT (10X,8E12.4)
 16   FORMAT (5X,'UL',3X,8E12.4)
```

```
 17   FORMAT (5X,'FL',3X,8E12.4)
 18   FORMAT (5X,'TL',3X,8E12.4)
 19   FORMAT (//5X,'ISW, NEQ, MEQ, JKL = NCOEF,',4I10/)
 20   FORMAT (/5X,'JDIAG',20I6)
 21   FORMAT (5X,'A  ',2X,10E12.4)
 22   FORMAT (5X,'C  ',2X,10E12.4)
 23   FORMAT (5X,'B  ',2X,10E12.4)
 24   FORMAT (5X,'U  ',2X,10E12.4)
 25   FORMAT (5X,'F  ',2X,10E12.4)
 26   FORMAT (1H1,5X,20A4,/,5X,'.... ELEMENT STRAINS .... CCW ROTATION',
     1' IS POSITIVE ....',/,5X,'.... 1-DIRECTION IS RADIAL DIRECTION,','
     2 2-DIRECTION IS AXIAL DIRECTION FOR AXISYMMETRIC PROBLEMS ....',//
     3/,2X,'EL  1-COORD.   2-COORD.   1-STRAIN   2-STRAIN   3-STR','AI
     4N  12-STRAIN',T78,'P1',T89,'P2',T100,'P3',T111,'A1',/)
      END
```

```
      SUBROUTINE GENVEC (JKL,X,CDD,PRT,ERR,TRD,TWR)
C     ---------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... GENERATE REAL DATA ARRAYS BY LINEAR INTERPOLATION
C
C         CALLED BY - ACNTRL
C         CALLS     - PCOMP, ZERO
      LOGICAL PRT,ERR,PCOMP,TRD,TWR
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /LABL/ PD(9),A(9),BC(2),DI(6),CD(3),TE(3),FD(3),FOL(3),FON(
     14)
      DIMENSION X(JKL,1), XL(6), CDD(3), FACT(6)
      DATA BL/4HBLAN/,GEOM/4H COO/
C
      FON(2)=A(JKL)
      IF(TRD) GO TO 12
      CALL ZERO (X,JKL*NUMNP)
      IF(.NOT.PCOMP(CDD(1),GEOM)) GO TO 2
      DO 1 N=1,NUMNP
    1 X(1,N)=BL
    2 N=0
      NG=0
    3 L=N
      LG=NG
C     IF(LG .EQ. 0) READ(5,1000) N,NG,(XL(I),I=1,JKL),(FACT(J),J=1,JKL)
      IF(LG.EQ.0) READ (5,17) N,NG,(XL(I),I=1,JKL)
      IF(LG.NE.0) READ (5,17) N,NG,(XL(I),I=1,JKL)
      IF(N.LE.0.OR.N.GT.NUMNP) GO TO 11
      DO 4 I=1,JKL
    4 X(I,N)=XL(I)
      IF(LG) 5,3,5
    5 LG=ISIGN(LG,N-L)
      LI=(IABS(N-L+LG)-1)/IABS(LG)
      DO 7 I=1,JKL
      F=1.+FACT(I)
      FO=1.0DO
      FF=FO
      DO 6 J=2,LI
      FO=FO*F
    6 FF=FF+FO
    7 XL(I)=(X(I,N)-X(I,L))/FF
      FF=1.0DO
    8 L=L+LG
      IF((N-L)*LG.LE.0) GO TO 3
      IF(L.LE.0.OR.L.GT.NUMNP) GO TO 10
      DO 9 I=1,JKL
      X(I,L)=X(I,L-LG)+XL(I)
    9 XL(I)=XL(I)*(1.+FACT(I))
      GO TO 8
   10 WRITE (6,21) L,(CDD(I),I=1,3)
      ERR=.TRUE.
```

```
      GO TO 3
 11   IF(TWR) WRITE (NTAPE,FON) (J,(X(I,J),I=1,JKL),J=1,NUMNP)
      GO TO 13
 12   READ (NTAPE,FON) (J,(X(I,J),I=1,JKL),J=1,NUMNP)
 13   IF(.NOT.PRT) RETURN
      DO 14 I=1,NUMNP,50
      WRITE (6,18) HEAD,(CDD(L),L=1,3),(L,CDD(1),CDD(2),L=1,JKL)
      N=MINO(NUMNP,I+49)
      DO 14 J=I,N
      IF(PCOMP(X(1,J),BL)) WRITE (6,19) N
 14   IF(.NOT.PCOMP(X(1,J),BL)) WRITE (6,20) J,(X(L,J),L=1,JKL)
      IF(TWR) WRITE (6,15) NTAPE
      IF(TRD) WRITE (6,16) NTAPE
      RETURN
C
 15   FORMAT (//5X,'.....SAVED ON FILE NO :',I4)
 16   FORMAT (//5X,'.....READ FROM FILE NO :',I4)
 17   FORMAT (2I5,7F10.0)
 18   FORMAT (1H1,5X,20A4//5X,5HNODAL,3A4//6X,4HNODE,9(I7,A4,A2))
 19   FORMAT (I10,32H HAS NOT BEEN INPUT OR GRNERATED)
 20   FORMAT (I10,9F13.4)
 21   FORMAT (5X,43H**FATAL ERROR 02** ATTEMPT TO GENERATE NODE,I5,3H IN
     1,3A4)
      END
```

```
      SUBROUTINE MATLIN (D,NADD,IE,NUMMAT,NRD,DL)
C     ------------------------------------------
C     .... CALLED BY : ACNTRL
C     .... CALLS     : MATRLD, MELAW
C      READ ALL MATERIAL DATA AND STORE IN (D).
C      FORM STRESS - STRAIN LAW IF ISOTHERMAL MATERIAL (NTEMP = 1).
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 K1,K2
      LOGICAL STRES
      COMMON /MLIST/ IEL,NTEMP,NGAUSS,NSTRES,ANG,T,E1,E2,XNU1,XNU2,G2,AL
     1PA1,ALPA2,SC,ST,PHI,RHO,K1,K2,C1,C2,A,C,A1,A2,A3,A4
      COMMON /ELABL/ ITYPE(6,10)
      DIMENSION D(NRD,NUMMAT,1), IE(4,1), MHEAD(20), DL(NRD)
      NAMELIST /MATRL/ T,E1,E2,XNU1,XNU2,G2,RHO,ALPA1,ALPA2,SC,ST,PHI,K1
     1,K2,C1,C2,A,C,A1,A2,A3,A4
      NAMELIST /MATID/ MATNO,ELTYPE,NTEMP,NGAUSS,NSTRES,ANG
      MTEMP=1
      WRITE (6,7)
C
      DO 3 MA=1,NUMMAT
C
      READ (5,6) MHEAD
      READ (5,MATID)
      IEL=ELTYPE
      STRES=.FALSE.
      STRES=.TRUE.
      MTEMP=MAXO(MTEMP,NTEMP)
      IF(MATNO.LT.1.OR.MATNO.GT.NUMMAT) GO TO 4
      IE(1,MATNO)=IEL
      IE(2,MATNO)=NTEMP
      IE(3,MATNO)=NGAUSS
      IE(4,MATNO)=NSTRES
C     IF(STRES) WRITE(6,610) MATNO,MHEAD,IEL,(ITYPE(I,IEL),I=1,6)
      IF(STRES) WRITE (6,8) MATNO,MHEAD,IEL
      IF(.NOT.STRES) WRITE (6,8) MATNO,MHEAD,IEL
      WRITE (6,9) NGAUSS,NSTRES,NTEMP,ANG
C
      DO 2 NT=1,NTEMP
      G2=0.0D0
      READ (5,MATRL)
      IF(G2.LE.0.0D0) G2=0.5D0*E1/(1.0D0+XNU1)
      IF(STRES) GO TO 1
C
C     .... MATERIAL PROPERTIES FOR A TRANSPORT PROBLEM
      CALL MATRLD (14,1,6,MATNO,NT,D,NUMMAT,NRD)
C
      D(7,MATNO,NT)=RHO
      WRITE (6,10) (D(I,MATNO,NT),I=1,7)
C
 1    CALL MATRLD (1,11,16,MATNO,NT,D,NUMMAT,NRD)
      WRITE (6,11) NT,(D(I,MATNO,NT),I=12,26)
C
```

```
 2    CONTINUE
C
      IF(.NOT.STRES.OR.NTEMP.GE.2) GO TO 3
C     .... ISOTHERMAL MATERIALS - FORM ELASTIC LAW HERE
      CALL MELAW (IEL,D(1,MATNO,1),DL)
 3    CONTINUE
C
      NADD=NUMMAT*MTEMP*NRD
C
      GO TO 5
 4    WRITE (6,12) MATNO
      STOP
 5    RETURN
C
 6    FORMAT (20A4)
 7    FORMAT (///5X,'M A T E R I A L  D A T A '/)
 8    FORMAT (//5X,'MATERIAL SET NO.        =',I4,' : ',20A4//5X,'ELEMEN
     1T TYPE NO.        =',I4,' : ',6A4)
 9    FORMAT (5X,'NO. OF GAUSS PTS.       =',I4/5X,'NO. OF STRESS PTS.
     1    =',I4/5X,'NO. OF TEMP - MATRL PTS.=',I4/5X,'ANGLE FROM X1 TO E
     21     =',F10.3,'  DEGREE ( C. C. W. IS POSITIVE).'/)
 10   FORMAT (8X,'CONDUCTIVITY     K1  =',E14.5/8X,'CONDUCTIVITY     K2
     1 =',E14.5/8X,'CONVECTION       C1  =',E14.5/8X,'CONVECTION       C
     22  =',E14.5/8X,'SOURCE/SINK      A   =',E14.5/8X,'CAPACITY
     3 C   =',E14.5/8X,'MASS DENSITY     RHO =',E14.5)
 11   FORMAT (5X,'TEMP. PT. NO. =',I4,' : TEMPERATURE =',E14.5//8X,'MOD.
     1 OF ELASTICITY        E1    =',E14.5/8X,'MOD. OF ELASTICITY
     2 E2    =',E14.5/8X,'POISSONS RATIO            XNU1  =',E14.5/8X,'P
     3OISSONS RATIO            XNU2  =',E14.5/8X,'INDEP. SHEAR MOD.
     4    G2    =',E14.5/8X,'FLUID COMPRESSIBILITY  ALPA1    =',E14.5/8X
     5,'SOLID COMPRESSIBILITY  ALPA2    =',E14.5/8X,'STIFFNESS DAMPING C
     6OEF.    SC   =',E14.5/8X,'MASS DAMPING COEF.          ST  =',E14.5
     7/8X,'PROSOITY                 PHI   =',E14.5/8X,'MASS DENSITY
     8          RHO   =',E14.5/8X,'PERMEABILITY              K1    =',E1
     94.5/8X,'PERMEABILITY              K2    =',E14.5/8X,'FLUID DENSITY
     £            C1    =',E14.5/)
 12   FORMAT (////5X,'..... FATAL ERROR IN THE NAMELIST INPUT *MATID* ..
     1...'/10X,'MATERIAL SET NO. (MATNO) =',I5,' IS NOT POSSIBLE.
     2  EXECUTION TERMINATED.')
      END
```

```
      SUBROUTINE MATRLD (NC1,ND1,N,MA,NT,D,NUMMAT,NRD)
C     ------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /MLIST/ NDUM(4),DUMY(23)
      DIMENSION D(NRD,NUMMAT,1)
C
C     .... CALLED BY : MATLIN
C      PUT NAMELIST INPUT VARIABLES ( STORED IN DUMY ) INTO THE ARRAY D
C       D(11,MA,NT) = ANG
C       D(12,MA,NT) = T
C       D(13,MA,NT) = E1
C       D(14,MA,NT) = E2
C       D(15,MA,NT) = XNU1
C       D(16,MA,NT) = XNU2
C       D(17,MA,NT) = G2
C       D(18,MA,NT) = ALPA1 :  FLUID COMPRESSIBILITY
C       D(19,MA,NT) = ALPA2 :  SOLID COMPRESSIBILITY
C       D(20,MA,NT) = SC    :  STIFFNESS DAMPING COEFFICIENT
C       D(21,MA,NT) = ST    :  MASS DAMPING COEFFICIENT
C       D(22,MA,NT) = PHI   :  POROSITY
C       D(23,MA,NT) = RHO
C       D(24,MA,NT) = K1 : PERMEABILITY IN 1
C       D(25,MA,NT) = K2 : PERMEABILITY IN 2
C       D(26,MA,NT) = C1 : FLUID DENSITY
C
      DO 1 I=1,N
      II=I+NC1-1
      ID=I+ND1-1
 1    D(ID,MA,NT)=DUMY(II)
      RETURN
      END
```

```
      SUBROUTINE MELAW (IEL,D,DL)
C  ------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... CALLED BY : MATLIN
C     .... CALLS     : METRAN
C      COMPUTE STRESS-STRAIN RELATIONS USING CONSTANTS IN D(11-23).
C
      DIMENSION D(1), DL(1)
      E1=D(13)
      E2=D(14)
      XNU1=D(15)
      XNU2=D(16)
      XN=E1/E2
      I=2
      D(8)=D(18)*(1.0D0+XNU1*(I-1))
      D(9)=D(19)*(1.0D0+XNU2*(I-1))
      DD=E2/(1.0D0+(I-1)*XNU1)/(1.0D0-(I-1)*XNU1-I*XN*XNU2*XNU2)
      D(1)=XN*(1.0D0-(I-1)*XN*XNU2*XNU2)*DD
      D(2)=XN*XNU2*(1.0D0+(I-1)*XNU1)*DD
      D(3)=(1.0D0-(I-1)*XNU1*XNU1)*DD
      D(4)=D(17)
      D(5)=0.0D0
      D(6)=0.0D0
      D(10)=0.0D0
C     WRITE(6,600) (D(I),I=1,25)
      RETURN
 1    RETURN
C
      END
```

```
      LOGICAL FUNCTIONPCOMP(A,B)
C     ----------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C               CALLED BY - GENVEC,ACNTRL,PRTDIS
C               CALLS     -
C
      PCOMP=.FALSE.
C     .... IT MAY BE NECESSARY TO REPLACE THE FOLLOWING ALPHANUMERIC
C     .... COMPARISON STATEMENT IF COMPUTER PRODUCES AN OVERFLOW
      IF(A.EQ.B) PCOMP=.TRUE.
      RETURN
      END
```

```
      SUBROUTINE PLOAD (ID,F,B,R,T,NN,P,IW,BND)
C     ------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL BND
C
C     .... FORM LOAD VECTOR IN COMPACT FORM
C
C               CALLED BY - ACNTRL
C               CALLS     - ZERO
C
      DIMENSION ID(1), F(1), B(1), T(1), R(1)
      IF(BND) GO TO 4
      IF(IW.NE.0) GO TO 2
C     ....CLEAR THE R.H.S.(B) AND GET NODAL BOUNDARY LOADING.
      CALL ZERO (B,NN)
      DO 1 N=1,NN
      J=ID(N)
    1 IF(J.GT.0) B(J)=F(N)*P+R(J)
      RETURN
C     ....STORE PRESCRIBED TEMPERATURE AND SOLUTION IN (T).
    2 DO 3 N=1,NN
      J=ID(N)
      T(N)=F(N)
    3 IF(J.GT.0) T(N)=B(J)
      RETURN
    4 IF(IW.NE.0) GO TO 6
      CALL ZERO (B,NN)
      DO 5 I=1,NN
    5 IF(ID(I).EQ.0) B(I)=F(I)*P+R(I)
      RETURN
    6 DO 7 I=1,NN
    7 T(I)=B(I)
      RETURN
      END
```

```
      SUBROUTINE PROFIL (JDIAG,ID,IX,NDF,NEN1,NAD,TRD,TWR)
C     -----------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... COMPUTE PROFILE OF GLOBAL ARRAYS
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      LOGICAL TRD,TWR
      DIMENSION JDIAG(1), ID(NDF,1), IX(NEN1,1)
C     .... SET UP THE EQUATION NUMBERS
C
C               CALLED BY - ACNTRL
C
      NEQ=0
      DO 4 N=1,NUMNP
      DO 3 I=1,NDF
      J=ID(I,N)
      IF(J) 2,1,2
 1    NEQ=NEQ+1
      ID(I,N)=NEQ
      JDIAG(NEQ)=0
      GO TO 3
 2    ID(I,N)=0
 3    CONTINUE
 4    CONTINUE
C     .... COMPUTE COLUMN HEIGHTS
      DO 9 N=1,NUMEL
      DO 8 I=1,NEN
      II=IX(I,N)
      IF(II.EQ.0) GO TO 8
      DO 7 K=1,NDF
      KK=ID(K,II)
      IF(KK.EQ.0) GO TO 7
      DO 6 J=I,NEN
      JJ=IX(J,N)
      IF(JJ.EQ.0) GO TO 6
      DO 5 L=1,NDF
      LL=ID(L,JJ)
      IF(LL.EQ.0) GO TO 5
      M=MAXO(KK,LL)
      JDIAG(M)=MAXO(JDIAG(M),IABS(KK-LL))
 5    CONTINUE
 6    CONTINUE
 7    CONTINUE
 8    CONTINUE
 9    CONTINUE
C     .... COMPUTE DIAGONAL POINTERS FOR PROFILE
      NAD=1
      JDIAG(1)=1
      IF(NEQ.EQ.1) RETURN
      DO 10 N=2,NEQ
 10   JDIAG(N)=JDIAG(N)+JDIAG(N-1)+1
      NAD=JDIAG(NEQ)
```

```
74
      RETURN
C
      END
```

```
      SUBROUTINE PRTDIS (ID,X,U,F,NM,NF,TRD,TWR,PRT,NSTEP,INCR,TIME,BND)
C     ------------------------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
C     .... OUTPUT NODAL VALUES
C
C              CALLED BY - ACNTRL,CONSOL
C              CALLS     - PCOMP
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /LABL/ PD(9),A(9),BC(2),DI(6),CD(3),TE(3),FD(3),FOL(3),FON(
     14)
      COMMON /PRLOD/ PROP
      LOGICAL PCOMP,TRD,TWR,PRT,BND
      DIMENSION X(NM,1), F(NF,1), ID(NF,1), U(1), UL(6), BV(2), IBC(6)
      DATA BL/4HBLAN/,BV/4H B.V,4HALUE/
C
      NTAPE=11
      NDM=NM
      PD(2)=A(NF)
      PD(4)=A(NM)
      PD(6)=A(NF)
      PD(8)=A(NF)
C
      IF(TWR) WRITE (NTAPE,4) NF
      DO 3 II=1,NUMNP,50
      IF(PRT) WRITE (6,5) HEAD,NSTEP,INCR,TIME,BC(1),BC(2),(I,CD(1),CD(2
     1),I=1,NDM),(I,BV(1),BV(2),I=1,NF),(I,DI(1),DI(2),I=1,NF)
C     IF(II .EQ. 1 .AND. TWR) WRITE(NTAPE,610) HEAD,NSTEP,INCR,TIME
      IF(II.EQ.1.AND.TWR) WRITE (NTAPE,5) HEAD,NSTEP,INCR,TIME,BC(1),BC(
     12),(I,CD(1),CD(2),I=1,NDM),(I,BV(1),BV(2),I=1,NF),(I,DI(1),DI(2),I
     2=1,NF)
      JJ=MINO(NUMNP,II+49)
      DO 3 N=II,JJ
      IF(PCOMP(X(1,N),BL)) GO TO 2
      DO 1 I=1,NF
      UL(I)=F(I,N)*PROP
      K=IABS(ID(I,N))
      IBC(I)=0
      IF(K.EQ.0) IBC(I)=1
      IF(.NOT.BND) GO TO 1
      IBC(I)=K
      K=(N-1)*NF+I
    1 IF(K.GT.0) UL(I)=U(K)
      IF(PRT) WRITE (6,PD) N,(IBC(I),I=1,NF),(X(I,N),I=1,NDM),(F(I,N),I=
     11,NF),(UL(I),I=1,NF)
      IF(TWR) WRITE (NTAPE,PD) N,(IBC(I),I=1,NF),(X(I,N),I=1,NDM),(F(I,N
     1),I=1,NF),(UL(I),I=1,NF)
    2 CONTINUE
    3 CONTINUE
C     .       '  TIME =',E12.4//1X,'NODE',1X,A4,A2,I2,A4,A2,9(I4,A4,A2))
C     610  FORMAT(20A4,3X,'STEP NO. =',I5,' INCR. NO. =',I5,' TIME =',E1
```

```
      RETURN
C
 4    FORMAT (I5)
      END
```

```
      SUBROUTINE PVALUE (S,IS,N,X,Y,Z,LNO,IEL,TE,D)
C     ---------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,TWR
      DIMENSION S(6), D(1), IFAIL(4)
      DATA RT2/1.41421356237309/,PI23/2.09439510239321/
      DATA IFAIL/1H ,1HT,1HC,1HS/
      LOGICAL TWR
C     .... COMPUTE AND PRINT PRINCIPAL VALUES OF STRESS (IS=1) OR
C      STRAIN (IS=2)
C
C      S = STRAIN OR STRESS TENSOR STORED IN VECTOR FORM
C      IS = 1 IF S IS STRESS
C         = 2 IF S IS STRAIN
C      N = ORDER OF THE TENSOR S ( 2 OR 3 )
C     .... COMPONENTS OF S MUST BE IN THE FOLLOWING ORDERS
C
C      S11,S12,S22,S13,S23,S33
C
      IF(IS.EQ.2) GO TO 2
C
      IF(LINE.NE.0) GO TO 1
      WRITE (6,7) HEAD
      IF(N.EQ.2) WRITE (6,6) ST(IS),ST(IS),ST(IS)
      IF(N.EQ.3) WRITE (6,8) ST(IS),ST(IS),ST(IS),ST(IS)
 1    LINE=LINE+1
 2    IF(N.EQ.3) GO TO 3
C     .... 2ND ORDER TENSOR ....
      S12=S(2)/IS
      SS=(S(1)-S(3))/2.0D0
      TT=(S(1)+S(3))/2.0D0
      RR=DSQRT(SS*SS+S12*S12)
      P1=TT+RR
      P2=TT-RR
      A1=45.0D0
      IF(S12.EQ.0.0D0.OR.SS.NE.0.0D0) A1=DATAN2(S12,SS)*ANG(1)/2.0D0
      IF(LINE.EQ.50) LINE=0
      IF(IS.NE.1) GO TO 5
      P3=0.0D0
      GO TO 5
C
C     .... 3RD ORDER TENSOR ....
C
 3    RR=0.0D0
      U=(S(1)+S(3)+S(6))/3.0D0
      V=S(1)*(S(3)+S(6))+S(3)*S(6)-S(2)*S(2)-S(4)*S(4)-S(5)*S(5)
      W=S(1)*S(3)*S(6)+2.0D0*S(2)*S(4)*S(5)-S(1)*S(5)*S(5)-S(3)*S(4)*S(4
     1)-S(6)*S(2)*S(2)
      TT=3.0D0*U*U-V
      SS=0.0D0
```

```
      IF(TT.EQ.0.0D0) GO TO 4
      UU=DSQRT(2.0D0*TT/3.0D0)
      AA=(W+(TT-U*U)*U)*RT2/UU**3
      RR=DSQRT(DABS(1.0D0-AA*AA))
      RR=DATAN2(RR,AA)/3.0D0
 4    P1=U+UU*RT2*DCOS(RR)
      P2=U+UU*RT2*DCOS(RR-PI23)
      P3=U+UU*RT2*DCOS(RR+PI23)
      IF(LINE.GE.50) LINE=0
      IF(IS.EQ.1) WRITE (6,9) LNO,IFAIL(IM),X,Z,S(1),S(6),S(3),S(4),P1,P
     12,P3,A1,D(12),D(13)
 5    IF(TWR.AND.IS.EQ.1) WRITE (NTAPE,10) LNO,IFAIL(IM),X,Y,Z,P1,P2,P3,
     1A1,A2,A3,TE,D(13)
      IF(IS.EQ.2) WRITE (14) X,Y,Z,S,P1,P2,P3,A1
      RETURN
C
 6    FORMAT (/,1X,'ELFAIL  1-COOR.   2-COOR.   1-STR',A3,3X,'2-STR',A
     13,'  12-STR',A3,T66,'P1',T77,'P2',T86,'TAU-MAX',T99,'A1',T109,'PRE
     2S',T120,'E(T)',/)
 7    FORMAT (1H1,5X,20A4/5X,'....ELEMENT  STRESS ....','CCW ROTATION IS
     1 POSITIVE....'/5X,'....1-DIRECTION IS RADIAL AND 2-DIRECTION IS AX
     2IAL FOR AN',1X,'AXISYMMETRIC PROBLEM....'//)
 8    FORMAT (/,1X,'ELFAIL 1-COOR.  2-COOR   1-STR',A3,3X,'2-STR',A3,'
     1   3-STR',A3,'  12-STR',A3,T77,'P1',T88,'P2',T99,'P3',T108,'A1',T1
     216,'TEMP',T127,'E(T)',/)
 9    FORMAT (1X,I4,1X,A1,1P2E10.2,1P7E11.3,F7.2,1P2E11.3)
 10   FORMAT (1X,I4,A1,6E11.3/6X,5E11.3)
      END
```

```
      SUBROUTINE SETMPT (NEXT,NOW,INCR,MTOT,TITLE)
C     ---------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /SIZE/ MAX
C     ....SET THE STORAGE POINTER NEXT = NOW + INCR AND CHECK FOR SUFFIC
C     MEMORY.
C
C               CALLED BY - ACNTRL
C
      MTOT=MTOT+INCR
      NEXT=NOW+INCR
      NN=NEXT-1
      N2=2*(NN/2)
      IF(NN.EQ.N2) GO TO 1
      NEXT=NEXT+1
      MTOT=MTOT+1
 1    IF(MTOT.LE.MAX) RETURN
      WRITE (6,2) TITLE,MTOT,MAX
      STOP
C
 2    FORMAT (///5X,'...FATAL ERROR...INSUFFICIENT STORAGE IN BLANK COMM
     1ON'/5X,'REQUIRED TO EXECUTE *H',A4,'* IS',I8,'  BUT AVAILABLE ONLY
     2',I8)
      END
```

```
      SUBROUTINE SHAPE2 (XL,IX,NDM,NEL,XSJ,M,N,NPT)
C     ------------------------------------------------
C
C     CALLS          SHAP22
C
C     SHAPE FUNCTION ROUTINE FOR 2-D ISOPARAMETRIC ELEMENTS
C     CALLED BY SUBROUTINE ELEMNT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      DIMENSION IX(1), XL(NDM,1), XS(3,2)
      X=GPT(M,NPT)
      E=GPT(N,NPT)
C     FORM 4-NODE QUADRILATERAL SHAPE FUNCTION
      DO 1 I=1,4
      SHP(3,I)=(1.+LX(I)*X)*(1.+LE(I)*E)/4.
      SHP(1,I)=LX(I)*(1.+LE(I)*E)/4.
 1    SHP(2,I)=LE(I)*(1.+LX(I)*X)/4.
C
      IF(NEL.GE.4) GO TO 3
C     FORM TRIANGULAR SHAPE FUNCTIONS BY ADDING 3-RD AND 4-TH
      DO 2 I=1,3
 2    SHP(I,3)=SHP(I,3)+SHP(I,4)
C     ADD QUADRATIC TERMS IF NECESSARY
 3    IF(NEL.GT.4) CALL SHAP22 (X,E,SHP,NEL,IX)
C
C     CONSTRUCT JACOBIAN XJ AND DETERMINANT OF XJ
      DO 4 I=1,NDM
      DO 4 J=1,2
      XS(I,J)=0.0D0
      DO 4 K=1,NEL
 4    XS(I,J)=XS(I,J)+XL(I,K)*SHP(J,K)
C
      IF(NDM.NE.3) GO TO 5
C     COMPUTE SURFACE AREA AND RETURN AS DETJ
      A1=XS(2,1)*XS(3,2)-XS(3,1)*XS(2,2)
      A2=XS(3,1)*XS(1,2)-XS(1,1)*XS(3,2)
      A3=XS(1,1)*XS(2,2)-XS(2,1)*XS(1,2)
      DETJ=DSQRT(A1*A1+A2*A2+A3*A3)
      RETURN
C
 5    DETJ=XS(1,1)*XS(2,2)-XS(1,2)*XS(2,1)
C     COMPUTE DNI/DX AND STORE IN SHP(1,I)
C     COMPUTE DNI/DY AND STORE IN SHP(2,I)
      DO 6 I=1,NEL
      TEMP=(XS(2,2)*SHP(1,I)-XS(2,1)*SHP(2,I))/DETJ
      SHP(2,I)=(-XS(1,2)*SHP(1,I)+XS(1,1)*SHP(2,I))/DETJ
 6    SHP(1,I)=TEMP
C
      XSJ=DETJ*WT(M,NPT)*WT(N,NPT)
C
      RETURN
      END
```

```
      SUBROUTINE SHAP22 (S,T,SHP,NEL,IX)
C     ----------------------------------
C
C
C     CALLED BY       SHAPE2
C
C     ADD QUADRATIC TERMS AS NECESSARY
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION IX(1), SHP(4,9)
C
      S2=(1.0D0-S*S)/2.0D0
      T2=(1.0D0-T*T)/2.0D0
      DO 1 I=5,NEL
      DO 1 J=1,3
 1    SHP(J,I)=0.0D0
C     .... MIDSIDE NODES (SERENDIPITY)
      IF(IX(5).EQ.0) GO TO 2
      SHP(1,5)=-S*(1.0D0-T)
      SHP(2,5)=-S2
      SHP(3,5)=S2*(1.0D0-T)
 2    IF(NEL.LT.6) GO TO 8
      IF(IX(6).EQ.0) GO TO 3
      SHP(1,6)=T2
      SHP(2,6)=-T*(1.0D0+S)
      SHP(3,6)=T2*(1.0D0+S)
 3    IF(NEL.LT.7) GO TO 8
      IF(IX(7).EQ.0) GO TO 4
      SHP(1,7)=-S*(1.0D0+T)
      SHP(2,7)=S2
      SHP(3,7)=S2*(1.0D0+T)
 4    IF(NEL.LT.8) GO TO 8
      IF(IX(8).EQ.0) GO TO 5
      SHP(1,8)=-T2
      SHP(2,8)=-T*(1.0D0-S)
      SHP(3,8)=T2*(1.0D0-S)
C     .... INTERIOR NODE (LAGRANGIAN)
 5    IF(NEL.LT.9) GO TO 8
      IF(IX(9).EQ.0) GO TO 8
      SHP(1,9)=-S*T2
      SHP(2,9)=-T*S2
      SHP(3,9)=4.0D0*S2*T2
C     .... CORRECT EDGE NODES FOR INTERIOR NODE (LAGRANGIAN)
      DO 7 J=1,3
      DO 6 I=1,4
 6    SHP(J,I)=SHP(J,I)-0.25D0*SHP(J,9)
      DO 7 I=5,8
 7    IF(IX(I).NE.0) SHP(J,I)=SHP(J,I)-0.5D0*SHP(J,9)
C     .... CORRECT CORNER NODES FOR PRESENCE OF MIDSIDE NODES
 8    K=8
      DO 10 I=1,4
      L=I+4
      DO 9 J=1,3
```

```
9     SHP(J,I)=SHP(J,I)-0.5DO*(SHP(J,K)+SHP(J,L))
10    K=L
      RETURN
      END
```

```
      SUBROUTINE ZERO (A,N)
C     ---------------------
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(1)
C
C               CALLED BY - ACNTRL,FORMEL
C
      DO 1 I=1,N
 1    A(I)=0.0D0
      RETURN
      END
```

```
      SUBROUTINE CONSOL (X,T,IX,IE,ID,IDL,IXL,KPRS,XL,DL,SPU,UL,FL,TL,
C     ----------------------------------------------------------------
     1W,F,R,D,AK,TIMES,NDT,NOPT,NPRINT,NEN,NEN1, NPN,
     2NDF,NDM,MEQ,IBAND,NUMEL,NUMNP,NSTEPS,NST,NRD,BND)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,TWR
      LOGICAL BACK,FORW,BND,SIG,TWR
      DIMENSION X(NDM,1), T(1), IX(NEN1,1), IE(4,1), ID(NDF,1), IDL(1),
     1IXL(1), KPRS(1), XL(1), DL(1), SPU(NST,1), UL(1), FL(1), TL(1), W(
     2NDF,1), F(NDF,1), D(1), AK(MEQ,IBAND), R(NDF,NUMNP), TIMES(1), NDT
     3(1), NOPT(1), NPRINT(1)
C
C     FILE ASSIGNMENT
C     15 - ELEMENT STIFFNESS
C     16 - GLOBAL STIFFNESS (MEQ,IBAND)
C     17 - GLOBAL INITIAL FORCE (NDF,NUMNP)
C
      SIG=.FALSE.
C
C     CORRECT BCODE IN THE 8-4 ELEMENT
C
      IF(NPN.EQ.NEN) GO TO 2
      NP1=NPN+1
      DO 1 M=1,NUMEL
      DO 1 I=NP1,NEN
      J=IX(I,M)
      ID(3,J)=1
 1    CONTINUE
 2    CONTINUE
C
C     ...INPUT TIME FACTORS
      READ 9, (TIMES(I),NDT(I),NOPT(I),NPRINT(I),KPRS(I),I=1,NSTEPS)
      PRINT 10, NEN,NPN
      PRINT 11
      PRINT 12, (I,TIMES(I),NDT(I),NOPT(I),NPRINT(I),KPRS(I),I=1,NSTEPS)
C
C     ....ADD EXTERNALLY APPLIED CONC. NODAL LOADS TO R AND INITIATE W
      DO 3 N=1,NUMNP
      DO 3 J=1,NDF
      IF(ID(J,N).NE.0) GO TO 3
      R(J,N)=R(J,N)+F(J,N)
      W(J,N)=0.0D0
 3    CONTINUE
C
C     ....STORE THE GLOBAL STIFFNESS MATRIX TO BE MULTIPED BY
C     ALPHA*DT IN SUBROUTINE TMSTP AND THE INITIAL LOAD VECTOR
      WRITE (16) AK,R
C
C
C     ....CONSOLIDATION COMPUTATIONS
      DO 8 LT=1,NSTEPS
      BACK=.FALSE.
```

```
      FORW=.TRUE.
      LT1=LT-1
      STEP=TIMES(LT)
      IF(LT.GT.1) STEP=TIMES(LT)-TIMES(LT1)
      DT=STEP/NDT(LT)
      IF(NOPT(LT).GT.4) ALPHA=1.0D0+(1.0D0/DT)-(1.0D0/(DLOG(1.0D0+DT)))
      IF(NOPT(LT).EQ.1) ALPHA=0.5D0
      IF(NOPT(LT).EQ.2) ALPHA=0.6666D0
      IF(NOPT(LT).EQ.3) ALPHA=0.871D0
      IF(NOPT(LT).EQ.4) ALPHA=01.D0
      IF(LT.EQ.1) GO TO 4
 4    CONTINUE
C
C     MULTIPLY  K2 IN AK BY ALPHA * DT, APPLY GEOMETRIC BOUNDARY
C     CONDITIONS, AND SAVE R (LOAD VECTOR)
      CALL TMSTP (ALPHA,DT,R,W,F,AK,IBAND,MEQ,NUMNP,NUMEL,IX,ID,NEN1,NDF
     1,SPU,NST,NPN)
C
C     ....STORE THE GEOMETRICALLY MODEFIED GLOBAL LOAD
C     VECTOR TO BE UBDATED IN SUBROUTINE TMINC
      REWIND 17
      WRITE (17) R
C
C     TRAINGULIZE THE STIFFNESS MATRIX
      CALL BANSOL (AK,R,R,MEQ,IBAND,BACK,FORW)
C
C     UPDATE THE LOAD VECTOR R, FOR EVERY TIME INCREMENT
      NDTS=NDT(LT)
C
C     SET AN OUTPUT COUNTER
      NO=NPRINT(LT)
      DO 7 LDT=1,NDTS
      CALL TMINC (R,W,SPU,ID,IX,NDF,NEN,NEN1,NUMNP,NUMEL,NST,NPN,ALPHA,D
     1T)
C     SOLVE FOR DISPL. AND PORE PRESS. CORRESPONDING TO R
      BACK=.TRUE.
      CALL BANSOL (AK,R,R,MEQ,IBAND,BACK,FORW)
      DO 5 I=1,NUMNP
      DO 5 J=1,NDF
 5    W(J,I)=R(J,I)
      TIME=TIMES(LT)+LDT*DT-NDTS*DT
      IF(LDT.EQ.NO.OR.LDT.EQ.NDTS) GO TO 6
      GO TO 7
 6    CALL PRTDIS (ID,X,R,F,NDM,NDF,.FALSE.,TWR,.TRUE.,LT,LDT,TIME,BND)
      NO=NPRINT(LT)+NO
      IF(KPRS(LT).EQ.0) GO TO 7
      LINE=0
      CALL FORMEL (X,XL,D,DL,SPU,UL,FL,TL,R,F,T,IX,IE,ID,IDL,IXL,IXL,NDF
     1,NDM,NEN1,NST,5,AK,AK,R,SIG,SIG,SIG,SIG,SIG,NRD,NOW,IBAND,BND)
 7    CONTINUE
 8    CONTINUE
      RETURN
```

```
C
 9    FORMAT (1F10.0,4I5)
 10   FORMAT (1H1/15X,'CONSOLIDATION ANALYSIS'//10X,'NUMBER OF ELEMENT N
     1ODES   = ',I5/10X,'NUMBER OF PRESSURE NODES  = ',I5//)
 11   FORMAT (6X,'STEP',7X,'ELLAPSED',2X,'NUMBER OF',6X,'TIME',4X,'OUTPU
     1T',3X,'STRESS'/4X,'NUMBER',11X,'TIME',4X,'TIM INC',2X,'INTERPOL',3
     2X,'COUNTER',3X,'OUTPUT')
 12   FORMAT (I10,1D15.5,4I10)
      END
```

```
      SUBROUTINE TMSTP (ALPHA,DT,R,W,F,AK,IBAND,
     1MEQ,NUMNP,NUMEL,IX,ID,NEN1,NDF,SPU,NST,NPN)
C     ------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION R(NDF,NUMNP), W(NDF,1), F(NDF,1), AK(MEQ,IBAND), ID(NDF,
     11), IX(NEN1,1), CPP(8,8), SPU(NST,NST)
      REWIND 16
      READ (16) AK,R
C
C     MULTIPLY K2 IN THE GLOBAL STIFFNESS BY ALPHA*DT
      DO 2 I=1,NUMNP
      II=NDF*I
      DO 1 J=1,NUMNP
      JJ=NDF*J
      KK=JJ-II+1
      IF(KK.LE.0.OR.KK.GT.IBAND) GO TO 1
      AK(II,KK)=AK(II,KK)*ALPHA*DT
 1    CONTINUE
 2    CONTINUE
C
C     ADD CPP TO AK GLOBALLY AND TO SPU LOCALLY
      REWIND 15
      DO 4 M=1,NUMEL
      READ (15) NSL,((SPU(I,J),J=1,NSL),I=1,NPN),((CPP(I,J),J=1,NPN),I=1
     1,NPN)
      DO 3 I=1,NPN
      II=NDF*IX(I,M)
      DO 3 J=1,NPN
      JJ=NDF*IX(J,M)
      KK=JJ-II+1
      IF(KK.LE.0) GO TO 3
      AK(II,KK)=AK(II,KK)-CPP(I,J)
 3    CONTINUE
 4    CONTINUE
C     INTRODUCE KINEMATIC CONSTRAINTS
      DO 6 M=1,NUMNP
      DO 5 J=1,NDF
      IF(ID(J,M).EQ.0) GO TO 5
      IDF=(M-1)*NDF+J
      CALL GEOMBC (F(J,M),IDF,IBAND,MEQ,R,AK)
 5    CONTINUE
 6    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TMINC (R,W,SPU,ID,IX,NDF,NEN,NEN1,NUMNP,NUMEL,NST,NPN,
C     ------------------------------------------------------------------
     1ALPHA,DT)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION R(NDF,NUMNP), W(1), ID(NDF,1), IX(NEN1,1), SPU(NST,NST),
     1 CPP(8,8)
      REWIND 15
      REWIND 17
      READ (17) R
      DO 4 M=1,NUMEL
      READ (15) NSL,((SPU(I,J),J=1,NSL),I=1,NPN),((CPP(I,J),J=1,NPN),I=1
     1,NPN)
      NEL=NSL/NDF
      DO 3 I=1,NPN
      J=IX(I,M)
      IF(ID(3,J).NE.0) GO TO 3
      DO 2 K=1,NEL
      IF(K.GT.NPN) GO TO 1
      KK=K*NDF
      SPU(I,KK)=SPU(I,KK)*((ALPHA-1.D0)*DT)-CPP(I,K)
    1 CONTINUE
      N=IX(K,M)
      DO 2 L=1,NDF
      KLOC=(K-1)*NDF+L
      KGLB=(N-1)*NDF+L
    2 R(3,J)=R(3,J)+SPU(I,KLOC)*W(KGLB)
    3 CONTINUE
    4 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ELMT01 (IX,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,
C     ------------------------------------------------------------
     1IL,LN,NG)
C
C     CALLED BY ELMLIB
C
C     CALLS       SHAPE2,PVALUE,GABOSI
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,EWR
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      COMMON /STRES/ EPS(6),SIG(6)
      DIMENSION D(1), XL(NDM,1), FL(NDF,1), S(NST,NST), UL(NDF,1), TL(1)
     1, IX(1), SS(3,8), CPP(8,8)
C
      NGP=NG
      NDU=NDF-1
      GO TO (1,1,15,16,21), ISW
 1    CONTINUE
C
C     ....COMPUTE ELEMENT STIFFNESS MATRIX BY(NGP*NGP)
      NSL=NEL*NDF
      DO 2 I=1,NPN
      DO 2 J=1,NPN
 2    CPP(I,J)=0.0D0
      IF(NEL.EQ.8) GO TO 3
      CALL GABOSI (IX,XL,D,S,CPP,NDM,NDF,NST,6,NPN,NGP)
      GO TO 14
 3    AK1=-D(24)
      AK2=-D(25)
      COMP=D(22)
      DO 10 M=1,NGP
      DO 10 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XDJ,M,N,NGP)
      DO 4 I=1,NDU
      DO 4 J=1,NEL
 4    SS(I,J)=SHP(I,J)
      CALL SHAPE2 (XL,IX,NDM,NPN,XPJ,M,N,NGP)
C
C     ....COMPUTE STIFFNESS MATRIX OF THE SOLID PHASE
C     COLUMN
      J1=1
      DO 6 J=1,NEL
      W1J=SS(1,J)*XDJ
      W2J=SS(2,J)*XDJ
      K1=J1
C     ROW
      DO 5 K=J,NEL
      S(J1,K1)=S(J1,K1)+W1J*SS(1,K)
      S(J1,K1+1)=S(J1,K1+1)+W1J*SS(2,K)
      S(J1+1,K1)=S(J1+1,K1)+W2J*SS(1,K)
      S(J1+1,K1+1)=S(J1+1,K1+1)+W2J*SS(2,K)
```

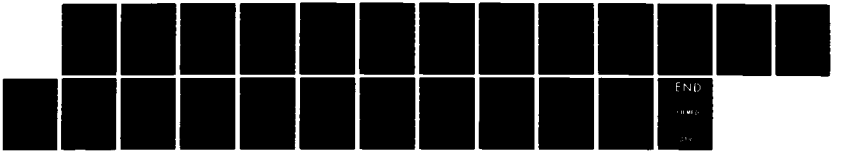
```
 90


  5    K1=K1+NDF
  6    J1=J1+NDF
C
C      COMPUTE STIFFNESS MATRIX OF THE FLUID PHASE
       DO 7 I=1,NPN
       II=I*NDF
       WII=SHP(3,I)*XPJ*COMP
       W1I=SHP(1,I)*XPJ*AK1
       W2I=SHP(2,I)*XPJ*AK2
       DO 7 J=I,NPN
       JJ=J*NDF
       CPP(I,J)=CPP(I,J)+WII*SHP(3,J)
  7    S(II,JJ)=S(II,JJ)+W1I*SHP(1,J)+W2I*SHP(2,J)
C
C      COMPUTE COUPLING STIFFNESS MATRIX
       II=1
       DO 9 I=1,NEL
       W1I=SS(1,I)*XDJ
       W2I=SS(2,I)*XDJ
       DO 8 J=1,NPN
       JJ=J*NDF
       S(II,JJ)=S(II,JJ)+W1I*SHP(3,J)
  8    S(II+1,JJ)=S(II+1,JJ)+W2I*SHP(3,J)
  9    II=II+NDF
  10   CONTINUE
       DO 11 J=1,NSL,NDF
       DO 11 K=J,NSL,NDF
       W11=S(J,K)
       W12=S(J,K+1)
       W21=S(J+1,K)
       W22=S(J+1,K+1)
       S(J,K)=W11*D(1)+W22*D(4)+W12*D(5)+W21*D(5)
       S(J,K+1)=W12*D(2)+W21*D(4)+W11*D(5)+W22*D(6)
       S(J+1,K)=W21*D(2)+W12*D(4)+W22*D(6)+W11*D(5)
       S(J+1,K+1)=W22*D(3)+W11*D(4)+W21*D(6)+W12*D(6)
       S(K,J)=S(J,K)
       S(K,J+1)=S(J+1,K)
       S(K+1,J)=S(J,K+1)
  11   S(K+1,J+1)=S(J+1,K+1)
       II=1
       DO 13 I=1,NEL
       DO 12 J=1,NPN
       JJ=J*NDF
       S(JJ,II)=S(II,JJ)
       S(JJ,II+1)=S(II+1,JJ)
       DO 12 K=1,NPN
       KK=K*NDF
       CPP(K,J)=CPP(J,K)
  12   S(KK,JJ)=S(JJ,KK)
  13   II=II+NDF
  14   NPL=NDF*NPN
       WRITE (15) NSL,((S(I,J),J=1 NSL),I=NDF,NPL,NDF),((CPP(I,J),J=1,NPN
```

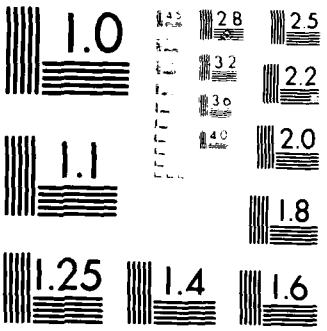
```
     1),I=1,NPN)
  15 RETURN
  16 CONTINUE
C
C     ....GET FLUID AND SOLID BODY FORCES
      AK2=D(25)
      ROWF=-D(26)
      DO 20 M=1,NGP
      DO 20 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XDJ,M,N,NGP)
      DO 17 I=1,NEL
  17  SS(3,I)=SHP(3,I)
      CALL SHAPE2 (XL,IX,NDM,NPN,XPJ,M,N,NGP)
      D2=XDJ*D(23)
      DO 19 I=1,NEL
      D1=D2*SS(3,I)
      DO 18 J=1,NDU
  18  FL(J,I)=FL(J,I)+D1*DC(J)
      IF(I.GT.NPN) GO TO 19
      FL(3,I)=FL(3,I)+AK2*ROWF*SHP(2,I)*XPJ
  19  CONTINUE
  20  CONTINUE
      RETURN
  21  CONTINUE
C
C     ....COMPUTE STRESSES AT NGP GAUSS PTS.
      Z=0.0D0
      DO 24 M=1,NGP
      DO 24 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XSJ,M,N,NGP)
      DO 22 I=1,3
  22  EPS(I)=0.0D0
      T=0.0D0
      X=0.0D0
      Y=0.0D0
      DO 23 J=1,NEL
      T=T+SHP(3,J)*TL(J)
      X=X+SHP(3,J)*XL(1,J)
      Y=Y+SHP(3,J)*XL(2,J)
C     EPS(1) = STRAIN IN  X-DIRECTION  , EPSILON-XX
C     EPS(2) = STRAIN IN XY-DIRECTION  ,   GAMMA-XY
C     EPS(3) = STRAIN IN  Y-DIRECTION  , EPSILON-YY
      EPS(1)=EPS(1)+SHP(1,J)*UL(1,J)
      EPS(3)=EPS(3)+SHP(2,J)*UL(2,J)
  23  EPS(2)=EPS(2)+SHP(2,J)*UL(1,J)+SHP(1,J)*UL(2,J)
      EPS(1)=EPS(1)-D(8)*T
      EPS(3)=EPS(3)-D(9)*T
C     SIG(1) = STRESS IN  X-DIRECTION     ,  SIGMA-XX
C     SIG(2) = STRESS IN XY-DIRECTION     ,    TAU-XY
C     SIG(3) = STRESS IN  Y-DIRECTION     ,  SIGMA-YY
C
C     ....CALCULATE EFFECTIVE STRESSES
```

```
      SIG(1)=D(1)*EPS(1)+D(2)*EPS(3)
      SIG(2)=D(4)*EPS(2)
      SIG(3)=D(2)*EPS(1)+D(3)*EPS(3)
      CALL PVALUE (SIG,1,2,X,Y,Z,LN,IL,TE,D)
C
C     ....CALCULATE TOTAL STRESSES
C     CALL SHAPE2(XL,IX,NDM,NPN,XSJ,M,N,NGP)
C     DO 49 J=1,NPN
C     SIG(1)=SIG(1)+SHP(3,J)*UL(3,J)
C     49 SIG(3)=SIG(3)+SHP(3,J)*UL(3,J)
C     CALL PVALUE(SIG,1,2,X,Y,Z,LN,IL,TE,D)
 24   CONTINUE
      RETURN
      END
```

AD-A151 922 A COMPUTER PROGRAM FOR CONSOLIDATION AND DYNAMIC RESPONSE ANALYSIS OF FLU..(U) OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS B L ABOUSTIT ET AL. JUN 83 2/2

UNCLASSIFIED OSURF-715107-84-5 AFOSR-TR-85-0266 F/G 8/13 NL





```
      SUBROUTINE GEOMBC (U,N,IBAND,MEQ,R,AK)
C     ---------------------------------------
C     ....THIS SUBROUTINE APPLIES THE KINEMATIC CONSTRAINTS FOR EVERY D.
C     ....CALLED BY TMSTP
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AK(MEQ,1), R(1)
      DO 2 M=2,IBAND
      K=N-M+1
      IF(K.LE.0) GO TO 1
      R(K)=R(K)-AK(K,M)*U
      AK(K,M)=0.0D0
 1    K=N+M-1
      IF(K.GT.MEQ) GO TO 2
      R(K)=R(K)-AK(N,M)*U
      AK(N,M)=0.0D0
 2    CONTINUE
      AK(N,1)=1.0D0
      R(N)=U
      RETURN
      END
```

```
      SUBROUTINE GABOSI (IX,XL,D,S,CPP,NDM,NDF,NST,NEL,NPN,NGP)
C     ------------------------------------------------------------
C
C
C     CALLS       SHAPEX
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      DIMENSION D(1), XL(NDM,1), S(NST,NST), IX(1), CPP(8,8), SPP(4,4),
     1SUU(12,12), SPU(12,4), LP(4), LD(8)
C
C     ....INITIALIZE STIFFNESS MATRICES
      NDU=NDF-1
      NPL=NPN*NDU
      NSL=NEL*NDU
      DO 3 I=1,NSL
      DO 1 J=1,NSL
 1    SUU(I,J)=0.0D0
      DO 2 K=1,NPN
 2    SPU(I,K)=0.0D0
 3    CONTINUE
      DO 4 I=1,NPN
      DO 4 J=1,NPN
 4    SPP(I,J)=0.0D0
C
C     ....COMPUTE ELEMENT STIFFNESS MATRIX BY(NGP*NGP)
      AK1=-D(24)
      AK2=-D(25)
      COMP=D(22)
      DO 10 M=1,NGP
      DO 10 N=1,NGP
      CALL SHAPEX (XL,IX,NDM,NEL,XDJ,M,N,NGP)
C
C     ....COMPUTE STIFFNESS MATRIX OF THE SOLID PHASE
C     COLUMN
      J1=1
      DO 6 J=1,NEL
      W1J=SHP(1,J)*XDJ
      W2J=SHP(2,J)*XDJ
      K1=J1
C     ROW
      DO 5 K=J,NEL
      SUU(J1,K1)=SUU(J1,K1)+W1J*SHP(1,K)
      SUU(J1,K1+1)=SUU(J1,K1+1)+W1J*SHP(2,K)
      SUU(J1+1,K1)=SUU(J1+1,K1)+W2J*SHP(1,K)
      SUU(J1+1,K1+1)=SUU(J1+1,K1+1)+W2J*SHP(2,K)
 5    K1=K1+NDU
 6    J1=J1+NDU
C
C     COMPUTE STIFFNESS MATRIX OF THE FLUID PHASE
      DO 7 I=1,NPN
      WII=COMP*SHP(3,I)*XDJ
      W1I=SHP(1,I)*XDJ*AK1
      W2I=SHP(2,I)*XDJ*AK2
```

```
      DO 7 J=I,NPN
      CPP(I,J)=CPP(I,J)+WII*SHP(3,J)
 7    SPP(I,J)=SPP(I,J)+W1I*SHP(1,J)+W2I*SHP(2,J)
C
C     COMPUTE COUPLING STIFFNESS MATRIX
      II=1
      DO 9 I=1,NEL
      W1I=SHP(1,I)*XDJ
      W2I=SHP(2,I)*XDJ
      DO 8 J=1,NPN
      SPU(II,J)=SPU(II,J)+W1I*SHP(3,J)
 8    SPU(II+1,J)=SPU(II+1,J)+W2I*SHP(3,J)
 9    II=II+NDU
 10   CONTINUE
      DO 11 J=1,NSL,NDU
      DO 11 K=J,NSL,NDU
      W11=SUU(J,K)
      W12=SUU(J,K+1)
      W21=SUU(J+1,K)
      W22=SUU(J+1,K+1)
      SUU(J,K)=W11*D(1)+W22*D(4)+W12*D(5)+W21*D(5)
      SUU(J,K+1)=W12*D(2)+W21*D(4)+W11*D(5)+W22*D(6)
      SUU(J+1,K)=W21*D(2)+W12*D(4)+W22*D(6)+W11*D(5)
      SUU(J+1,K+1)=W22*D(3)+W11*D(4)+W21*D(6)+W12*D(6)
      SUU(K,J)=SUU(J,K)
      SUU(K,J+1)=SUU(J+1,K)
      SUU(K+1,J)=SUU(J,K+1)
 11   SUU(K+1,J+1)=SUU(J+1,K+1)
      DO 12 I=1,NPN
      DO 12 J=1,NPN
      SPP(J,I)=SPP(I,J)
 12   CPP(J,I)=CPP(I,J)
C     ....STATIC CONDENSATION
C
      IF(NPN.EQ.NEL) GO TO 18
      INTER=NDU*(NEL-NPN)
      FAC=1.DO
      IF(COMP.NE.0.0D0) FAC=2.DO
      DO 17 N=1,INTER
      L=NSL-N
      M=L+1
      PIVOT=SUU(M,M)
      DO 13 I=1,NPN
      FP=FAC*SPU(M,I)/PIVOT
      DO 13 J=1,NPN
 13   CPP(I,J)=CPP(I,J)+FP*SPU(M,J)
      DO 16 J=1,L
      FU=SUU(M,J)/PIVOT
      DO 14 K=1,NPN
 14   SPU(J,K)=SPU(J,K)-FU*SPU(M,K)
      DO 15 I=J,L
      SUU(I,J)=SUU(I,J)-FU*SUU(I,M)
```

```
 15   SUU(J,I)=SUU(I,J)
 16   CONTINUE
 17   CONTINUE
 18   CONTINUE
C
C     ....RELOCATE ELEMENT MATRICES
C
      MM=0
      DO 19 I=NDU,NPL,NDU
      LD(I-1)=MM+1
      LD(I)=MM+2
 19   MM=MM+NDF
      DO 20 I=1,NPN
 20   LP(I)=NDF*I
      DO 21 I=1,NPL
      LI=LD(I)
      DO 21 J=1,NPL
      LJ=LD(J)
 21   S(LI,LJ)=SUU(I,J)
      DO 22 I=1,NPN
      LI=LP(I)
      DO 22 J=1,NPN
      LJ=LP(J)
 22   S(LI,LJ)=SPP(I,J)
      DO 23 I=1,NPL
      LI=LD(I)
      DO 23 J=1,NPN
      LJ=LP(J)
      S(LI,LJ)=SPU(I,J)
 23   S(LJ,LI)=SPU(I,J)
      RETURN
      END
```

```
      SUBROUTINE SHAPEX (XL,IX,NDM,NEL,XSJ,M,N,NPT)
C     ------------------------------------------------
C
C     CALLED BY      GBABOSI
C     SHAPE FUNCTION ROUTINE FOR 2-D ISOPARAMETRIC ELEMENTS
C     CALLED BY SUBROUTINE ELEMNT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      DIMENSION IX(1), XL(NDM,1), XS(3,2)
      X=GPT(M,NPT)
      E=GPT(N,NPT)
C
C     FORM 4-NODE QUADRILATERAL SHAPE FUNCTION
      DO 1 I=1,4
      SHP(3,I)=(1.+LX(I)*X)*(1.+LE(I)*E)/4.
      SHP(1,I)=LX(I)*(1.+LE(I)*E)/4.
 1    SHP(2,I)=LE(I)*(1.+LX(I)*X)/4.
C
C     ADD GHABOUSSI INCOMPATIBLE SHAPE FUNCTIONS
      SHP(3,5)=1.D0-X*X
      SHP(1,5)=-2.D0*X
      SHP(2,5)=0.0D0
      SHP(3,6)=1.D0-E*E
      SHP(1,6)=0.0D0
      SHP(2,6)=-2.D0*E
C
C     CONSTRUCT JACOBIAN XJ AND DETERMINANT OF XJ
      DO 2 I=1,NDM
      DO 2 J=1,2
      XS(I,J)=0.0D0
      DO 2 K=1,4
 2    XS(I,J)=XS(I,J)+XL(I,K)*SHP(J,K)
C
C
      DETJ=XS(1,1)*XS(2,2)-XS(1,2)*XS(2,1)
C     COMPUTE DNI/DX AND STORE IN SHP(1,I)
C     COMPUTE DNI/DY AND STORE IN SHP(2,I)
      DO 3 I=1,NEL
      TEMP=(XS(2,2)*SHP(1,I)-XS(2,1)*SHP(2,I))/DETJ
      SHP(2,I)=(-XS(1,2)*SHP(1,I)+XS(1,1)*SHP(2,I))/DETJ
 3    SHP(1,I)=TEMP
C
      XSJ=DETJ*WT(M,NPT)*WT(N,NPT)
C
      RETURN
      END
```

```
      SUBROUTINE DYNAMC (X,T,IX,IE,ID,IDL,IXL,KPRS,XL,DL,SK,UL,FL,TL,
C     ------------------------------------------------------------------
     1UO,F,R,D,AK,TIMES,NDT,NPRINT,THETA,GAMA,BETA,VO,AO,U1,V1,A1,
     2NEN,NEN1,NDF,NDM,MEQ,IBAND,NUMEL,NUMNP,NSTEPS,NST,NRD,BND)
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,TWR
      LOGICAL BACK,FORW,BND,SIG,TWR
      DIMENSION X(NDM,1), T(1), IX(NEN1,1), IE(4,1), ID(NDF,NUMNP), IDL(
     11), IXL(1), KPRS(1), XL(1), DL(1), SK(NST,1), UL(1), FL(1), TL(1),
     2 UO(NDF,1), F(NDF,1), D(1), AK(MEQ,IBAND), R(NDF,NUMNP), TIMES(1),
     3 NDT(1), NPRINT(1), THETA(1), GAMA(1), BETA(1), VO(NDF,1), AO(NDF,
     41), U1(NDF,1), V1(NDF,1), A1(NDF,1), COEF(10), TRECD(20), ACEL(20)
     5, IACEL(4)
C
C     FILE ASSIGNMENT
C     15 - ELEMENT STIFFNESS
C     16 - GLOBAL STIFFNESS (MEQ,IBAND)
C     17 - GLOBAL INITIAL FORCE (NDF,NUMNP)
C
      SIG=.FALSE.
C
C     INPUT INITIAL CONDITIONS
C
      READ 9, USO,VSO,UFO,VFO
      PRINT 10, USO,VSO,UFO,VFO
      NDF2=NDF/2
      DO 2 I=1,NUMNP
      DO 1 K=1,NDF2
      L=NDF-K+1
      UO(K,I)=USO
      UO(L,I)=UFO
      VO(K,I)=VSO
 1    VO(L,I)=VFO
      DO 2 J=1,NDF
      IF(ID(J,I).EQ.0) GO TO 2
      UO(J,I)=F(J,I)
      VO(J,I)=0.0D0
 2    CONTINUE
      READ 11, LOAD,OMEGA,AZERO,BZERO,CZERO
      PRINT 12, LOAD,OMEGA,AZERO,BZERO,CZERO
      IF(LOAD.EQ.0) GO TO 3
      READ 13, LACEL,(IACEL(I),I=1,NDF)
      PRINT 14, LACEL
      PRINT 15
      PRINT 16, (IACEL(I),I=1,NDF)
      IF(LACEL.EQ.0) GO TO 3
      READ 17, NTAPE,NDTACL
      PRINT 18, NTAPE,NDTACL
      READ (NTAPE) (TRECD(I),ACEL(I),I=1,NDTACL)
      PRINT 19, (ACEL(I),I=1,NDTACL)
 3    CONTINUE
```

```
C
C      ...INPUT TIME FACTORS
       READ 20, (TIMES(I),THETA(I),GAMA(I),BETA(I),NDT(I),NPRINT(I),KPRS(
      1I),I=1,NSTEPS)
       PRINT 21
       PRINT 22, (I,TIMES(I),THETA(I),GAMA(I),BETA(I),NDT(I),NPRINT(I),KP
      1RS(I),I=1,NSTEPS)
C
C      ....STORE THE GLOBAL STIFFNESS MATRIX TO BE UPDATED
C      EVERY STEP AND THE INITIAL LOAD VECTOR
       WRITE (16) AK,R
C
C      DETERMINE THE INITIAL ACCELERATION FROM THE EQUATION OF MOTION
C
       CALL INACL (AK,UO,VO,AO,R,F,SK,IX,ID,NEN,NEN1,NST,MEQ,IBAND,NUMNP,
      1NUMEL,NDF,BZERO)
C      CALL ZERO(AO,NDF*NUMNP)
C
C
C      ....DYNAMIC COMPUTATIONS
       DO 8 LT=1,NSTEPS
       NO=NPRINT(LT)
       NDTS=NDT(LT)
       LT1=LT-1
       STEP=TIMES(LT)
       IF(LT.GT.1) STEP=TIMES(LT)-TIMES(LT1)
       DT=STEP/NDT(LT)
C
C      CALCULATE THE DYNAMIC STIFFNESS MATRIX, APPLY GEOMETRIC BOUNDARY
C      CONDITIONS, AND SAVE R (LOAD VECTOR)
       CALL STEPS (THETA(LT),GAMA(LT),BETA(LT),DT,R,F,AK,SK,ID,COEF,MEQ,I
      1BAND,NUMEL,NUMNP,NDF,NST)
C
C      ....STORE THE GEOMETRICALLY MODEFIED GLOBAL LOAD
C      VECTOR TO BE UBDATED IN SUBROUTINE INCRS
       REWIND 17
       WRITE (17) R
C
C      TRAINGULIZE THE STIFFNESS MATRIX
       BACK=.FALSE.
       FORW=.TRUE.
       CALL BANSOL (AK,R,R,MEQ,IBAND,BACK,FORW)
C
C      UPDATE THE LOAD VECTOR R, FOR EVERY TIME INCREMENT
C
       DO 7 LDT=1,NDTS
       TIME=TIMES(LT)+LDT*DT-NDTS*DT
       CALL INCRS (THETA(LT),GAMA(LT),BETA(LT),DT,R,UO,VO,AO,U1,V1,SK,ID,
      1IX,NDF,NUMNP,NUMEL,NST,NEN,NEN1,F,A1,OMEGA,AZERO,BZERO,CZERO,TIME,
      2LOAD,LACEL,IACEL,TRECD,ACEL,NDTACL)
C
C      SOLVE FOR DISPLACEMENTS(R) AT TO+THETA*DT
```

```
100
C
      BACK=.TRUE.
      CALL BANSOL (AK,R,R,MEQ,IBAND,BACK,FORW)
C
C     OBTAIN DISPLACEMENT, VEL. AND ACC. AT THE END OF AN INCREMENT
C     (IE. AT TO+DT)
C
      DO 4 I=1,NUMNP
      DO 4 J=1,NDF
      U1(J,I)=R(J,I)*COEF(1)+UO(J,I)*COEF(2)+VO(J,I)*COEF(3)+AO(J,I)*COE
     1F(4)
      V1(J,I)=(R(J,I)-UO(J,I))*COEF(5)+VO(J,I)*COEF(6)+AO(J,I)*COEF(7)
 4    A1(J,I)=(R(J,I)-UO(J,I))*COEF(8)-VO(J,I)*COEF(9)+AO(J,I)*COEF(10)
C
C     OBTAIN INITIAL CONDITION FOR NEXT INCREMENT
C
      DO 5 I=1,NUMNP
      DO 5 J=1,NDF
      UO(J,I)=U1(J,I)
      VO(J,I)=V1(J,I)
      AO(J,I)=A1(J,I)
 5    CONTINUE
C     CALL ZERO(AO,NDF*NUMNP)
C     CALL ZERO(VO,NDF*NUMNP)
      IF(LDT.EQ.NO.OR.LDT.EQ.NDTS) GO TO 6
      GO TO 7
 6    CALL PRTDIS (ID,X,U1,F,NDM,NDF,.FALSE.,TWR,.TRUE.,LT,LDT,TIME,BND)
C
C     *IN CASE THE VELOCITY (V1) IS REQUIRED TO BE PRINTED,
C     *REMOVE THE COMMECENT (C) ON THE FOLLOWING TWO CARDS
C     CALL PRTDIS(ID,X,V1,F,NDM,NDF,.FALSE.,TWR,.TRUE.,LT,LDT,TIME,
C     1            BND)
      NO=NPRINT(LT)+NO
      IF(KPRS(LT).EQ.0) GO TO 7
      LINE=0
      CALL FORMEL (X,XL,D,DL,SPU,UL,FL,TL,U1,F,T,IX,IE,ID,IDL,IXL,IXL,ND
     1F,NDM,NEN1,NST,5,AK,AK,R,SIG,SIG,SIG,SIG,SIG,NRD,NOW,IBAND,BND)
 7    CONTINUE
 8    CONTINUE
      RETURN
C
 9    FORMAT (4F10.0)
 10   FORMAT (1H1/10X,'INITIAL CONDITIONS.. '//10X,'SOLID INITIAL DISPLA
     1CEMENTS = ',E10.3/10X,'SOLID INITIAL VELOCITY      = ',E10.3/10X,'
     2FLUID INITIAL DISPLACEMENT  = ',E10.3/10X,'FLUID INITIAL VELOCITY
     3     = ',E10.3///)
 11   FORMAT (I5,4F10.0)
 12   FORMAT (10X,'LOADING CRITERION'//10X,'LOAD =0 TRACTION ONLY'/10X,'
     1LOAD =1 ACCELERATION ONLY'/10X,'LOAD =2 TRACTION AND ACCEL'/10X,'I
     2N THIS CASE LOAD =',I5//10X,'LOADING FUNCTION IS'/10X,'AZERO+BZERO
     3*COS(WT)+CZERO*SIN(WT) WHERE,'/10X,'OMEGA=',E15.4,'AZERO=',E15.4,'
     4BZERO=',E15.4,'CZERO=',E15.4//)
```

```
13    FORMAT (5I5)
14    FORMAT (10X,'ACCELERATION INPUT'/10X,'LACEL =0 LOADING FUNCTION'/1
     10X,'LACEL =1 ACCELEGRAM RECORD'/10X,'IN THIS CASE LACEL =',I5)
15    FORMAT (10X,'ACCELERATION DEGREES OF FREEDOM')
16    FORMAT (10X,4I5)
17    FORMAT (2I5)
18    FORMAT (//10X,'TAPE NUMBER FOR ACCELEGRAM RECORDS=',I5/10X,'NUMBER
     1 OF ACCELEGRAM RECORDS =',I5///10X,'SUPPORT ACCELERACTION')
19    FORMAT (10X,E10.3)
20    FORMAT (4F10.0,3I5)
21    FORMAT (3X,'STEP NO',2X,'END TIME',5X,'THETA',6X,'GAMA',6X,'BETA',
     11X,'NO OF INC',5X,'PRINT',4X,'STRESS'/)
22    FORMAT (I10,4D10.3,3I10)
      END
```

```
      SUBROUTINE STEPS (THETA,GAMA,BETA,DT,R,F,AK,SK,ID,COEF,MEQ,
C     ------------------------------------------------------------
     1IBAND,NUMEL,NUMNP,NDF,NST)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AK(MEQ,IBAND), R(NDF,NUMNP), ID(NDF,NUMNP), F(NDF,1), SK
     1(NST,NST), SM(32,32), SD(32,32), LP(32), COEF(10)
      REWIND 16
      READ (16) AK,R
C
C     ASSEMBLE THE DYNAMIC STIFFNESS MATRIX
      C1=BETA*THETA*THETA*DT*DT
      C2=BETA*THETA*DT/GAMA
      REWIND 15
      DO 3 M=1,NUMEL
      READ (15) NSL,((SK(I,J),J=1,NSL),I=1,NSL),((SM(I,J),J=1,NSL),I=1,N
     1SL),((SD(I,J),J=1,NSL),I=1,NSL),(LP(I),I=1,NSL)
      DO 2 I=1,NSL
      II=LP(I)
      DO 1 J=1,NSL
      JJ=LP(J)
      KK=JJ-II+1
      IF(KK.LE.0) GO TO 1
      AK(II,KK)=AK(II,KK)+SM(I,J)/C1+SD(I,J)/C2
 1    CONTINUE
 2    CONTINUE
 3    CONTINUE
C
C     INTRODUCE KINEMATIC CONSTRAINTS
      DO 5 M=1,NUMNP
      DO 4 J=1,NDF
      IF(ID(J,M).EQ.0) GO TO 4
      IDF=(M-1)*NDF+J
      CALL GEOMBC (F(J,M),IDF,IBAND,MEQ,R,AK)
 4    CONTINUE
 5    CONTINUE
C
C     OBTAIN TIME DOMAIN INTEGRATION FACTORS
      T1=THETA
      T2=T1*T1
      T3=T2*T1
      COEF(1)=1.DO/T3
      COEF(2)=1.DO-COEF(1)
      COEF(3)=(1.DO-1.DO/T2)*DT
      COEF(4)=(1.DO-1.DO/T1)*DT*DT/2.DO
      COEF(5)=GAMA/(BETA*DT*T3)
      COEF(6)=1.DO-GAMA/(BETA*T2)
      COEF(7)=(1.DO-GAMA/(BETA*T1*2.DO))*DT
      COEF(8)=COEF(5)/(DT*GAMA)
      COEF(9)=COEF(5)*T1/GAMA
      COEF(10)=1.DO-1.DO/(BETA*T1*2.DO)
      RETURN
      END
```

```
      SUBROUTINE INCRS (THETA,GAMA,BETA,DT,R,UO,VO,AO,AN,BN,SK,
C     ------------------------------------------------------------
     1ID,IX,NDF,NUMNP,NUMEL,NST,NEN,NEN1,F,ABASE,OMEGA,AZERO,
     2BZERO,CZERO,TIME,LOAD,LACEL,IACEL,TRECD,ACEL,NDTACL)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION R(NDF,NUMNP), UO(NDF,1), VO(NDF,1), AO(NDF,1), AN(NDF,1)
     1, BN(NDF,1), ID(NDF,NUMNP), SK(NST,NST), F(NDF,1), ABASE(NDF,1), I
     2ACEL(4), TRECD(20), ACEL(20), IX(NEN1,1), SM(32,32), SD(32,32), LP
     3(32)
      REWIND 17
      READ (17) R
      CALL ZERO (ABASE,NDF*NUMNP)
C
C     LOADING FUNCTION AT TO+THETA*DT
      T=TIME+(THETA-1.D0)*DT
      ARGUM=OMEGA*T
      FACT=AZERO+BZERO*DCOS(ARGUM)+CZERO*DSIN(ARGUM)
      IF(LOAD.EQ.0) GO TO 2
C
C     ACCELERATION LOADING
      IF(LACEL.NE.0) CALL RECORD (T,TRECD,ACEL,NDTACL,ACLRCD)
      DO 1 I=1,NUMNP
      DO 1 J=1,NDF
      IF(ID(J,I).NE.0) GO TO 1
      IF(IACEL(J).EQ.0) GO TO 1
C     HARMONIC ACCELERACTION
      IF(LACEL.EQ.0) ABASE(J,I)=FACT
C     ACCELEGRAM RECORDS ACCELERACTION
      IF(LACEL.NE.0) ABASE(J,I)=ACLRCD
 1    CONTINUE
      IF(LOAD.EQ.1) GO TO 4
 2    CONTINUE
C
C     ....ADD EXTERNALLY APPLIED CONC. NODAL LOADS TO R
      DO 3 N=1,NUMNP
      DO 3 J=1,NDF
      IF(ID(J,N).NE.0) GO TO 3
      R(J,N)=R(J,N)+FACT*F(J,N)
 3    CONTINUE
 4    CONTINUE
      X=THETA*DT
      XX=X*X
      Y=.5D0-BETA
      YY=1.D0-BETA/GAMA
      ZZ=.5D0-BETA/GAMA
      DO 5 I=1,NUMNP
      DO 5 J=1,NDF
      AN(J,I)=(UO(J,I)+X*VO(J,I)+Y*XX*AO(J,I))/(BETA*XX)
 5    BN(J,I)=(UO(J,I)+YY*X*VO(J,I)+ZZ*XX*AO(J,I))*GAMA/(BETA*X)
      REWIND 15
      DO 9 M=1,NUMEL
```

```
      READ (15) NSL,((SK(I,J),J=1,NSL),I=1,NSL),((SM(I,J),J=1,NSL),I=1,N
     1SL),((SD(I,J),J=1,NSL),I=1,NSL),(LP(I),I=1,NSL)
      NEL=NSL/NDF
      DO 8 I=1,NEL
      J=IX(I,M)
      DO 7 IDF=1,NDF
      II=(I-1)*NDF+IDF
      IF(ID(IDF,J).NE.0) GO TO 7
      DO 6 K=1,NEL
      N=IX(K,M)
      DO 6 L=1,NDF
      KK=(K-1)*NDF+L
    6 R(IDF,J)=R(IDF,J)+SM(II,KK)*(AN(L,N)-ABASE(L,N))+SD(II,KK)*BN(L,N)
    7 CONTINUE
    8 CONTINUE
    9 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INACL (AM,UO,VO,AO,R,F,SK,IX,ID,NEN,NEN1,NST,MEQ,
C     ------------------------------------------------------------
     1IBAND,NUMNP,NUMEL,NDF,BZERO)
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL BACK,FROW
      DIMENSION AM(MEQ,IBAND), UO(NDF,1), VO(NDF,1), AO(NDF,1), R(NDF,1)
     1, F(NDF,1), SK(NST,NST), SM(32,32), SD(32,32), LP(32), IX(NEN1,1),
     2 ID(NDF,1)
C
C     THIS SUBROUTINE ASSEMPLE THE MASS MATRIX AND SOLVE THE
C     EQUATION OF MOTION FOR THE INITIAL ACCELERATION
      CALL ZERO (AM,IBAND*MEQ)
      CALL ZERO (AO,NDF*NUMNP)
      REWIND 15
      DO 6 M=1,NUMEL
      READ (15) NSL,((SK(I,J),J=1,NSL),I=1,NSL),((SM(I,J),J=1,NSL),I=1,N
     1SL),((SD(I,J),J=1,NSL),I=1,NSL),(LP(I),I=1,NSL)
      DO 2 I=1,NSL
      II=LP(I)
      DO 1 J=1,NSL
      JJ=LP(J)
      KK=JJ-II+1
      IF(KK.LE.0) GO TO 1
      AM(II,KK)=AM(II,KK)+SM(I,J)
 1    CONTINUE
 2    CONTINUE
C
C     OBTAIN THE RIGHT HAND SIDE
      NEL=NSL/NDF
      DO 5 I=1,NEL
      J=IX(I,M)
      DO 4 IDF=1,NDF
      II=(I-1)*NDF+IDF
      DO 3 K=1,NEL
      N=IX(K,M)
      DO 3 L=1,NDF
      KK=(K-1)*NDF+L
 3    AO(IDF,J)=AO(IDF,J)+SK(II,KK)*UO(L,N)+SD(II,KK)*VO(L,N)
 4    CONTINUE
 5    CONTINUE
 6    CONTINUE
C
C     INTRODUCE TRACTION AT TIME=0.
      DO 8 M=1,NUMNP
      DO 8 J=1,NDF
      IF(ID(J,M).NE.0) GO TO 7
      AO(J,M)=R(J,M)-AO(J,M)+BZERO*F(J,M)
      GO TO 8
 7    AO(J,M)=R(J,M)-AO(J,M)
 8    CONTINUE
C
C     INTRODUCE CONSTRAINT ACCELERATIONS
```

```
      ASUPR=0.0D0
      DO 10 M=1,NUMNP
      DO 9 J=1,NDF
      IF(ID(J,M).EQ.0) GO TO 9
      IDF=(M-1)*NDF+J
      CALL GEOMBC (ASURP,IDF,IBAND,MEQ,AO,AM)
 9    CONTINUE
 10   CONTINUE
C
C     TRIANGULIZE TH MASS MATRIX
      BACK=.FALSE.
      FROW=.TRUE.
      CALL BANSOL (AM,AO,AO,MEQ,IBAND,BACK,FROW)
C
C     SOLVE FOR INITIAL ACCEL
      BACK=.TRUE.
      CALL BANSOL (AM,AO,AO,MEQ,IBAND,BACK,FROW)
      RETURN
      END
```

```
      SUBROUTINE RECORD (TIME,TRECD,ACEL,NDTACL,ACLRCD)
C     ------------------------------------------------
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION TRECD(20), ACEL(20)
C     THIS SUBROUTINE INTERPOLATE ACELOGRAM DATA
C
      ACLRCD=0.0D0
      RATIO=0.0D0
      DO 1 I=2,NDTACL
      IF(TIME.GE.TRECD(I)) GO TO 2
 1    CONTINUE
      RETURN
 2    DTRCD=TRECD(I)-TRECD(I-1)
      IF(DTRCD.EQ.0.0D0) GO TO 3
      RATIO=(TIME-TRECD(I-1))/DTRCD
 3    ACLRCD=ACEL(I-1)*(1.D0-RATIO)+ACEL(I)*RATIO
      RETURN
      END
```

```
      SUBROUTINE ELMT02 (IX,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,
C     ------------------------------------------------------------
     1IL,LN,NG)
C
C     CALLED BY
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,EWR
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      DIMENSION D(1), XL(NDM,1), FL(NDF,1), S(NST,NST), UL(NDF,1), TL(1)
     1, IX(1), SM(32,32), SD(32,32), LP(32)
C
      NGP=NG
      GO TO (1,1,4,5,6), ISW
C
C     COMPUTE ELEMNT STIFFNESS, MASS AND DAMPING MATRICES
 1    CONTINUE
      NSL=NEL*NDF
C     ELEMENT LENGTH
      H=XL(NDM,2)-XL(NDM,1)
C     SOLID STIFFNESS
      F=(D(1)+D(19)*D(19)*D(18))/H
      S(1,1)=F
      S(3,3)=F
      S(1,3)=-F
C     SOLID MASS
      F=D(23)*H/3.DO
      SM(1,1)=F
      SM(3,3)=F
      SM(1,3)=F/2.DO
C     COUPLING STIFFNESS
      F=D(19)*D(18)/H
      S(1,2)=F
      S(1,4)=-F
      S(2,3)=-F
      S(3,4)=F
C     COUPLING MASS
      F=D(26)*H/(D(22)*3.DO)
      SM(1,2)=F
      SM(1,4)=F/2.DO
      SM(2,3)=F/2.DO
      SM(3,4)=F
C     FLUID STIFFNESS
      F=D(18)/H
      S(2,2)=F
      S(4,4)=F
      S(2,4)=-F
C     FLUID MASS
      F=D(26)*H/(D(22)*D(22)*3.DO)
      SM(2,2)=F
      SM(4,4)=F
      SM(2,4)=F/2
```

```
      DO 2 I=1,NSL
      DO 2 J=1,NSL
      S(J,I)=S(I,J)
      SM(J,I)=SM(I,J)
 2    SD(I,J)=0.0D0
C     FLUID DAMPING
      F=H/(D(24)*3.D0)
      SD(2,2)=F
      SD(4,4)=F
      SD(2,4)=F/2.D0
      SD(4,2)=F/2.D0
C     SOLID DAMPING
      F1=D(21)
      F2=D(22)*D(22)*D(21)
      F3=D(20)
      F4=D(20)*D(19)*D(19)
      SD(1,1)=F1*SM(1,1)-F2*SM(2,2)+F3*S(1,1)-F4*S(2,2)
      SD(3,3)=F1*SM(3,3)-F2*SM(4,4)+F3*S(3,3)-F4*S(4,4)
      SD(1,3)=F1*SM(1,3)-F2*SM(2,4)+F3*S(1,3)-F4*S(2,4)
      SD(3,1)=SD(1,3)
      DO 3 I=1,NEL
      II=2*I
      LP(II)=2*IX(I)
 3    LP(II-1)=LP(II)-1
      WRITE (15) NSL,((S(I,J),J=1,NSL),I=1,NSL),((SM(I,J),J=1,NSL),I=1,N
     1SL),((SD(I,J),J=1,NSL),I=1,NSL),(LP(I),I=1,NSL)
      RETURN
 4    RETURN
 5    CONTINUE
C     GET FLUID AND SOLID BODY FORCES
      H=XL(NDM,2)-XL(NDM,1)
      RHO=D(23)
      RHOF=D(26)
      FL(1,1)=H*RHO*DC(2)/2.D0
      FL(2,1)=H*RHOF*DC(2)/2.D0
      FL(1,2)=FL(1,1)
      FL(2,2)=FL(2,1)
      RETURN
 6    CONTINUE
C     CALCULATE EFFECTIVE,TOTAL STRESSES AND PORE PRESSURES AT
C     ELEMENT CENTRID
      H=XL(NDM,2)-XL(NDM,1)
      XC=XL(NDM,1)+H/2.D0
C     STRAIN IN X-DIR
      EPS=(UL(1,2)-UL(1,1))/H
C     FLUID VOL STRAIN
      ZETA=(UL(2,2)-UL(2,1))/H
C     PORE PRESSURE
      P=D(18)*(D(19)*EPS+ZETA)
C     EFFECTIVE STRESS IN X-DIR
      SIG=EPS*D(1)+(D(19)-1.D0)*P
C     TOTAL STRESS
```

```
110

      SIGT=SIG+D(19)*P
      IF(LINE.NE.0) GO TO 7
      WRITE (6,10) HEAD
      PRINT 8
 7    LINE=LINE+1
      IF(LINE.EQ.50) LINE=0
      PRINT 9, LN,XC,SIG,P,SIGT
      RETURN
C
 8    FORMAT (6X,'EL',8X,'X-COORD',6X,'EFF-STRES',7X,'POR-PRES',6X,'TT
     1L-STRES'/)
 9    FORMAT (5X,I5,4E15.4)
 10   FORMAT (1H1,5X,20A4/5X,'....ELEMENT  STRESS ....','CCW ROTATION IS
     1 POSITIVE....'/5X,'....1-DIRECTION IS RADIAL AND 2-DIRECTION IS AX
     2IAL FOR AN',1X,'AXISYMMETRIC PROBLEM....'//)
      END
```

```
      SUBROUTINE ELMT03 (IX,XL,D,FL,S,UL,TL,NDM,NDF,NST,NEL,ISW,
C     -------------------------------------------------------------
     1IL,LN,NGP)
C
C     CALLED BY ELMLIB
C     CALLS       SHAPE2,PVALUE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,MEQ,NTAPE,NPN
      COMMON /ANGLE/ ANG(4),DC(3),ST(2),IEPS,LINE,EWR
      COMMON /GQDRT/ GPT(4,4),WT(4,4),SHP(4,9),LX(4),LE(4)
      COMMON /STRES/ EPS(6),SIG(6)
      DIMENSION D(1), XL(NDM,1), FL(NDF,1), S(NST,NST), UL(NDF,1), TL(1)
     1, IX(1), SM(32,32), SD(32,32), LP(32)
C
      GO TO (1,1,8,9,11), ISW
 1    CONTINUE
C
C     ....COMPUTE ELEMENT STIFFNESS, MSASS AND DAMPING MATRICES BY(NGP*N
      NSL=NEL*NDF
      DO 2 I=1,NSL
      DO 2 J=1,NSL
      SD(I,J)=0.0D0
 2    SM(I,J)=0.0D0
      DO 5 M=1,NGP
      DO 5 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XDJ,M,N,NGP)
C
C     COLUMN
      J1=1
      DO 4 J=1,NEL
      W1J=SHP(1,J)*XDJ
      W2J=SHP(2,J)*XDJ
      W3J=SHP(3,J)*XDJ
      K1=J1
C     ROW
      DO 3 K=J,NEL
      S(J1,K1)=S(J1,K1)+W1J*SHP(1,K)
      S(J1,K1+1)=S(J1,K1+1)+W1J*SHP(2,K)
      S(J1+1,K1)=S(J1+1,K1)+W2J*SHP(1,K)
      S(J1+1,K1+1)=S(J1+1,K1+1)+W2J*SHP(2,K)
      SM(J1,K1)=SM(J1,K1)+W3J*SHP(3,K)
 3    K1=K1+NDF
 4    J1=J1+NDF
 5    CONTINUE
      D1=D(1)+D(19)*D(19)*D(18)
      D2=D(2)+D(19)*D(19)*D(18)
      D3=D(3)+D(19)*D(19)*D(18)
      F1=D(21)
      F2=D(21)*D(22)*D(22)
      F3=D(20)
      F4=D(20)*D(19)*D(19)
C
```

```
      DO 6 J=1,NSL,NDF
      DO 6 K=J,NSL,NDF
      W11=S(J,K)
      W12=S(J,K+1)
      W21=S(J+1,K)
      W22=S(J+1,K+1)
      A11=W11*D(18)
      A12=W12*D(18)
      A21=W21*D(18)
      A22=W22*D(18)
      B11=A11*D(19)
      B12=A12*D(19)
      B21=A21*D(19)
      B22=A22*D(19)
      DFF=SM(J,K)
      WSS=DFF*D(23)
      WFF=DFF*D(26)/(D(22)*D(22))
      WSF=DFF*D(26)/D(22)
C
C     SOLID STIFFNESS
      S(J,K)=W11*D1+W22*D(4)+W12*D(5)+W21*D(5)
      S(J,K+1)=W12*D2+W21*D(4)+W11*D(5)+W22*D(6)
      S(J+1,K)=W21*D2+W12*D(4)+W22*D(6)+W11*D(5)
      S(J+1,K+1)=W22*D3+W11*D(4)+W21*D(6)+W12*D(6)
      S(K,J)=S(J,K)
      S(K,J+1)=S(J+1,K)
      S(K+1,J)=S(J,K+1)
      S(K+1,J+1)=S(J+1,K+1)
C
C     COUPLING STIFFNESS
      S(J,K+2)=B11
      S(J,K+3)=B12
      S(J+1,K+2)=B21
      S(J+1,K+3)=B22
      S(J+2,K)=B11
      S(J+2,K+1)=B12
      S(J+3,K)=B21
      S(J+3,K+1)=B22
      S(K+2,J)=B11
      S(K+3,J)=B12
      S(K+2,J+1)=B21
      S(K+3,J+1)=B22
      S(K,J+2)=B11
      S(K+1,J+2)=B12
      S(K,J+3)=B21
      S(K+1,J+3)=B22
C
C     FLUID STIFFNESS
      S(J+2,K+2)=A11
      S(J+2,K+3)=A12
      S(J+3,K+2)=A21
      S(J+3,K+3)=A22
```

```
      S(K+2,J+2)=A11
      S(K+3,J+2)=A12
      S(K+2,J+3)=A21
      S(K+3,J+3)=A22
C
C     SOLID MASS
      SM(J,K)=WSS
      SM(J+1,K+1)=WSS
      SM(K,J)=WSS
      SM(K+1,J+1)=WSS
C
C     COUPLING MASS
      SM(J+1,K+3)=WSF
      SM(J+2,K)=WSF
      SM(J+3,K+1)=WSF
      SM(K+2,J)=WSF
      SM(K+3,J+1)=WSF
      SM(K,J+2)=WSF
      SM(K+1,J+3)=WSF
      SM(J,K+2)=WSF
C
C     FLUID MASS
      SM(J+2,K+2)=WFF
      SM(J+3,K+3)=WFF
      SM(K+2,J+2)=WFF
      SM(K+3,J+3)=WFF
C
C     FLUID DAMPING
      SD(J+2,K+2)=DFF/D(24)
      SD(J+3,K+3)=DFF/D(25)
      SD(K+2,J+2)=SD(J+2,K+2)
      SD(K+3,J+3)=SD(K+3,J+3)
C
C     SOLID DAMPING
      SD(J,K)=F1*WSS-F2*WFF+F3*S(J,K)-F4*S(J+2,K+2)
      SD(J,K+1)=F1*WSS-F2*WFF+F3*S(J,K+1)-F4*S(J+2,K+3)
      SD(J+1,K)=F1*WSS-F2*WFF+F3*S(J+1,K)-F4*S(J+3,K+2)
      SD(J+1,K+1)=F1*WSS-F2*WFF+F3*S(J+1,K+1)-F4*S(J+3,K+3)
      SD(K,J)=SD(J,K)
      SD(K+1,J)=SD(J,K+1)
      SD(K,J+1)=SD(J+1,K)
    6 SD(K+1,J+1)=SD(J+1,K+1)
      DO 7 I=1,NEL
      II=NDF*I
      LP(II)=NDF*IX(I)
      LP(II-1)=LP(II)-1
      LP(II-2)=LP(II)-2
    7 LP(II-3)=LP(II)-3
      WRITE (15) NSL,((S(I,J),J=1,NSL),I=1,NSL),((SM(I,J),J=1,NSL),I=1,N
     1SL),((SD(I,J),J=1,NSL),I=1,NSL),(LP(I),I=1,NSL)
    8 RETURN
    9 CONTINUE
```

```
C
C     ....GET FLUID AND SOLID BODY FORCES
      DO 10 M=1,NGP
      DO 10 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XDJ,M,N,NGP)
      D2=XDJ*D(23)
      D4=XDJ*D(26)/D(22)
      DO 10 I=1,NEL
      D1=D2*SHP(3,I)
      D3=D4*SHP(3,I)
      DO 10 J=1,2
      FL(J,I)=FL(J,I)+D1*DC(J)
 10   FL(J+2,I)=FL(J+2,I)+D3*DC(J)
      RETURN
 11   CONTINUE
C
C     ....COMPUTE STRESSES AT NGP GAUSS PTS.
      Z=0.0D0
      DO 14 M=1,NGP
      DO 14 N=1,NGP
      CALL SHAPE2 (XL,IX,NDM,NEL,XSJ,M,N,NGP)
      DO 12 I=1,3
 12   EPS(I)=0.0D0
      ZETA=0.0D0
      X=0.0D0
      Y=0.0D0
      DO 13 J=1,NEL
      X=X+SHP(3,J)*XL(1,J)
      Y=Y+SHP(3,J)*XL(2,J)
C     EPS(1) = SOLID STRAIN IN  X-DIRECTION  , EPSILON-XX
C     EPS(2) = SOLID STRAIN IN XY-DIRECTION  ,   GAMMA-XY
C     EPS(3) = SOLID STRAIN IN  Y-DIRECTION  , EPSILON-YY
C     EPSV= SOLID VOLUMETRIC STRAIN
C     ZETA= FLUID VOLUMETRIC STRAIN
      EPS(1)=EPS(1)+SHP(1,J)*UL(1,J)
      EPS(3)=EPS(3)+SHP(2,J)*UL(2,J)
      EPS(2)=EPS(2)+SHP(2,J)*UL(1,J)+SHP(1,J)*UL(2,J)
 13   ZETA=ZETA+SHP(1,J)*UL(3,J)+SHP(2,J)*UL(4,J)
      EPSV=EPS(1)+EPS(3)
C     P =PORE PRESSURE
      P=D(18)*(D(19)*EPSV+ZETA)
C     SIG(1) = STRESS IN  X-DIRECTION     ,  SIGMA-XX
C     SIG(2) = STRESS IN XY-DIRECTION     ,    TAU-XY
C     SIG(3) = STRESS IN  Y-DIRECTION     ,  SIGMA-YY
C
C     ....CALCULATE EFFECTIVE STRESSES
      SIG(1)=D(1)*EPS(1)+D(2)*EPS(3)+(D(19)-1.D0)*P
      SIG(2)=D(4)*EPS(2)
      SIG(3)=D(2)*EPS(1)+D(3)*EPS(3)+(D(19)-1.D0)*P
      CALL PVALUE (SIG,1,2,X,Y,Z,LN,IL,P,D)
C
C     ....CALCULATE TOTAL STRESSES
```

```
C       SIG(1)=SIG(1)+P
C       SIG(3)=SIG(3)+P
C       CALL PVALUE(SIG,1,2,X,Y,Z,LN,IL,P,D)
  14    CONTINUE
        RETURN
        END
```

END

FILMED

5-85

DTIC