Finite Element Program EPITOK—II

Tetsuhiko MIYOSHI*, Ken KAMINISHI** and Shun-ichi KAWANO**

(Received July 15, 1986)

Abstract

A finite element program for elastic—plastic stress analysis based on a new approximation of the constitutive equations is presented. A brief introduction to the theory of this new procedure and a detailed explanation of the subroutines that are essential for this method are also described.

Introduction

The purpose of this paper is to explain the finite element program EPITOK—II developed at Yamaguchi University for elastic—plastic stress analysis. This program is based on a new method of approximation for plasticity problem and characterized by the following.

(1) A new nonlinear scheme is used as the discrete constitutive equations instead of the conventional finite difference approximation. This scheme satisfies a consistency relation which does not hold in usual one step finite difference approximation of the constitutive equations.
(2) An iterative method is applied at each loading step using LU decomposition of the total stiffness matrix.
(3) An effective acceleration technique is employed.

Our main intention in publishing this program is to explain the essential features of the new procedure. Therefore the program presented in this paper can apply only to plane stress problems with isotropic hardening rule. However, the basic idea of the procedure is simple and general. Therefore it is applicable to all problems which can be treated by the usual iterative methods.

In §1 we summarize the theory of the method. §2 and §3 are devoted to the explanation of the main program and subroutines. In §4 the program itself is presented and a simple example of input data is shown for practical use of the program.

Notation.

u_i : displacements. We assume that the displacements of a certain portion of the boundary $\partial\Omega$ are prescribed so that a unique solution of the problem exists.

$\sigma = (\sigma_{11}, \sigma_{22}, \sigma_{12})$: stresses $(\sigma_{21} = \sigma_{12})$
$\varepsilon = (\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{12})$: strains

*Department of Applied Science
**Department of Mechanical Engineering

$$D = \frac{E}{1-\nu^2}\begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{pmatrix}$$

ν : Poisson's ratio
E : Young's modulus

$f^2(\sigma) = \sigma_{11}^2 + \sigma_{22}^2 - \sigma_{11}\sigma_{22} + 3\sigma_{12}^2$: yield function. The initial yielding is given by $f(\sigma) = z_0$.

$u_{,i} = \partial u / \partial x_i$

$\partial f = \{\partial f / \partial \sigma\}$

∂f^* : transposed ∂f

$D' = D\partial f \partial f^* D/(H' + \partial f^* D \partial f)$

$\dot{\sigma}$: time derivative of σ

$\bar{\varepsilon}^p$: effective plastic strain

$H(\bar{\varepsilon}^p)$: function to determine the yielding

$$S = \begin{pmatrix} 1 & -0.5 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

: this matrix is used to express ∂f as $\partial f = S\sigma / f$.

B : The so called "B" matrix defined as follows. Let N be the piecewise linear function taking value 1 at node n and value 0 at other nodes. Let i, j, k be the nodes of an element. Then B is a 3×6 matrix of the from

$$B^{(n)} = \begin{pmatrix} N_{,1}^{(n)} & 0 \\ 0 & N_{,2}^{(n)} \\ N_{,2}^{(n)} & N_{,1}^{(n)} \end{pmatrix} \qquad n = i,\ j,\ k$$

1. The theory of EPITOK—II

Let us consider a plane stress problem formulated as follows :

(1.1) $\sum_j \sigma_{ij,j} + F_i = 0 \qquad \text{in } \Omega \ (i = 1, 2)$

(1.2) $\begin{cases} \dot{\sigma} = D\dot{\varepsilon} & \text{if } f(\sigma) < H(\bar{\varepsilon}^p) \\ \dot{\sigma} = (D - D')\dot{\varepsilon} & \text{if } f(\sigma) = H(\bar{\varepsilon}^p) \text{ and } \partial f^* \dot{\sigma} > 0 \end{cases}$

(1.3) $\varepsilon_{11} = u_{1,1} \qquad \varepsilon_{22} = u_{2,2} \qquad \varepsilon_{12} = u_{1,2} + u_{2,1}.$

In EPITOK, unlike the conventional methods, the constitutive equations (1.2) are not approximated by the finite difference to increase the "accuracy" of the approximate solution. We introduce a new hardening parameter $\hat{\varepsilon}^p$ and a new function $G(\hat{\varepsilon}^p)$ by

(1.4) $G(\hat{\varepsilon}^p) = H(\bar{\varepsilon}^p) \qquad \hat{\varepsilon}^p = \int_0^{\bar{\varepsilon}_p} \sqrt{H'(s)}\, ds$

Then (1.2) can be written equivalently as

(1.5) $$\begin{cases} \dot{\varepsilon} - C\dot{\sigma} = 0, \ \dot{\hat{\varepsilon}}^p = 0 & \text{if } f(\sigma) < G(\hat{\varepsilon}^p) \\ \dot{\varepsilon} - C\dot{\sigma} = \dfrac{1}{\{G'(\hat{\varepsilon}^p)\}^2} \partial f \partial f^* \dot{\sigma}, \ \dot{\hat{\varepsilon}}^p = \dfrac{\partial f^* \dot{\sigma}}{G'(\hat{\varepsilon}^p)} & \text{if } f(\sigma) = G(\hat{\varepsilon}^p) \text{ and } \partial f^* \dot{\sigma} > 0. \end{cases}$$

The plastic strain increments $\dot{\varepsilon} - C\dot{\sigma}$ and the hardening parameter $\hat{\varepsilon}^p$ then satisfies, respectively,

(1.6) $$\dot{\varepsilon} - C\dot{\sigma} = \frac{\partial f^* \dot{\sigma}}{(G')^2} \partial f, \ \dot{\hat{\varepsilon}}^p = \frac{\partial f^* \dot{\sigma}}{(G')^2} G',$$

which represent a normality condition in a wider sense (see [1] for details). Now we introduce a parameter λ defined by

$$\lambda = \frac{\partial f^* \dot{\sigma}}{(G')^2 f(\sigma)}$$

Then the equations in (1.6) are written as

(1.7) $$\dot{\varepsilon} - C\dot{\sigma} = \lambda S \sigma, \ \dot{\hat{\varepsilon}}^p = \lambda f(\sigma) G'(\hat{\varepsilon}^p).$$

Hence we approximate these equations by

(1.8) $$\Delta\varepsilon^n - C\Delta\sigma^n = \lambda^{n+1} S \sigma^{n+1}, \ \Delta\hat{\varepsilon}^{p,n} = \lambda^{n+1} f(\sigma^{n+1}) G'(\hat{\varepsilon}^{p,n+1})$$

at n–th step of the integration, where $\Delta\varepsilon^n = \varepsilon^{n+1} - \varepsilon^n$. The parameter λ^{n+1} is determined by the condition $f(\sigma^{n+1}) = G(\hat{\varepsilon}^{p,n+1})$.

In EPITOK the increments of displacements, strains, stresses and $\Delta\hat{\varepsilon}^{p,n}$ at n–th step of integration are determined by the following system of equations.

(1.9) $$\Sigma B^* \Delta\sigma^n \cdot \text{area(element)} = \Delta F$$

(1.10) $$\Delta\varepsilon^n - C\Delta\sigma^n = 0, \ \Delta\hat{\varepsilon}^{p,n} = 0 \qquad \text{if } f(D\Delta\varepsilon^n + \sigma^n) < G(\hat{\varepsilon}^{p,n})$$

(1.11) $$\begin{cases} \Delta\varepsilon^n - C\Delta\sigma^n = \lambda^{n+1} S \sigma^{n+1} \\ \Delta\hat{\varepsilon}^{p,n} = \lambda^{n+1} f(\sigma^{n+1}) G'(\hat{\varepsilon}^{p,n+1}) \\ f(\sigma^{n+1}) = G(\hat{\varepsilon}^{p,n+1}), \end{cases} \qquad \text{if } f(D\Delta\varepsilon^n + \sigma^n) \geq G(\hat{\varepsilon}^{p,n}),$$

where ΔF is the increments of the load at n–th step.

Since the above system is nonlinear we need some iterative method to solve it. There are several methods which are useful to this end. In EPITOK the following method is employed. Let $(u^n, \varepsilon^n, \sigma^n, \hat{\varepsilon}^{p,n}, \lambda^n)$ be the values at n–th step. Give a starting value $\Delta u^n_{(0)}$. Iterate (A)

$\sim$ (B) for $k = 0, 1, 2, \ldots$.

(A) Calculate $(\Delta\sigma^n_{(k)}, \Delta\hat{\varepsilon}^{p,n}_{(k)}, \lambda^{n+1}_{(k)})$ by

$$(1.12) \quad \begin{cases} \Delta\varepsilon^n_{(k)} - C\sigma^n_{(k)} = 0 \\ \Delta\hat{\varepsilon}^{p,n}_{(k)} = 0 \qquad \text{if } f(D\Delta\varepsilon^n_{(k)} + \sigma^n) < G(\hat{\varepsilon}^{p,n}_{(k)}) \end{cases}$$

$$(1.13) \quad \begin{cases} \Delta\varepsilon^n_{(k)} - C\Delta\sigma^n_{(k)} = \lambda^{n+1}_{(k)} S\sigma^{n+1}_{(k)} \\ \Delta\hat{\varepsilon}^{p,n}_{(k)} = \lambda^{n+1}_{(k)} f(\sigma^{n+1}_{(k)}) G'(\hat{\varepsilon}^{p,n+1}_{(k)}) \\ f(\sigma^{n+1}_{(k)}) = G(\hat{\varepsilon}^{p,n+1}_{(k)}) \qquad \text{if } f(D\Delta\varepsilon^n_{(k)} + \sigma^n) \geq G(\hat{\varepsilon}^{p,n}_{(k)}) \end{cases}$$

where $\Delta\varepsilon^n_{(k)} = \varepsilon^{n+1}_{(k)} - \varepsilon^n$ etc., and $\varepsilon^{n+1}_{(k)} = \varepsilon(u^{n+1}_{(k)})$.

(B) Calculate $\Delta u^n_{(k+1)}$ by

$$(1.14) \qquad \tilde{K}\Delta u^n_{(k+1)} = \tilde{K}\Delta u^n_{(k)} - (\Sigma B^* \Delta\sigma^n \cdot \text{area(element)} - \Delta F),$$

where $\tilde{K}$ is the total elastic stiffness matrix with Poisson's ratio $\tilde{\nu}$ and Young's modulus $\tilde{E}$ modified by the formulas

$$\tilde{\nu} = \nu, \tilde{E} = E \qquad \text{for elastic elements}$$

$$\tilde{\nu} = \nu\frac{1+\lambda^n E/2\nu}{1+\lambda^n E}, \quad \tilde{E} = \frac{E}{1+\lambda^n E} \qquad \text{for plastic elements,}$$

The above replacements of the elastic constants are effective for accelerating the convergence. It was motivated by the observation that the matrix

$$(I + \lambda^{n+1}_{(k)} DS)^{-1} D$$

is the only variable matrix to determine $\Delta\sigma^n_{(k)}$, and it will not be so different from the matrix

$$(I + \lambda^n DS)^{-1} D = \frac{\tilde{E}}{1-\tilde{\nu}^2} \begin{pmatrix} 1 & \tilde{\nu} & 0 \\ \tilde{\nu} & 1 & 0 \\ 0 & 0 & (1-\tilde{\nu})/2 \end{pmatrix}$$

We summarize the theoretical results on this type of approximation obtained up to the present.

(1) The accuracy of the scheme (1.9) $\sim$ (1.11) is better than those obtained by the finite difference approximation to (1.1) $\sim$ (1.3).

(2) $\lambda^{n+1}_{(k)}$, $\Delta\sigma^n_{(k)}$ and $\Delta\hat{\varepsilon}^{p,n}_{(k)}$ in (1.12) $\sim$ (1.13) are uniquely determined.

(3) If $\tilde{\nu} = \nu$ and $\tilde{E} = E$ for all elements, and the hardening is kinematic, then the iteration (1.14) converges for any load increments ΔF. For the proof, see [2]. Some numerical results obtained by using this program are reported in [3].

References

1) T. Miyoshi, Foundations of the numerical analysis of plasticity, North−Holland Mathematics Studies 107, Kinokuniya/North−Holland, (1985).
2) T. Miyoshi, A new iterative method for solving quasi−static problem of plasticity (to appear in Japan Journal of Applied Mathematics).
3) T. Miyoshi, K. Kaminishi, S. Kawano and S. Shimizu, an iterative method for elastic−plastic stress analysis, Computational Mechanics '86, Theory and Applications, Splinger, IV−89 ~ IV−94 (1986).

2. The main program of EPITOK−II

The main program consists of four main blocks : Input and output, initiation, determination of yielding or unloading and inner iteration. Fig. 1 is the flow chart of the

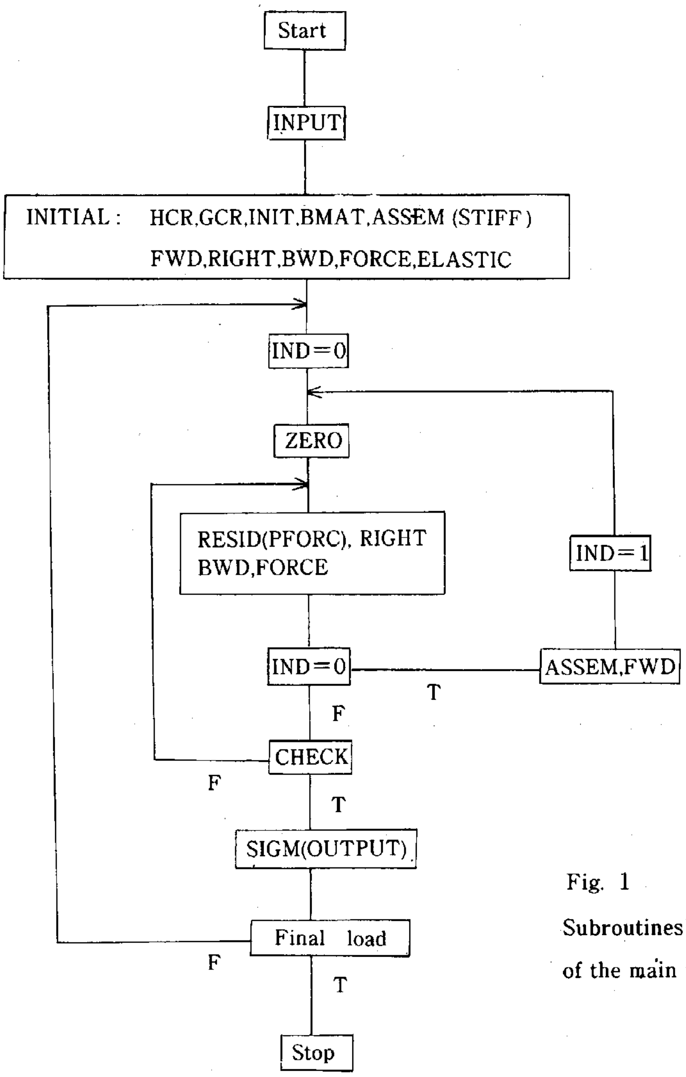

Fig. 1

Subroutines in the flow chart of the main program main program in which the disposition of the subroutines is shown.

00290-00380 Initiation and elastic solution.

00400-00880 Computation of the increments of displacements, strains, stresses and plasticity parameters by the iteration

$$\tilde{K}\Delta u^n_{(k+1)} = \tilde{K}\Delta u^n_{(k)} - (\Sigma B^* \Delta\sigma^n_{(k)} \cdot \text{area(element)} - \Delta F)$$

00450 Compute $G^2(\hat{\varepsilon}^{p,n})$, and clear the arrays of $D\Delta\varepsilon^n$, $\Delta\varepsilon^n$ and Δu^n.

00470 QP : Parameter to check the convergence of the iteration 00490–00830. This check is done every ten iterations.

00490-00580 Check if unloading occurs. This check is done by using old TK and new load increment during five extra iterations.

00590-00630 Assemble the new iteration matrix TK associated with $(I+\lambda^{n+1}DS)^{-1}D$ and carry a new LU decomposition to start the next step.

00650 If NUE is not a multiple of ten, the next convergence check is skipped.

00670-00680 Convergence check by the criterion

$$\| \Delta u^n_{(k+1)} - \Delta u^n_{(k)} \| / \| \Delta u^n_{(k)} \| \leq \text{DELTA}.$$

00720-00750 Check if NUE is a multiple of ten.

00770-00810 Store $u^{n+1}_{(k+1)}$ in AL1 and continue the iteration.

00850 If the convergence criterion is satisfied, stop the iteration and add the increments Δu^n etc. to u^n etc. to get u^{n+1} etc..

00900-00910 If NUE becomes greater than 300, the iteration is regarded as divergent.

3. The subroutines of EPITOK—II

In this section we explain the details of the subroutines used in EPITOK—II. For simplicity the explantion of the well known routines in the finite element technique is omitted.

SUBROUTINE INPUT

01080 Read NN, NE, NLN, NBN, E, ANU

01190 Read ZO, DF, DELTA, ETA, NEQ, IFINAL

01290 Read EQPSN, H

01420 Read X, Y

01530 Read ICON

01650 Read IB, DISP

01860 Read IZAL, ZAL, AL

01990-02000 Read IS1, IS2, IS3, IS4.

Read UN for nodes IS1 ~ IS2, STRESS etc. for elements IS3 ~ IS4 are printed.

SUBROUTINE INITIAL

Initiation to begin plastic computation

02270 Compute and store the vector HC (= H').

02280 Compute and store the vector GOC (= G).

02290 Initialise the variables of plastic parameters and compute the half bandwidth.

02300 Compute and store "B" matrix.

02320-02330 Assemble the total stiffness matrix associated with

$$(I+\lambda DS)^{-1}D \qquad (\lambda = 0 \text{ at this stage}).$$

and introduce the boundary conditions.

02340 Forward elimination of the symmetric, banded matrix TK.

02350 Modify the right side of the equations corresponding to the forward elimination.

02360 Backward substitution.

02370 Compute the stress $D\varepsilon^n$.

02380 Compute the minimun load and corresponding displacements,strains and stresses to cause the first yielding.

SUBROUTINE HCR

Computation and store of $H'(\bar{\varepsilon}^p(i))$ under the assumption that $G'(\hat{\varepsilon}^p)$ is piecewise linear. $H'(0)$ is determined by the derivative of the interpolating function of H(1), H(2) and H(3) by a quadratic polynomial. $H'(i+1)$ is calculated by the formula

$$H'(i+1) = H'(i)(1- ax)$$

where

$$a = [H(i+1)-H(i)]/[H'(i)(\bar{\varepsilon}^p(i+1)- \bar{\varepsilon}^p(i)]$$

and x is the solution of the equation

$$ax = 1-\exp(-x).$$

To get the above formula, solve the following equation on H' assuming that $G'(\hat{\varepsilon}^p)$ is linear in the same interval:

$$\sqrt{H'(\bar{\varepsilon}^p)} = G'(\hat{\varepsilon}^p)$$

$$\hat{\varepsilon}^p = \int_0^{\bar{\varepsilon}_p} \sqrt{H'(s)}\, ds \qquad (\bar{\varepsilon}(i) \leq \bar{\varepsilon}^p \leq \bar{\varepsilon}^p(i+1))$$

02660-02690 Newton's method applied to the above equation.

SUBROUTINE GCR

Compute and store $G'(\hat{\varepsilon}^p(i))$ and $G(\hat{\varepsilon}^p(i))$ by the formulas

$$\hat{\varepsilon}^p = \int_0^{\bar{\varepsilon}_p} \sqrt{H'(s)}\, ds, \qquad G'(\hat{\varepsilon}^p) = \sqrt{H'(\bar{\varepsilon}^p)}$$

and

$$G(\hat{\varepsilon}^p) = Z_0 + \int_0^{\bar{\varepsilon}^p} G'(S)\, ds.$$

$G'(\hat{\varepsilon}^p)$ is assumed to be linear in $[\hat{\varepsilon}^p(i), \hat{\varepsilon}^p(i+1)]$.

FUNCTION GOCR(X)

Function subroutine to define

$$G(\hat{\varepsilon}^p) = Z_0 + \int_0^{\bar{\varepsilon}_p} G'(s) ds.$$

FUNCTION G1CR(X)

Function subroutine to define

$$G'(\hat{\varepsilon}^p) = dG(\hat{\varepsilon}^p)/d\,\hat{\varepsilon}^p.$$

SUBROUTINE INIT

Initialisation of the arrays for parameters and calculation of common variables. Half bandwidth MS is calculated in this subroutine.

SUBROUTINE BMAT

The so called "B" matrix is calculated and stored in arrays B and D. The area of each element is also calculated and stored.

SUBROUTIME ASSEM

Assembling of the total stiffness matrix associated with $(I + \lambda DS)^{-1}D$, and introduction of the fixed boundary condition.

SUBROUTINE STIFF

Computation of the element stiffness matrix associated with $(I + \lambda DS)^{-1}D$.

SUBROUTINE FWD

Forward elimination of the symmetric, banded matrix A.

SUBROUTINE RIGHT

Modification of the right side B of the equation $AU = B$ corresponding to the forward elimination.

SUBROUTINE BWD

SUBROUTINE BWD

Backward substitution for the equation AU = B.

SUBROUTINE FORCE

Computation of the vectors $D\Delta\varepsilon^n_{(k)}$ and $\Delta\varepsilon^n_{(k)}$.

SUBROUTINE ELASTIC

The minimun load to cause the first yielding is calculated with the corresponding displacements, strains and stresses.

SUBROUTINE ZERO

Computation of $G^2(\hat{\varepsilon}^{p,n})$ and initiation of $D\Delta\varepsilon^n$, $\Delta\varepsilon^n$ and u^{n+1}.

SUBROUTINE RESID

Right side of the equation to determine $\Delta u^n_{(k+1)}$ is calculated and superposed in the array AL.

07120-07240 if $f(D\Delta\varepsilon^n + \sigma^n) < G(\hat{\varepsilon}^{p,n}) - 0.001$, then IEP = 0
if $f(D\Delta\varepsilon^n + \sigma^n) \geq G(\hat{\varepsilon}^{p,n}) - 0.001$, then IEP = 1.

07320-07790 if IEP = 0, then skip this loop
if IEP = 1, then calculate

ADF $(=(I+\lambda^n DS)^{-1} D\Delta\varepsilon^n_{(k)}) - DDF(=\Delta\sigma^n_{(k)})$.
DDF, $\lambda^{n+1}_{(k)}$ and EHNP1 are stored in arrays DSIGM, ARAMDA and AEH.

07810-07870 Introduce the boundary conditions.

07890-07970 Add the next load increments.

SUBROUTINE PFORC

The vector DDF = $\Delta\sigma^n_{(k)} = (I+\lambda^{n+1}_{(k)} DS)^{-1}(D\Delta\varepsilon^n_{(k)} + \sigma^n) - \sigma^n$ is calculated in this subroutine.

08210-08710 Newton's method applied to the equation

$$\begin{aligned} \text{FRAM}(\lambda) &= \text{FSIGM}(\lambda) - \text{GEHP}(\lambda) \\ &(= [\, f(\sigma) - G(\hat{\varepsilon}^p)\,](\lambda)) \\ &= 0 \end{aligned}$$

The iteration ends if the following is satisfied.

$$\text{ABS}(\, f(\sigma) - G(\hat{\varepsilon}^p))/f(\sigma) \leq \text{ETA}.$$

08210-08260 Compute

$$(I+\lambda DS)^{-1} = \begin{pmatrix} AA & BB & 0 \\ BB & AA & 0 \\ 0 & 0 & CC \end{pmatrix}$$

08270-08290	Compute $(I+\lambda DS)^{-1}(D\Delta\varepsilon^{n}+\sigma^{n}) = \sigma^{n+1}$.
08300-08310	FSIGM $= f(\sigma^{n+1})$.
08320	$A = \lambda f(\sigma^{n+1})$
08340-08370	Find such i that $\hat{\varepsilon}^{p}(i-1) \leq \hat{\varepsilon}^{p,n} < \hat{\varepsilon}^{p}(i)$.
08380-08450	Find such j that $\hat{\varepsilon}^{p}(j) - \hat{\varepsilon}^{p,n} > \lambda f(\sigma^{n+1})G'(\hat{\varepsilon}^{p}(j))$.
08460	GK1 $= G'(j-1)$
08460-08500	Compute such $x = \hat{\varepsilon}^{p}$ that $x - \hat{\varepsilon}^{p,n} = \lambda f(\sigma^{n+1})G'(x)$.
08520	G1CRM $= G'(\hat{\varepsilon}^{p})$
08530	G2CRM $= G''(\hat{\varepsilon}^{p})$
08550	FRAM $= f(\sigma^{n+1}) - G(\hat{\varepsilon}^{p})$
08560-08580	Compute $S\sigma^{n+1}$
08590-08610	Compute $(I+\lambda DS)^{-1}S$
08620-08640	Compute $SD(I+\lambda DS)^{-1}S\sigma^{n+1}$
08650	$df/d\lambda = -\langle SD(I+\lambda DS)^{-1}S\sigma^{n+1},\ \sigma^{n+1}\rangle / f(\sigma^{n+1})$
08660-08670	$d\hat{\varepsilon}^{p}/d\lambda = \langle (I+\lambda DS)^{-1}S\sigma^{n+1},\ \sigma^{n+1}\rangle G'(\hat{\varepsilon}^{p}) / [f(\sigma^{n+1})(1-\lambda f(\sigma^{n+1})G''(\hat{\varepsilon}^{p})]$
08680	FCR $= F'(\lambda) = df/d\lambda - G'(\hat{\varepsilon}^{p})d\hat{\varepsilon}^{p}/d\lambda$
08690	$\lambda_{(k+1)} = \lambda_{(k)} - F(\lambda_{(k)})/F'(\lambda_{(k)})$
08710	Iterate until the following is satisfied. $\lvert f(\sigma^{n+1}) - G(\hat{\varepsilon}^{p})\rvert / f(\sigma^{n+1}) \leq$ ETA.
08730	EHNP1 $= \hat{\varepsilon}^{p,n+1} = \hat{\varepsilon}^{p,n} + \lambda f(\sigma^{n+1})G'(\hat{\varepsilon}^{p,n+1})$
08740-08760	DDF $= \sigma^{n+1} - \sigma^{n}$.

SUBROUTINE CHECK

Convergence check by a relative error of displacement increments.

SUBROUTINE SIGM

Preparation to proceed to the next step of integration.

09230-09270 $u^{n+1} = u^{n} + \Delta u^{n}$.

09350-09640 if IEP = 0

$$\sigma^{n+1} = \sigma^{n} + D\Delta\varepsilon^{n}$$
$$\varepsilon^{n+1} = \varepsilon^{n} + \Delta\varepsilon^{n}$$
$$\lambda = 0$$

and calculate $f(\sigma^{n+1})$,

if IEP = 1

$$\sigma^{n+1} = \sigma^{n} + \Delta\sigma^{n}$$
$$\varepsilon^{n+1} = \varepsilon^{n} + \Delta\varepsilon^{n}$$
$$\lambda^{n+1} = \text{ARAMADA(NEL)}$$
$$\hat{\varepsilon}^{p,n+1} = \text{AEH(NEL)}$$
$$\bar{\varepsilon}^{p,n+1} = \bar{\varepsilon}^{p,n} + 0.5(1/G'(\hat{\varepsilon}^{p,n+1}) + 1/G'(\hat{\varepsilon}^{p,n}))\Delta\hat{\varepsilon}^{p,n}$$

(approximate value of

$$\bar{\varepsilon}^{p,n+1} = \int_0^{\hat{\varepsilon}^{p,n+1}} \frac{1}{G'(s)}\,ds$$

by the trapezoidal rule),

and calculate $f(\sigma^{n+1})$.

09660-09700 Compute the total load.

SUBROUTINE OUTPUT

Print: ISTAT, FX, FY, NUE
IEP(I) (I = 1, NE)
UN(2(IS1−1)+1) ~ UN(2(IS2−1)+2
FSO(IS3) ~ FSO(IS4)
EEQ(IS3) ~ EEQ(IS4)
ZRAMDA(IS3) ~ ZRAMDA(IS4)
STRESS(k), STRAIN(k)
(k = 3(IS3−1)+1, 3(IS4−1)+3)

4. The program of EPITOK−Ⅱ

1. Comments

C1. Problem .... incremental, quasi−static, plane stress problem
C2. Hardening .... isotropic hardening
C3. Finite element .... linear on triangular element
C4. Numerical method .... accelerated iterative method
C5. Maximum number of elements 300
C6. Maximum number of loaded nodes 20
C7. Maximum number of fixed nodes 20
C8. Maximum half bandwidth 150
C9. Maximun number of values of $H(\bar{\varepsilon}^p)$ as input data 20
C10. The change of the thickness is not considered.

2. Data, variables and arrays

NN : number of nodes
NE : number of elements
NLN : number of loaded nodes
NBN : number of boundary nodes
N : 2×NN (number of unknowns)
MS : half bandwidth of total stiffness matrix
E : Young's modulus
ANU : Poisson's ratio
ZO : yield stress
ETA : relative error to stop Newton's iteration in PFORC
NUE : number of iteration in the main program
DELTA : relative error to stop iteration on NUE

DF	: ratio to determine the load increment
IFINAL	: maximum number of integration step
NEQ	: number of $\bar{\varepsilon}^p(i)$ as imput data (≤ 20)
EQPSN(i)	: input data $\bar{\varepsilon}^p(i)$
H(i)	: input data $H(\bar{\varepsilon}^p(i))$
HC(i)	: $H'(i)$ calculated from H(i)
EQPSNH(i)	: calculated $\hat{\varepsilon}^p(i)$ from input data $\bar{\varepsilon}^p(i)$
GOC(i)	: calculated $G(\hat{\varepsilon}^p(i)$ from $G'(\hat{\varepsilon}^p(i))$
G1C(i)	: calculated $G'(\hat{\varepsilon}^p(i))$ from HC(i)
X(i)	: x–coordinate of i–th node
Y(i)	: y–coordinate of i–th node
ICON(i)	: one dimensional array of node. ICON(3(k−1)+1), ICON(3(k−1)+2) and ICON(3(k−1)+3) denote the first, second and third nodes of k–th elements placed counterclockwise.
IB(i)	: one dimensional array of boundary data. IB(3(k−1)+1) denotes the node number of k–th fixed node, and IB(3(k−1)+2) and IB(3(k−1)+3) denote the status of the displacements in x and y directions. 1 denotes free and 0 denotes prescribed.
(BA, DA)	: "B" matrix of an element
IZAL(i)	: node number of i–th loaded node
ZAL(i)	: one dimensional array of the load vector given initially. ZAL(2(k−1)+1) and ZAL(2(k−1)+2) include x and y components of the load at k–th node. DF*ZAL is the load increment at one step.
AL(i)	: one dimensional array of load or increments of displacements. AL(2(k−1)+1) and AL(2(k−1)+2) denote x and y components of the load. After solving the equilibrium equation this array is used to store the increments of displacements.
TK(i, j)	: total stiffness matrix
FORC(i)	: one dimensional array of $D\Delta\varepsilon^n$
DEPS(i)	: one dimensional array of strain increments $\Delta\varepsilon^n$ at each step.
ELST(i, j)	: element stiffness matrix
EH(i)	: $\hat{\varepsilon}^{p,n}$ of element i
AEH(i)	: $\hat{\varepsilon}^{p,n+1}$ of element i
STRAIN(i)	: one dimensional array of strains
STRESS(i)	: one dimensional array of stresses
DSIGM(i)	: one dimensional array of stress increments
UN(i)	: one dimensional array of displacements
FSO(i)	: $f(\sigma)$ of element i
ZN2(i)	: $f^2(\sigma)-0.001$. $ZN2(i)\geq G(\hat{\varepsilon}^p)$ is used to determine yielding.
EEQ(i)	: $\bar{\varepsilon}^{p,n}$ of element i
ZRAMDA(i)	: λ^n of element i
ARAMDA(i)	: λ^{n+1} of element i
IPE(i)	: parameter to distinct elastic and plastic. 1 denotes plastic and 0 elastic.
ADF(i)	: the vector $(I+\lambda^n DS)^{-1}\, D\Delta\varepsilon^n$ of an element
DDF(i)	: the vector $\Delta\sigma^n$ of an element

FX : x—component of total load
FY : y—component of total load

Finite Element Program EPITOK—II

```
C-----------------------------------------------------------------------
C
C
C           ***********      EPITOK-II      ***********
C
C
C                         ELASTIC-PLASTIC ITERATIVE
C                           METHOD TOKIWA-II
C
C
C-----------------------------------------------------------------------
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /RR/  RA,RB,RC
      COMMON /EQ/  NEQ,EQPSN(20),H(20)
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
      COMMON /OUT/ IN,IO,IS1,IS2,IS3,IS4,ISTAT
      COMMON /ELT/ NUE,FX,FY
C
      DIMENSION X(300),Y(300),ICON(900),IB(60),ZRAMDA(300),
     &          TK(600,150),AL(600),FORC(900),ELST(20,20),
     &          IEP(300),AL1(600),ARAMDA(300),DSIGM(900),
     &          EH(300),DEPS(900),STRAIN(900),ZAL(40),AEH(300),
     &          STRESS(900),UN(600),FSO(300),EEQ(300),ZN2(300),
     &          B(300,3),D(300,3),BA(300,3),DA(300,3),IZAL(20)
C
C
      NRMX=600
      NCMX=150
      IN=5
      IO=6
C
C
      CALL INPUT(X,Y,ICON,AL,IB,ZAL,IZAL)
      CALL INITIAL(X,Y,ICON,AL,IB,ZAL,ZRAMDA,EH,ISTAT,
     &             B,D,BA,DA,TK,ELST,IEP,NRMX,NCMX,FORC,
     &             DEPS,STRESS,FSO,STRAIN,UN)
C
   10 NUE=0
      IND=0
      KK=0
      ISTAT=ISTAT+1
C
   20 CALL ZERO(EH,ZN2,FORC,DEPS,AL1)
C
      QP=1.0
C
   30 CALL RESID(FORC,STRESS,ZN2,IEP,AL,ZRAMDA,EH,RAMDA,EHNP1,
     &           ICON,IB,ZAL,B,D,DSIGM,ARAMDA,AEH,IZAL)
C
      CALL RIGHT(TK,AL,N,MS,NRMX,NCMX)
      CALL BWD(TK,AL,N,MS,NRMX,NCMX)
      CALL FORCE(ICON,FORC,AL,DEPS,BA,DA)
C
      IF(IND.NE.0) GOTO  50
      KK=KK+1
```

```
      IF(KK-5) 30,30,40
   40 CALL ASSEM(ICON,TK,ELST,ZRAMDA,B,D,BA,DA,IEP,
     &             AL,IB)
      CALL FWD(TK,N,MS,NRMX,NCMX)
      IND=1
      GOTO 20
C
   50 IF(QP.GT.0.01) GOTO 60
C
      CALL CHECK(AL,AL1,RSD,RDEL)
      IF(RSD-RDEL) 90,90,60
C
   60 IF(NUE-300) 70,100,100
C
   70 NUE=NUE+1
      RN=FLOAT(NUE)
      QP=RN*0.1-AINT(RN*0.1)
      IF(QP.GT.0.01) GOTO 30
C
      DO 80 I=1,NN
      L=2*(I-1)+1
      AL1(L)=AL(L)
      AL1(L+1)=AL(L+1)
   80 CONTINUE
C
      GO TO 30
C
   90 CALL SIGM(UN,AL,IEP,FORC,STRESS,EH,ZRAMDA,DEPS,STRAIN,
     &            EEQ,ZAL,DSIGM,ARAMDA,AEH)
C
      IF(ISTAT-IFINAL) 10,120,120
C
  100 WRITE(IO,110)
  110 FORMAT(5X,"DIVERGENT")
C
  120 STOP
      END
C
C
      SUBROUTINE INPUT(X,Y,ICON,AL,IB,ZAL,IZAL)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /OUT/ IN,IO,IS1,IS2,IS3,IS4,ISTAT
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /EQ/  NEQ,EQPSN(20),H(20)
C
      DIMENSION X(300),Y(300),ICON(900),AL(600),IB(60),
     &            W(3),IC(3),ZAL(40),IZAL(20)
C
C
      READ(IN,10) NN,NE,NLN,NBN,E,ANU
      WRITE(IO,20) NN,NE,NLN,NBN,E,ANU
   10 FORMAT(4I5,2F10.0)
   20 FORMAT(///57X," FUNDAMENTAL DATA  "
     &          //15X,"     NUMBER OF NODES       =",I5,
     &            30X,"    NUMBER OF ELEMENTS     =",I5
     &          //15X," NUMBER OF LOADED NODES  =",I5,
     &            30X,"NUMBER OF BOUNDARY NODES =",I5
     &          //15X,"     YOUNG'S MODULUS        = ",1PE12.6,
     &            22X,"     POISSON'S RATIO        = ",1PE12.6)
```

```
C
      READ(IN,30) ZO,DF,DELTA,ETA,NEQ,IFINAL
   30 FORMAT(4E10.3,2I10)
      WRITE(IO,40) ZO,DF,DELTA,ETA,NEQ,IFINAL
   40 FORMAT(/17X,"   YIELD STRESS          =",1PE14.7,
     &          23X,"  INCREMENT OF LOAD     =",1PE14.7
     &        //17X,"  CONVERGENCE OF U      =",1PE14.7,
     &          23X,"CONVERGENCE OF RAMDA   =",1PE14.7
     &        //26X,"NEQ",11X,"=",I4,40X,"IFINAL",10X,"=",I4)
C
      DO 50 I=1,NEQ
      READ(IN,60) J,EQPSN(I),H(I)
   50 CONTINUE
   60 FORMAT(I5,2F10.0)
      WRITE(IO,70)
   70 FORMAT(//3(5X,"NO.",5X,"EQ.P.STRAIN",5X,"EQ.STRESS"))
      WRITE(IO,80) (I,EQPSN(I),H(I),I=1,NEQ)
   80 FORMAT(/3(I8,6X,F10.3,4X,F10.3))
C
      WRITE(IO,90)
   90 FORMAT(///48X,"  COORDINATES OF NODES "
     &        ///14X,"NODE",8X,"X",9X,"Y",17X,"NODE",8X,"X",
     &           9X,"Y",17X,"NODE",8X,"X",9X,"Y"//)
      DO 100 I=1,NN
      READ(IN,110) J,X(I),Y(I)
  100 CONTINUE
  110 FORMAT(I5,2F10.0)
      WRITE(IO,120)(I,X(I),Y(I),I=1,NN)
  120 FORMAT(8X,I10,2F10.2,10X,I10,2F10.2,10X,I10,2F10.2)
C
      WRITE(IO,130)
  130 FORMAT(///62X," ELEMENT CONNECTIONS "
     &     //8X,3(2X,"ELEMENT NO. ",5X,"I",5X,"J",5X,"K",8X)/)
      DO 140 I=1,NE
      L=3*(I-1)+1
      READ(IN,150) J,ICON(L),ICON(L+1),ICON(L+2)
  140 CONTINUE
      WRITE(IO,160) (J,ICON(3*J-2),ICON(3*J-1),ICON(3*J),J=1,NE)
  150 FORMAT(4I5)
  160 FORMAT(3(11X,I6,5X,3I6))
C
      WRITE(IO,170)
  170 FORMAT(///43X,"  BOUNDARY CONDITIONS ",
     &             "(FREE=1,FIXED=0)   "
     &        //10X,3("NODE",9X,"X",9X,"Y",16X)//)
C
      DO 180 I=1,NBN
      READ(IN,190) J,(IC(K),K=1,2)
      L1=3*(I-1)+1
      L2=2*(J-1)
      IB(L1)=J
      DO 180 K=1,2
      N1=L1+K
      N2=L2+K
      IB(N1)=IC(K)
  180 CONTINUE
  190 FORMAT(3I10)
      WRITE(IO,200) (IB(3*(K-1)+1),IB(3*(K-1)+2),
     &               IB(3*(K-1)+3),K=1,NBN)
  200 FORMAT(10X,I4,2I10,16X,I4,2I10,16X,I4,2I10)
```

```
C
      WRITE(IO,210)
  210 FORMAT(///50X," LOADED CONDITIONS  "
     &       //7X,3("NODE",9X,"FX",9X,"FY",16X)//)
      N=NN*2
      DO 220 I=1,N
  220 AL(I)=0.
      DO 240 I=1,NLN
      READ(IN,230) J, (W(K),K=1,2)
  230 FORMAT(I10,2F10.0)
      N1=2*(I-1)+1
      IZAL(I)=J
      ZAL(N1)=W(1)
      ZAL(N1+1)=W(2)
      DO 240 K=1,2
      L=2*(J-1)+K
  240 AL(L)=W(K)
      WRITE(IO,250) (IZAL(J),ZAL(2*(J-1)+1),
     &               ZAL(2*(J-1)+2),J=1,NLN)
  250 FORMAT(3(I11,5X,1PE10.4,1X,1PE10.4,5X))
C
      READ(IN,260) IS1,IS2
      READ(IN,260) IS3,IS4
  260 FORMAT(2I10)
C
      WRITE(IO,270)
  270 FORMAT(///5X,12("----------")//)
C
      RETURN
      END
C
C
      SUBROUTINE INITIAL(X,Y,ICON,AL,IB,ZAL,ZRAMDA,EH,ISTAT,
     &                   B,D,BA,DA,TK,ELST,IEP,NRMX,NCMX,
     &                   FORC,DEPS,STRESS,FSO,STRAIN,UN)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /RR/  RA,RB,RC
      COMMON /EQ/  NEQ,EQPSN(20),H(20)
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
      COMMON /ELT/ NUE,FX,FY
C
      DIMENSION X(300),Y(300),ICON(900),IB(60),ZRAMDA(300),
     &          TK(600,150),AL(600),FORC(900),ELST(20,20),
     &          EH(300),DEPS(900),STRAIN(900),ZAL(40),
     &          STRESS(900),UN(600),FSO(300),IEP(300),
     &          B(300,3),D(300,3),BA(300,3),DA(300,3)
C
      CALL HCR
      CALL GCR
      CALL INIT(ICON,ZRAMDA,EH,ISTAT)
      CALL BMAT(X,Y,ICON,B,D,BA,DA)
C
      CALL ASSEM(ICON,TK,ELST,ZRAMDA,B,D,BA,DA,IEP,AL,IB)
C
      CALL FWD(TK,N,MS,NRMX,NCMX)
      CALL RIGHT(TK,AL,N,MS,NRMX,NCMX)
      CALL BWD(TK,AL,N,MS,NRMX,NCMX)
      CALL FORCE(ICON,FORC,AL,DEPS,BA,DA)
```

```
      CALL ELASTIC(FORC,STRESS,FSO,DEPS,STRAIN,AL,UN,ZAL)
C
      RETURN
      END
C
C
      SUBROUTINE HCR
C
      COMMON /EQ/  NEQ,EQPSN(20),H(20)
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      A1=H(2)-H(1)
      A2=H(3)-H(1)
      EQ1=EQPSN(2)-EQPSN(1)
      EQ2=EQPSN(3)-EQPSN(1)
      EQ3=EQPSN(3)-EQPSN(2)
C
      HC(1)=(A1/EQ1)*(EQ2/EQ3)-(A2/EQ2)*(EQ1/EQ3)
C
      I=0
   10 I=I+1
      A1=H(I+1)-H(I)
      EQ1=EQPSN(I+1)-EQPSN(I)
      A=HC(I)*EQ1
      A=A1/A
      IF(A.GE.1.0-0.0001) GOTO 30
      RLOG=10.0
      DO 20 II=1,6
      REX=EXP(-RLOG)
      RLOG=RLOG-(A*RLOG-1.0+REX)/(A-REX)
   20 CONTINUE
      HC(I+1)=HC(I)*(1.0-A*RLOG)
      IF(I.LT.NEQ-1) GOTO 10
      GOTO 40
   30 HC(I+1)=HC(I)
      IF(I.LT.NEQ-1) GOTO 10
C
   40 RETURN
      END
C
C
      SUBROUTINE GCR
C
      COMMON /EQ/  NEQ,EQPSN(20),H(20)
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      EQPSNH(1)=0.0
      DO 20 I=2,NEQ
      Z=0.0
      IM1=I-1
      DO 10 K=1,IM1
      Z=Z+0.5*(SQRT(HC(K+1))+SQRT(HC(K)))*(EQPSN(K+1)-EQPSN(K))
   10 CONTINUE
      EQPSNH(I)=Z
   20 CONTINUE
      DO 30 I=1,NEQ
      G1C(I)=SQRT(HC(I))
   30 CONTINUE
      GOC(1)=ZO
```

```
      DO 50 I=2,NEQ
      Z=0.0
      IM1=I-1
      DO 40 K=1,IM1
      Z=Z+0.5*(G1C(K+1)+G1C(K))*(EQPSNH(K+1)-EQPSNH(K))
   40 CONTINUE
      GOC(I)=Z+ZO
   50 CONTINUE
C
      RETURN
      END
C
C
      FUNCTION GOCR(X)
C
      COMMON /EQ/ NEQ,EQPSN(20),H(20)
      COMMON /GC/ HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      E=EQPSNH(NEQ)
      IF(X.GE.E) GOTO 30
      K=1
   10 EH=EQPSNH(K+1)
      IF(X.LT.EH) GOTO 20
      K=K+1
      GOTO 10
   20 A=G1C(K)
      B=EQPSNH(K)
      C=X-B
      GOCR=GOC(K)+A*C
      GOCR=GOCR+0.5*(G1C(K+1)-A)*C*C/(EH-B)
      GOTO 40
   30 GOCR=GOC(NEQ)+G1C(NEQ)*(X-EQPSNH(NEQ))
C
   40 RETURN
      END
C
C
      FUNCTION G1CR(X)
C
      COMMON /EQ/ NEQ,EQPSN(20),H(20)
      COMMON /GC/ HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      E=EQPSNH(NEQ)
      IF(X.GE.E) GOTO 30
      K=1
   10 EH=EQPSNH(K+1)
      IF(X.LT.EH) GOTO 20
      K=K+1
      GOTO 10
   20 A=G1C(K)
      B=EQPSNH(K)
      G1CR=A+(G1C(K+1)-A)*(X-B)/(EH-B)
      GOTO 40
   30 G1CR=G1C(NEQ)
C
   40 RETURN
      END
C
C
      SUBROUTINE INIT(ICON,ZRAMDA,EH,ISTAT)
```

```
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /ELT/ NUE,FX,FY
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /RR/  RA,RB,RC
C
      DIMENSION ICON(900),ZRAMDA(300),EH(300)
C
      ISTAT=0
      F=0.0
      FX=0.0
      FY=0.0
      RA=E*(1.0-0.5*ANU)/(1.0-ANU*ANU)
      RB=E*(ANU-0.5)/(1.0-ANU*ANU)
      RC=E*1.5*(1.0-ANU)/(1.0-ANU*ANU)
C
      DO 10 NEL=1,NE
      ZRAMDA(NEL)=0.0
      EH(NEL)=0.0
   10 CONTINUE
C
      N1=2
      MS=0
C
      DO 30 I=1,NE
      L1=3*(I-1)
      DO 30 J=1,N1
      L2=L1+J
      J1=J+1
      DO 30 K=J1,3
      L3=L1+K
      L=IABS(ICON(L2)-ICON(L3))
      IF(MS-L)20,30,30
   20 MS=L
   30 CONTINUE
      MS=2*(MS+1)
C
      RETURN
      END
C
C
      SUBROUTINE BMAT(X,Y,ICON,B,D,BA,DA)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
C
      DIMENSION X(300),Y(300),ICON(900),B(300,3),D(300,3),
     &          BA(300,3),DA(300,3)
C
      DO 20 I=1,NE
      L=3*(I-1)+1
      N1=ICON(L)
      N2=ICON(L+1)
      N3=ICON(L+2)
C
      B(I,1)=Y(N2)-Y(N3)
      B(I,2)=Y(N3)-Y(N1)
      B(I,3)=Y(N1)-Y(N2)
      D(I,1)=X(N3)-X(N2)
      D(I,2)=X(N1)-X(N3)
      D(I,3)=X(N2)-X(N1)
```

```
      ARE=(B(I,1)*D(I,2)-B(I,2)*D(I,1))
      DO 10 J=1,3
      BA(I,J)=B(I,J)/ARE
      DA(I,J)=D(I,J)/ARE
   10 CONTINUE
   20 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE ASSEM(ICON,TK,ELST,ZRAMDA,B,D,BA,DA,IEP,AL,IB)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
C
      DIMENSION ICON(900),TK(600,150),ELST(6,6),BA(300,3),ND(3),
     &          ZRAMDA(300),B(300,3),D(300,3),DA(300,3),IEP(300),
     &          AL(600),IB(60)
C
      DO 10 I=1,N
      DO 10 J=1,MS
   10 TK(I,J)=0.
C
      DO 30 I=1,NE
      NEL=I
C
      CALL STIFF(NEL,ELST,ZRAMDA,B,D,BA,DA,IEP)
C
      N1=3*(NEL-1)
      ND(1)=ICON(N1+1)
      ND(2)=ICON(N1+2)
      ND(3)=ICON(N1+3)
C
      DO 20 J=1,6
      JJ=(J+1)/2
      JM=2*JJ-J
      K=2*ND(JJ)-JM
      DO 20 L=1,6
      LL=(L+1)/2
      LM=2*LL-L
      M=2*ND(LL)-LM
      IF(K.GT.M) GOTO 20
      MM=M-K+1
      IF(J.GT.L) ELST(J,L)=ELST(L,J)
      TK(K,MM)=TK(K,MM)+ELST(J,L)
   20 CONTINUE
   30 CONTINUE
C
      DO 90 L=1,NBN
      L1=3*(L-1)+1
      NO=IB(L1)
      K1=2*(NO-1)
      DO 90 I=1,2
      L2=L1+I
      IF(IB(L2))90,40,90
   40 KR=K1+I
      DO 80 J=2,MS
      KV=KR+J-1
      IF(N-KV)60,50,50
```

```
 50 TK(KR,J)=0.
 60 KV=KR-J+1
    IF(KV)80,80,70
 70 TK(KV,J)=0.
 80 CONTINUE
    TK(KR,1)=1.
    AL(KR)=0.
 90 CONTINUE
C
    RETURN
    END
C
C
    SUBROUTINE STIFF(NEL,ELST,ZRAMDA,B,D,BA,DA,IEP)
C
    COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
C
    DIMENSION ELST(6,6),BA(300,3),IEP(300),
   &          B(300,3),D(300,3),ZRAMDA(300),DA(300,3)
C
    DO 10 I=1,6
    DO 10 J=1,6
 10 ELST(I,J)=0.
C
    Q=ZRAMDA(NEL)
    Q=Q*E
    NA=IEP(NEL)
    IF(NA.EQ.0) Q=0.0
    ANUU=(ANU+Q/2.0)/(1.0+Q)
    ED=E/(1.0+Q)
    ANUP=(1.-ANUU)/2.
    C=0.5*ED/(1.0-ANUU*ANUU)
C
    DO 20 I=1,3
    I2=2*I
    DO 20 J=1,3
    J2=2*J
    ELST(I2-1,J2-1)=C*(B(NEL,I)*BA(NEL,J)+ANUP*D(NEL,I)*DA(NEL,J))
    ELST(I2-1,J2)=C*(B(NEL,I)*DA(NEL,J)*ANU+D(NEL,I)*BA(NEL,J)*ANUP)
    ELST(I2,J2-1)=C*(B(NEL,J)*DA(NEL,I)*ANU+D(NEL,J)*BA(NEL,I)*ANUP)
 20 ELST(I2,J2)=C*(B(NEL,I)*BA(NEL,J)*ANUP+D(NEL,I)*DA(NEL,J))
C
    RETURN
    END
C
C
    SUBROUTINE FWD(A,N,MS,NX,MX)
C
    DIMENSION A(NX,MX)
C
    N1=N-1
    NM2=N-MS+2
    DO 10 I=NM2,N
    NI2=N-I+2
    DO 10 J=NI2,MS
 10 A(I,J)=0.0
    DO 40 I=1,N1
    IL=MIN0(MS+I-1,N)
    I1=I+1
    DO 30 K=I1,IL
```

```
      DO 20 JD=1,MS
      KJDI=K+JD-I
      KI1=K-I+1
   20 A(K,JD)=A(K,JD)-A(I,KJDI)*A(I,KI1)/A(I,1)
   30 CONTINUE
   40 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE RIGHT(A,B,N,MS,NX,MX)
C
      DIMENSION A(NX,MX),B(NX)
C
      N1=N-1
      DO 20 I=1,N1
      IL=MIN(MS+I-1,N)
      I1=I+1
      DO 10 K=I1,IL
      KI1=K-I+1
   10 B(K)=B(K)-B(I)*A(I,KI1)/A(I,1)
   20 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE BWD(A,B,N,MS,NX,MX)
C
      DIMENSION A(NX,MX),B(NX)
C
      B(N)=B(N)/A(N,1)
      N1=N-1
      M1=MS-1
      DO 20 I=1,M1
      NI=N-I
      RX=0.0
      DO 10 J=1,I
      J1=J+1
      NIJ=N-I+J
   10 RX=RX+A(NI,J1)*B(NIJ)
   20 B(NI)=(B(NI)-RX)/A(NI,1)
      DO 40 I=MS,N1
      NI=N-I
      RX=0.0
      DO 30 J=1,M1
      J1=J+1
      NIJ=N-I+J
   30 RX=RX+A(NI,J1)*B(NIJ)
   40 B(NI)=(B(NI)-RX)/A(NI,1)
      RETURN
      END
C
C
      SUBROUTINE FORCE(ICON,FORC,AL,DEPS,BA,DA)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
C
      DIMENSION ICON(900),FORC(900),BA(300,3),DA(300,3),
     &          AL(600),DEPS(900)
```

```
C
      C=E/(1.-ANU*ANU)
C
      DO 10 I=1,NE
      L=3*(I-1)+1
      K1=2*(ICON(L)-1)
      K2=2*(ICON(L+1)-1)
      K3=2*(ICON(L+2)-1)
      L=3*(I-1)
      DEPS(L+1)=BA(I,1)*AL(K1+1)+BA(I,2)*AL(K2+1)+BA(I,3)*AL(K3+1)
      DEPS(L+2)=DA(I,1)*AL(K1+2)+DA(I,2)*AL(K2+2)+DA(I,3)*AL(K3+2)
      DEPS(L+3)=DA(I,1)*AL(K1+1)+BA(I,1)*AL(K1+2)+DA(I,2)*AL(K2+1)+
     &          BA(I,2)*AL(K2+2)+DA(I,3)*AL(K3+1)+BA(I,3)*AL(K3+2)
      FORC(L+1)=C*(DEPS(L+1)+ANU*DEPS(L+2))
      FORC(L+2)=C*(ANU*DEPS(L+1)+DEPS(L+2))
      FORC(L+3)=C*(1.-ANU)*0.5*DEPS(L+3)
   10 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE ELASTIC(FORC,STRESS,FSO,DEPS,STRAIN,AL,UN,ZAL)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /ELT/ NUE,FX,FY
      COMMON /OUT/ IN,IO,IS1,IS2,IS3,IS4,ISTAT
C
      DIMENSION  FORC(900),STRESS(900),FSO(300),DEPS(900),STRAIN(900),
     &           AL(600),UN(600),ZAL(60)
      DO 10 I=1,NE
      N1=3*(I-1)+1
      FS1=FORC(N1)*FORC(N1)+FORC(N1+1)*FORC(N1+1)
     &      -FORC(N1)*FORC(N1+1)+3.0*FORC(N1+2)*FORC(N1+2)
      IF(FS1.GT.F) F=FS1
   10 CONTINUE
C
      F=SQRT(F)
      RATIO=ZO/F
C
      DO 20 I=1,NE
      N1=3*(I-1)+1
      STRESS(N1)=RATIO*FORC(N1)
      STRESS(N1+1)=RATIO*FORC(N1+1)
      STRESS(N1+2)=RATIO*FORC(N1+2)
      FSO(I)=SQRT(STRESS(N1)*STRESS(N1)+STRESS(N1+1)*STRESS(N1+1)-
     &       STRESS(N1)*STRESS(N1+1)+3.0*(STRESS(N1+2)*STRESS(N1+2)))
C
      STRAIN(N1)=RATIO*DEPS(N1)
      STRAIN(N1+1)=RATIO*DEPS(N1+1)
      STRAIN(N1+2)=RATIO*DEPS(N1+2)
   20 CONTINUE
C
      DO 30 I=1,NN
      N1=2*(I-1)+1
      UN(N1)=RATIO*AL(N1)
      UN(N1+1)=RATIO*AL(N1+1)
   30 CONTINUE
C
```

```
      DO 40 I=1,NLN
      N1=2*(I-1)+1
      FX=FX+RATIO*ZAL(N1)
      FY=FY+RATIO*ZAL(N1+1)
   40 CONTINUE
C
      CALL OUTPUT (STRESS,STRAIN,ZRAMDA,FSO,UN,IEP,EEQ)
C
      RETURN
      END
C
C
      SUBROUTINE ZERO(EH,ZN2,FORC,DEPS,AL1)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
C
      DIMENSION EH(300),ZN2(300),FORC(900),DEPS(900),AL1(600)
C
      DO 10 NEL=1,NE
      EHAT=EH(NEL)
      GNEP=GOCR(EHAT)
      ZN2(NEL)=GNEP*GNEP-0.001
      L=3*(NEL-1)+1
      FORC(L)=0.0
      FORC(L+1)=0.0
      FORC(L+2)=0.0
      DEPS(L)=0.0
      DEPS(L+1)=0.0
      DEPS(L+2)=0.0
   10 CONTINUE
C
      DO 20 I=1,NN
      L=2*(I-1)+1
      AL1(L)=0.0
      AL1(L+1)=0.0
   20 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE RESID(FORC,STRESS,ZN2,IEP,AL,ZRAMDA,EH,RAMDA,
     &                 EHNP1,ICON,IB,ZAL,B,D,DSIGM,ARAMDA,AEH,IZAL)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /RR/  RA,RB,RC
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      DIMENSION FORC(900),STRESS(900),ZN2(300),IEP(300),AL(600),
     &          ZRAMDA(300),EH(300),ICON(900),IB(60),
     &          ZAL(60),B(300,3),D(300,3),ADF(3),DDF(3),
     &          DSIGM(900),ARAMDA(300),AEH(300),IZAL(20)
C
      DO 30 NEL=1,NE
      L=3*(NEL-1)+1
      SIG11=FORC(L)+STRESS(L)
      SIG22=FORC(L+1)+STRESS(L+1)
      SIG12=FORC(L+2)+STRESS(L+2)
      ZETO=SIG11*SIG11+SIG22*SIG22-SIG11*SIG22+3.0*(SIG12*SIG12)
```

```
      GN2EP=ZN2(NEL)
      IF(ZETO-GN2EP) 20,10,10
C
   10 IEP(NEL)=1
      GO TO 30
   20 IEP(NEL)=0
   30 CONTINUE
C
      DO 40 L=1,NN
      N1=2*(L-1)+1
      AL(N1)=0.0
      AL(N1+1)=0.0
   40 CONTINUE
C
      DO 50 IQ1=1,NE
      IQ=IQ1
      ID=IEP(IQ)
      IF(ID.EQ.0) GO TO 50
      L=3*(IQ1-1)+1
      ZR=ZRAMDA(IQ1)
C
      AP=1.0+ZR*RA
      BP=ZR*RB
      CP=1.0+ZR*RC
      AA=AP/(AP*AP-BP*BP)
      BB=BP/(BP*BP-AP*AP)
      CC=1.0/CP
      ADF(1)=AA*FORC(L)+BB*FORC(L+1)
      ADF(2)=BB*FORC(L)+AA*FORC(L+1)
      ADF(3)=CC*FORC(L+2)
C
      CALL PFORC(IQ,FORC,STRESS,EH,DDF,RAMDA,EHNP1)
C
      L=3*(IQ-1)+1
      DSIGM(L)=DDF(1)
      DSIGM(L+1)=DDF(2)
      DSIGM(L+2)=DDF(3)
      ARAMDA(IQ)=RAMDA
      AEH(IQ)=EHNP1
C
      NN1=2*(ICON(L)-1)+1
      NN2=2*(ICON(L+1)-1)+1
      NN3=2*(ICON(L+2)-1)+1
C
      AD1=ADF(1)-DDF(1)
      AD2=ADF(2)-DDF(2)
      AD3=ADF(3)-DDF(3)
C
      RX1=(AD1*B(IQ,1)+AD3*D(IQ,1))*0.5
      RY1=(AD3*B(IQ,1)+AD2*D(IQ,1))*0.5
      RX2=(AD1*B(IQ,2)+AD3*D(IQ,2))*0.5
      RY2=(AD3*B(IQ,2)+AD2*D(IQ,2))*0.5
      RX3=(AD1*B(IQ,3)+AD3*D(IQ,3))*0.5
      RY3=(AD3*B(IQ,3)+AD2*D(IQ,3))*0.5
C
      AL(NN1)=RX1+AL(NN1)
      AL(NN1+1)=RY1+AL(NN1+1)
      AL(NN2)=RX2+AL(NN2)
      AL(NN2+1)=RY2+AL(NN2+1)
      AL(NN3)=RX3+AL(NN3)
```

```
      AL(NN3+1)=RY3+AL(NN3+1)
   50 CONTINUE
C
      DO 60 L=1,NBN
      L1=3*(L-1)+1
      NB=IB(L1)
      NB1=2*(NB-1)+1
      AL(NB1)=AL(NB1)*FLOAT(IB(L1+1))
      AL(NB1+1)=AL(NB1+1)*FLOAT(IB(L1+2))
   60 CONTINUE
C
      DO 70 K=1,NLN
      L=2*(K-1)+1
      I=IZAL(K)
      N1=2*(I-1)+1
      A1=ZAL(L)
      A2=ZAL(L+1)
      AL(N1)=AL(N1)+DF*A1
      AL(N1+1)=AL(N1+1)+DF*A2
   70 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE PFORC(NEL,FORC,STRESS,EH,DDF,RAMDA,EHNP1)
C
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /RR/  RA,RB,RC
      COMMON /GC/  HC(20),EQPSNH(20),GOC(20),G1C(20)
C
      DIMENSION DDF(3),FORC(900),STRESS(900),EH(300)
C
      L=3*(NEL-1)+1
      EHAT=EH(NEL)
      RAMDA=0.0
      S1N=STRESS(L)
      S2N=STRESS(L+1)
      S3N=STRESS(L+2)
      SA1=FORC(L)+S1N
      SA2=FORC(L+1)+S2N
      SA3=FORC(L+2)+S3N
C
   10 A=1.0+RAMDA*RA
      B=RAMDA*RB
      C=1.0+RAMDA*RC
      AA=A/(A*A-B*B)
      BB=B/(B*B-A*A)
      CC=1.0/C
      SMA1=AA*SA1+BB*SA2
      SMA2=BB*SA1+AA*SA2
      SMA3=CC*SA3
      FSIGM=SMA1*SMA1+SMA2*SMA2-SMA1*SMA2+3.0*SMA3*SMA3
      FSIGM=SQRT(FSIGM)
      A=RAMDA*FSIGM
      I=2
   20 B=EQPSNH(I)
      IF(EHAT.LT.B) GOTO 30
      I=I+1
      GOTO 20
```

```
   30 J=I
   40 EQK=EQPSNH(J)
      X=EQK-EHAT
      GK=G1C(J)
      Y=A*GK
      IF(X.GT.Y) GOTO 50
      J=J+1
      GOTO 40
   50 GK1=G1C(J-1)
      EQK1=EQPSNH(J-1)
      CD=(GK1-GK)/(EQK-EQK1)
      X=EHAT+A*(GK1+CD*EQK1)
      X=X/(1.0+A*CD)
      XEQ=X-EQK1
      G1CRM=GK1-CD*XEQ
      G2CRM=-CD
      GEHP=GOC(J-1)+(GK1-0.5*CD*XEQ)*XEQ
      FRAM=FSIGM-GEHP
      SSIG1=SMA1-0.5*SMA2
      SSIG2=-0.5*SMA1+SMA2
      SSIG3=3.0*SMA3
      A=AA*SSIG1+BB*SSIG2
      B=BB*SSIG1+AA*SSIG2
      C=CC*SSIG3
      SDA=RA*A+RB*B
      SDB=RB*A+RA*B
      SDC=RC*C
      DFDRAM=-(SDA*SMA1+SDB*SMA2+SDC*SMA3)/FSIGM
      DXDRAM=(A*SMA1+B*SMA2+C*SMA3)/FSIGM
      DXDRAM=DXDRAM*G1CRM/(1.0-RAMDA*FSIGM*G2CRM)
      FCR=DFDRAM-G1CRM*DXDRAM
      RAMDA=RAMDA-FRAM/FCR
      D=ETA*FSIGM
      IF(FRAM.GT.D) GOTO 10
C
      EHNP1=RAMDA*FSIGM*G1CRM+EHAT
      DDF(1)=SMA1-S1N
      DDF(2)=SMA2-S2N
      DDF(3)=SMA3-S3N
C
      RETURN
      END
C
C
      SUBROUTINE CHECK(AL,AL1,RSD,RDEL)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
C
      DIMENSION AL(600),AL1(600)
C
      RSD=0.0
      RSE=0.0
C
      DO 10 I=1,NN
      N1=2*(I-1)+1
      ALN1=AL(N1)
      ALN11=AL1(N1)
      ALN2=AL(N1+1)
      ALN12=AL1(N1+1)
```

```
      AB=ABS(ALN1-ALN11)+ABS(ALN2-ALN12)
      IF(AB.GE.RSD) RSD=AB
      AB=ABS(ALN1)+ABS(ALN2)
      IF(AB.GE.RSE) RSE=AB
   10 CONTINUE
C
      RDEL=RSE*DELTA
C
      RETURN
      END
C
C
      SUBROUTINE SIGM(UN,AL,IEP,FORC,STRESS,EH,ZRAMDA,DEPS,
     &                STRAIN,EEQ,ZAL,DSIGM,ARAMDA,AEH)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /OUT/ IN,IO,IS1,IS2,IS3,IS4,ISTAT
      COMMON /PRO/ E,ANU,ZO,ETA,DELTA,DF,IFINAL
      COMMON /ELT/ NUE,FX,FY
C
      DIMENSION  FSO(300),UN(600),AL(600),IEP(300),
     &           FORC(900),STRESS(900),EH(300),ZRAMDA(300),
     &           DEPS(900),STRAIN(900),EEQ(300),ZAL(60),
     &           DSIGM(900),ARAMDA(300),AEH(300)
C
      DO 10 I=1,NN
      N1=2*(I-1)+1
      UN(N1)=UN(N1)+AL(N1)
      UN(N1+1)=UN(N1+1)+AL(N1+1)
   10 CONTINUE
C
      DO 50 I=1,NE
      NEL=I
      N1=3*(NEL-1)+1
      INDEX=IEP(NEL)
      IF(INDEX) 30,30,20
C
   20 STRESS(N1)=STRESS(N1)+DSIGM(N1)
      STRESS(N1+1)=STRESS(N1+1)+DSIGM(N1+1)
      STRESS(N1+2)=STRESS(N1+2)+DSIGM(N1+2)
      STRAIN(N1)=STRAIN(N1)+DEPS(N1)
      STRAIN(N1+1)=STRAIN(N1+1)+DEPS(N1+1)
      STRAIN(N1+2)=STRAIN(N1+2)+DEPS(N1+2)
C
      ZRAMDA(NEL)=ARAMDA(NEL)
      EHN=EH(NEL)
      EHN1=AEH(NEL)
      DEPN=EHN1-EHN
      EH(NEL)=EHN1
      GC=G1CR(EHN)
      GC1=G1CR(EHN1)
      EEQ(I)=EEQ(I)+0.5*(1.0/GC1+1.0/GC)*DEPN
      GO TO 40
C
   30 STRESS(N1)=STRESS(N1)+FORC(N1)
      STRESS(N1+1)=STRESS(N1+1)+FORC(N1+1)
      STRESS(N1+2)=STRESS(N1+2)+FORC(N1+2)
      STRAIN(N1)=STRAIN(N1)+DEPS(N1)
      STRAIN(N1+1)=STRAIN(N1+1)+DEPS(N1+1)
      STRAIN(N1+2)=STRAIN(N1+2)+DEPS(N1+2)
```

```
      ZRAMDA(NEL)=0.0
C
   40 FSO(I)=SQRT(STRESS(N1)*STRESS(N1)+STRESS(N1+1)*
     &        STRESS(N1+1)-STRESS(N1)*STRESS(N1+1)+
     &        3.0*(STRESS(N1+2)*STRESS(N1+2)))
C
   50 CONTINUE
C
      DO 60 K=1,NLN
      N1=2*(K-1)+1
      FX=FX+ZAL(N1)*DF
      FY=FY+ZAL(N1+1)*DF
   60 CONTINUE
C
      CALL OUTPUT (STRESS,STRAIN,ZRAMDA,FSO,UN,IEP,EEQ)
C
      RETURN
      END
C
C
      SUBROUTINE OUTPUT (STRESS,STRAIN,ZRAMDA,FSO,UN,IEP,EEQ)
C
      COMMON /INP/ NN,NE,NLN,NBN,N,MS
      COMMON /ELT/ NUE,FX,FY
      COMMON /OUT/ IN,IO,IS1,IS2,IS3,IS4,ISTAT
C
      DIMENSION STRESS(900),STRAIN(900),ZRAMDA(300),
     &          FSO(300),UN(600),IEP(300),EEQ(300)
C
      IF(ISTAT) 10,10,30
C
   10 WRITE(IO,20) FX,FY
   20 FORMAT(///2X,"******* ELASTIC SOLUTION  *******"
     &       ///35X,"X - LOAD = ",1PE12.6,14X,"Y - LOAD = ",1PE12.6)
      GOTO 40
   30 WRITE(IO,50) ISTAT,FX,FY,NUE
      WRITE(IO,60)(IEP(I),I=1,NE)
   40 WRITE(IO,70)
      WRITE(IO,80) (I,UN(2*(I-1)+1),UN(2*(I-1)+2),I=IS1,IS2)
      WRITE(IO,90)
      WRITE(IO,100) (K,STRESS(3*(K-1)+1),STRESS(3*(K-1)+2),
     &              STRESS(3*(K-1)+3),FSO(K),
     &              STRAIN(3*(K-1)+1),STRAIN(3*(K-1)+2),
     &              STRAIN(3*(K-1)+3),EEQ(K),
     &              ZRAMDA(K),K=IS3,IS4)
      WRITE(IO,110)
C
   50   FORMAT(///2X,"*******  STEP  ",I3,"  *******"
     &       ///35X,"X - LOAD = ",1PE12.6,14X,"Y - LOAD = ",1PE12.6
     &       ///51X,"ITERATION NUMBER = ",I7//)
   60 FORMAT(13X,5I3,2X,5I3,2X,5I3,2X,5I3,2X,5I3,2X,5I3)
   70 FORMAT(///53X," DISPLACEMENT  "
     &       ///15X,"NODE",10X,"X",15X,"Y",30X,
     &          "NODE",10X,"X",15X,"Y",//)
   80 FORMAT(15X,I4,4X,1PE12.5,4X,1PE12.5,25X,
     &       I4,4X,1PE12.5,4X,1PE12.5)
   90 FORMAT(///53X," STRESS  AND  STRAIN  "
     &        ///2X,"ELEMENT NO. ",6X,"SX",11X,"SY",10X,"SXY"
     &          10X,"SEQ",11X,"EX",11X,"EY",10X,"EXY",
     &           10X,"EEQ",9X,"RAMDA"//)
```

```
  100 FORMAT(5X,I4,6X,1PE11.4,2X,1PE11.4,2X,1PE11.4,2X,1PE11.4,
     &          2X,1PE11.4,2X,1PE11.4,2X,1PE11.4,2X,1PE11.4,
     &          2X,1PE11.4)
  110 FORMAT(///5X,12("----------")//)
C
      RETURN
      END
```

Table 1. An example of input data

5	4	2	4	21000.0		0.3		
	32.0		4.0		0.1E-2	0.01	8	20
1		0.0		32.000				
2		0.01		37.000				
3		0.025		41.600				
4		0.045		46.400				
5		0.070		50.900				
6		0.100		55.400				
7		0.130		58.658				
8		0.195		61.908				
1		0.0		0.0				
2		0.0		5.0				
3		2.5		2.5				
4		5.0		0.0				
5		5.0		5.0				
1	1	3	2					
2	1	4	3					
3	3	5	2					
4	3	4	5					
	1		0		0			
	2		0		1			
	4		0		0			
	5		0		1			
	2		0.0		1.0			
	5		0.0		1.0			
	1		4					
	1		2