3/31/81

ME235B Advanced Finite Elements.

Conduct of Course:

HW's (8%); Computing; Etc < TA Reader

Midterm 1 hr in class

Final 3 hr in class

IBM 3033 (C.I.T.) can access evening + weekend
1½ hr constraint per session can relog on thru
6 hr/wk

Content: 1. Finish Linear Statics

2. Linear Dynamics

Overview of Final

expect on midterm { δ fns will reappear again.
Barlow stress point problem "

σ^0, ε^0 (last problem) become part of source term.

Ch 8 §9: Calculation of derivatives of shape fns (since they appear in our bilinear forms).

ie $N_{a,x}$: we normally derived $N_a(\underset{\sim}{\xi})$ where $\underset{\sim}{\xi}$ are the natural coordinates

it is also impossible to get the inverse map of $\underset{\sim}{\xi}(\underset{\sim}{x})$ from $\underset{\sim}{x}(\underset{\sim}{\xi})$ especially in multi dim.

let $n_{sd} = 2$ for our purpose

$$N_{a,x} = N_{a,\xi}\,\xi_{,x} + N_{a,\eta}\,\eta_{,x}$$

$$N_{a,y} = N_{a,\xi}\,\xi_{,y} + N_{a,\eta}\,\eta_{,y}$$

$$\langle N_{a,x} \quad N_{a,y} \rangle = \langle N_{a,\xi} \quad N_{a,\eta} \rangle \begin{bmatrix} \xi_{,x} & \xi_{,y} \\ \eta_{,x} & \eta_{,y} \end{bmatrix} = N_{a,\underset{\sim}{\xi}}\ \underset{\sim}{\xi}_{,\underset{\sim}{x}}$$

jacobian of inverse map

we normally have $\underset{\sim}{x}(\underset{\sim}{\xi}) = \sum_{a=1}^{nen} N_a(\underset{\sim}{\xi})\,\underset{\sim}{x}_a^e$ $\quad \underset{\sim}{x} = (x,y)$ $\quad \underset{\sim}{\xi} = (\xi,\eta)$

we can compute $\underset{\sim}{x}_{,\underset{\sim}{\xi}} = \begin{bmatrix} x_{,\xi} & x_{,\eta} \\ y_{,\xi} & y_{,\eta} \end{bmatrix}$ since we have $\underset{\sim}{x}(\underset{\sim}{\xi})$.

ie $x_{,\xi} = \sum_{a=1}^{4} N_{a,\xi}\, x_a^e \qquad x_{,\eta} = \sum_{a=1}^{4} N_{a,\eta}\, x_a^e \qquad y_{,\xi} = \sum N_{a,\xi}\, y_a^e \qquad y_{,\eta} = \sum N_{a,\eta}\, y_a^e$

now note $\underset{\sim}{\xi}_{,\underset{\sim}{x}} = \left[\underset{\sim}{X}_{,\underset{\sim}{\xi}}\right]^{-1} = \begin{bmatrix} y_{,\eta} & -x_{,\eta} \\ -y_{,\xi} & x_{,\xi} \end{bmatrix} \cdot \frac{1}{j}$

where $j = \det \underset{\sim}{X}_{,\underset{\sim}{\xi}} = x_{,\xi} y_{,\eta} - x_{,\eta} y_{,\xi}$.

These are calculations performed in shape fn subroutine. We will now look at such a subroutine. The one to be handed out will be more involved.

"Shape fn subroutine"

its output will (or can) be $j, Na, Na_{,x}, Na_{,y}$ evaluated at some point $(\tilde{\xi}, \tilde{\eta})$ for $1 \le a \le n_{en}$

likely to be at quadrature point.

the input would be the point $(\tilde{\xi}, \tilde{\eta})$ also $x_a^e, y_a^e \quad 1 \le a \le n_{en}$

in flowchart form:

1. Given : input
2. Calculate : evaluate $Na(\tilde{\xi}, \tilde{\eta})$; $Na_{,\xi}(\tilde{\xi}, \tilde{\eta})$; $Na_{,\eta}(\tilde{\xi}, \tilde{\eta})$ $\quad 1 \le a \le n_{en}$ — Na's are already in code
3. $x_{,\xi}(\tilde{\xi}, \tilde{\eta}) = \sum N_{a,\xi}(\tilde{\xi}, \tilde{\eta})\, x_a^e$ also for $x_{,\eta}$; $y_{,\xi}$; $y_{,\eta}$ — $N_{a,\xi}$ are " in code.

4. Compute $j(\tilde{\xi}, \tilde{\eta}) = x_{,\xi} y_{,\eta} - x_{,\eta} y_{,\xi}$ using results of (3).
5. Now we can calculate $\langle Na_{,x} \quad Na_{,y} \rangle$ using ie

$$\left.\begin{aligned} Na_{,x} &= (Na_{,\xi}\, y_{,\eta} - Na_{,\eta}\, y_{,\xi})/j \\ N_{a,y} &= -(N_{a,\xi}\, x_{,\eta} - N_{a,\eta}\, x_{,\xi})/j \end{aligned}\right\} \quad 1 \le a \le n_{en}$$

6. return to next step

Exercise do the same for $Na_{,\underset{\sim}{x}}$ for $\underset{\sim}{x} = x, y, z$

ie $Na_{,\underset{\sim}{x}} = Na_{,\underset{\sim}{\xi}}\ \underset{\sim}{\xi}_{,\underset{\sim}{x}}$ where $\underset{\sim}{\xi}_{,\underset{\sim}{x}}$ is 3-D j where $\underset{\sim}{\xi}_{,\underset{\sim}{x}} = \left[\underset{\sim}{X}_{,\underset{\sim}{\xi}}\right]^{-1}$

where $\left[\underset{\sim}{X}_{,\underset{\sim}{\xi}}\right]^{-1} = \dfrac{[\text{cofactor } \underset{\sim}{X}_{,\underset{\sim}{\xi}}]^T}{j}$

ie cofactor $= y_{,\eta} z_{,\varepsilon} - y_{,\varepsilon} z_{,\eta}$ | | | cofactor of 11

now $j = x_{,\xi}\, cof_{11} + x_{,\eta}\, cof_{12} + x_{,\zeta}\, cof_{13}$

and we follow this through as before.

§10. Programming Aspects in FORTRAN

"Shape fn. subroutine" - these names may appear world wide. we will do this for 4 node bilin quad

Our notation	Fortran	Array Dim	Our notation	Fortran names	Array Dim
n_{sd}	NSD(=2)		N_a	SH(3,I)	(NSD+1, NEN)
n_{en}	NEN(=4)		$N_{a,x}$	SH(1,I)	
quad points $\hat{\xi}_l$	S		$N_{a,y}$	SH(2,I)	
quad points $\tilde{\eta}_l$	T (actually I on a)		$\xi_a/2$	SA(I)	NEN (values in data statem)
x^e_a, y^e_a	XE(1,I) XE(2,I)	(NSD, NEN)	$\eta_a/2$	TA(I)	NEN
	XE or "XL"		$\begin{bmatrix} x_{,\xi} & x_{,\eta} \\ y_{,\xi} & y_{,\eta} \end{bmatrix}$	XS(I,J)	(NSD, NSD)
j	DET				

Coding col 7

```
                      input   output
      SUBROUTINE SHAPE (S,T, XE, DET, SH)
C ...
C ...                 ROUTINE TO CALCULATE
C ...
C ...
      DIMENSION  SA(4), TA(4), XS(2,2), XE(2,4), SH(3,4)
      DATA SA/-.5,+.5, .5,-.5/, TA/-.5,-.5, .5, .5/
      DATA ZERO, PS/0.0, 0.5/
C ...
C ...                 SHAPE FNS & DERIV IN LOCAL COORDS.
C ...
C ...
        DO  10  I=1,4
temporary storage of SH  SH(1,I) = PS + TA(I)*T  = 1/2 (1 + η_a η)
                         SH(2,I) = PS + SA(I)*S  = 1/2 (1 + ξ_a ξ)
                         SH(3,I) = SH(1,I) * SH(2,I) = N_a(ξ,η)
                         SH(1,I) = SA(I) * SH(1,I)   = N_{a,ξ} = ξ_a/2 · 1/2 (1+η_a η)
   10                    SH(2,I) = TA(I) * SH(2,I)   = N_{a,η} = η_a/2 · 1/2 (1+ξ_a ξ)
C ...
C ...                   jacobian matrix  (x_{i,j})
C ...
        DO 20 I=1,2
        DO 20 J=1,2
          XS(I,J) = ZERO
```

```
20    XS(I,J) = XS(I,J) + XE(I,K) * SH(J,K)
C...
C...  JACOBIAN DETERMINANT                                  Continuation   4/2/81
C...
      DET= XS(1,1) * XS(2,2) - XS(1,2)* XS(2,1)
C...
C...  SHAPE FN DERIVS IN GLOBAL COORDS
C...
      DO 30 I=1,4
      TEMP =    (XS(2,2) * SHAPE(1,I) - XS(2,1) * SHAPE(2,I))/DET   This is Na,x
      SH(2,I) = (-XS(1,2)* SH(1,I) + X(1,1) * SH(2,I)) /DET         This is Na,y
30    SH(1,I) = TEMP
C...
      RETURN
      END
```

$x_a^e \to$ (I=1), $y_a^e \to$ (I=2); $N_{a,x}$ (J=1), $N_{a,y}$ (J=2)

$N_{a,x} = N_{a,\xi}\,\xi_{,x} + N_{a,\eta}\,\eta_{,x}$

Now looking at the element stiffness formation for 2D, heat conduction element.

k^e_{ab} (a, b are local nodal indices = local eqn nos.) $= \int_{\Omega^e} \underset{\sim}{B}_a^T \underset{\sim}{D}\, \underset{\sim}{B}_b \, d\Omega$ — since 1 DOF/node

using numerical integration we had defined

$$\int_{\Omega^e} d\Omega = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\, j(\xi,\eta)\, d\xi d\eta = \int_{-1}^{1}\int_{-1}^{1} g(\xi,\eta)\, d\xi d\eta$$

for $n_{sd}=2$ $\quad \underset{\sim}{B}_a = \langle N_{a,x} \;\; N_{a,y} \rangle^T$ and $\underset{\sim}{D} = \underset{\approx}{K} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}$ w/ $K_{ij} = K_{ji}$

thus $$\int_{-1}^{1}\int_{-1}^{1} g(\xi,\eta)\, d\xi d\eta \cong \sum_{\ell=1}^{n_{int}} \left(\underset{\sim}{B}_a^T \underset{\sim}{D}\, \underset{\sim}{B}_b\, j\right)\Big|_{(\tilde\xi_\ell, \tilde\eta_\ell)} W_\ell$$

Now we will define $\underset{\sim}{\tilde D}_\ell = \underset{\sim}{D}\, j\, W_\ell$ $\qquad k^e_{ab} \cong \sum_{\ell=1}^{n_{int}} \left(\underset{\sim}{B}_a^T \underset{\sim}{\tilde D}\, \underset{\sim}{B}_b\right)_\ell$ by writing $\underset{\sim}{\tilde D}$ over $\underset{\sim}{D}$

we will define our fortran variable names

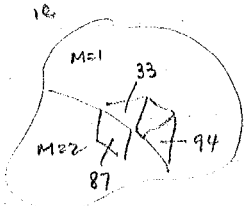
NOTATION	FORTRAN NAME	ARRAY DIM	NOTATION	FORTRAN NAME	ARRAY DIM
a, b	I, J		n_{en}	NEN	
e	N		material set nos.	M = MAT(N)	NUMEL
n_{el}	NUMEL		where M = no. of material sets		
n_{sd}	NSD (= 2)		no. of material sets	NUMAT	
n_{int}	NINT		material property array	C(I,M)	(3, NUMAT)
			ie K_{11} K_{12} K_{22}	C(1,M) C(2,M) C(3,M)	

Notation	Fortran Name	Array Dim	Notation	Fortran Name
k^e_{ab}	S(I,J)	(NEN,NEN)	D_{11} D_{12} D_{22}	D11, D12, D22
ξ_L, η_L	SG(L), TG(L)	NINT	$(\tilde{D}\tilde{B})_1$; $(\tilde{D}\tilde{B}_b)_2$ (2×1)	DB1, DB2
W_L	W(L)	NINT		
$N_{a,x}$	B1 = SH(1,I)	(NSD+1, NEN)	X^e_a, Y^e_a	XE(1,I) XE(2,I) (NSD,NEN)
$N_{a,y}$	B2 = SH(2,I)			
N_a	SH(3,I)			

(since we will interface this w/ main prog)

The following is general as far as coding style. We will not give full code, but we will assume all arrays are dimensioned appropriately, e.g.
NUMAT, NUMEL, NSD, NINT, MAT(NUMEL), C(3,MAT), SG(NINT), TG(NINT), W(INT)
are all given ie gauss quadrature is already picked.

ie



M=MAT(87)=2

```
C
C... ELEMENT LOOP
C
      DO 100 N=1,NUMEL
C
C     OBTAIN PROP. SET. NO.
C
      M=MAT(N)
C
C...  INITIALIZE EL STIFF.
C
      CALL CLEAR (S, NEN*NEN)        this sets S(I,J)=0.
C...
C...  OBTAIN EL COORDS OF LOCALIZED COORDS
C...
      CALL LCOORD(                   )
C...
C...  BEGIN INTEG LOOP
C...
      DO 10 L=1,NINT
C..
C..   OBTAIN SHAPE FNS, DERIV, DET
C...  CALL SHAPE (SG(L), TG(L), XE, DET, SH)
                  INPUT
C
C...  COMPUTE D MATRIX  (really D̃ matrix)
C
      DET = DET*W(L)                 ... (j We)
      D11 = C(1,M)*DET
      D12 = C(2,M)*DET               } this assumes that material prop/element = constant.
      D22 = C(3,M)*DET
C
C...  compute D̃*B
C
      DO 10 J=1,NEN                  column loop
      B1 = SH(1,J) =  Na,x
      B2 = SH(2,J) =  Na,y
C
      DB1 = D11*B1 + D12*B2          K11 j Na,x + K12 j Na,y
      DB2 = D12*B1 + D22*B2          K12 j Na,x + K22 j Na,y
C
C.... compute BᵀDB
C
      DO 10 I=1,J                    Since we have symmetry (row loop)   if we had no sym then replace J by N
```

```
      B1 = SH(1,I)
      B2 = SH(2,I)
10    S(I,J) = S(I,J) + B1*DB1 + B2*DB2
C
C... Compute lower triangular part.
      CALL SYMM (S, NEN).
C
C... ASSEMBLE S INTO GLOBAL STIFF
100   CALL ADDSTF (     )
```

for elasticity, we would calculate all elements of the block k^e_{ab} here.

pass in part of LM array dealing W/ this particular element.

this matrix $\underset{\sim}{K} = A\,\underset{\sim}{k}^e_{ab}$ (from last quarter).

Now we will do same for elasticity in 2D less further.

remember that $\underset{\sim}{k}^e$ was formed in terms of nodal submatrices $\underset{\sim}{k}^e_{ab}$

Element Stiffness Formulation for a 2-D elasticity element. 4/7/81

$\underset{\sim}{k}^e$ can be written in terms of $n_{en} \times n_{en}$ nodal submatrices $\underset{\sim}{k}^e_{ab}$
ie eqns (6)–(9) in § 2.8

$$\underset{n_{ed}\times n_{ed}}{\underset{\sim}{k}^e_{ab}} = \int_{\Omega^e} \underset{\sim}{B}_a^T \underset{\sim}{D} \underset{\sim}{B}_b \, d\Omega \qquad (7)$$

now we use the same format as in heat conduction by using gauss quadr

$$\underset{\sim}{k}^e_{ab} \doteq \sum_{l=1}^{n_{int}} \left(\underset{\sim}{B}_a^T \underset{\sim}{\tilde{D}} \underset{\sim}{B}_b \right)_e \quad \text{where} \quad \underset{\sim}{\tilde{D}} = \underset{\sim}{D}\, j\, W_l$$

in 2-D

$$\underset{\sim}{k}^e_{ab} \text{ is } 2\times 2 = \begin{bmatrix} k^e_{2a-1,2b-1} & k^e_{2a-1,2b} \\ k^e_{2a,2b-1} & k^e_{2a,k_{2b}} \end{bmatrix}$$

where these elements are refered to the entries of $\underset{\sim}{k}^e$

$1 \le a, b \le n_{en}$

because of the symmetry of $\underset{\sim}{k}^e$ we need only the nodal submatrices on or above the diagonal.

Before the flowcharting we will assume that $D_{13} = D_{23} = 0$
ie the material properties are constant over the material

Flow chart

1. Loop on the number of elements $1 \le e \le n_{el}$
   a. Obtain the element material properties set number
   b. Initialize the element stiffness matrix
   c. Obtain element nodal coordinates
2. Loop on the number of integration points $1 \le l \le n_{int}$
   a. Obtain shape fcn, derivatives, jacobian
   b. Calculate $\underset{\sim}{D} = (D j W)_l$
3. Loop on nodes $1 \le b \le n_{en}$
   a. Form $\underset{\sim}{D}\,\underset{\sim}{B}_b$ where $\underset{\sim}{B}_b = \begin{pmatrix} N_{b,x} & 0 \\ 0 & N_{b,y} \\ N_{b,y} & N_{b,x} \end{pmatrix}$ $\quad \underset{\sim}{D}\underset{\sim}{B}_b = \begin{pmatrix} (DB)_{11} & (DB)_{12} \\ (DB)_{21} & (DB)_{22} \\ (DB)_{31} & (DB)_{32} \end{pmatrix} = \begin{bmatrix} D_{11}B_1 & D_{12}B_2 \\ D_{12}B_1 & D_{22}B_2 \\ D_{33}B_2 & D_{33}B_1 \end{bmatrix}$

   $3\times3 \cdot 3\times2 = 3\times2$

   $\begin{bmatrix} \lambda+2\mu & \lambda & 0 \\ \lambda & \lambda+2\mu & 0 \\ 0 & 0 & \mu \end{bmatrix}$
4. Loop on nodes $1 \le a \le b$
   a. Form $\underset{\sim}{B}_a^T (\underset{\sim}{D}\underset{\sim}{B}_b)$

$$\begin{bmatrix} k^e_{2a-1,2b-1} & k^e_{2a-1,2b} \\ k^e_{2a,2b-1} & k^e_{2a,2b} \end{bmatrix} \xleftarrow{\text{replaces}} \begin{bmatrix} k^e_{2a-1,2b-1} & k^e_{2a-1,2b} \\ k^e_{2a,2b-1} & k^e_{2a,2b} \end{bmatrix} + \begin{pmatrix} B_1(DB)_{11}+B_2(DB)_{21} & B_1(DB)_{12}+ \\ & B_2(DB)_{22} \\ B_2(DB)_{31}+B_1(DB)_{31} & B_2(DB)_{12}+ \\ & B_1(DB)_{32} \end{pmatrix}$$

5. Define the lower triangular part by symmetry
6. Assembly into global stiffness using IEN arrays

Remarks

- Forces vectors are formulated outside this sequence except gravity forces and except for: the part of the element force due to prescribed displacement BC

$$f_p^e = \cancel{\int_{\Omega^e} N_a f_i \, d\Omega} + \int_{\Gamma_{h_i}^e} N_a h_i \, d\Gamma - \sum_{q=1}^{neq} k_{pq}^e g_q^e$$

see eq. (5) of § 2.3

References: "Computer Procedures for finite Element Analysis" of Chapter 24 in O. C. Zienkiewicz The finite Element Method McGraw-Hill, London 197[illegible] by R. L. Taylor

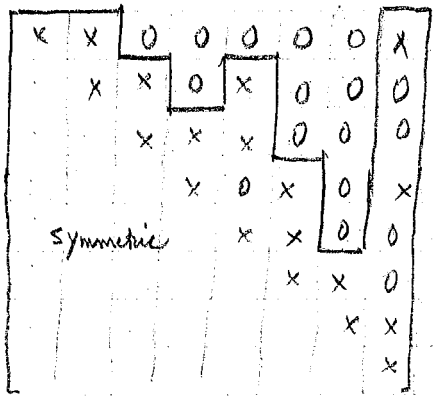
Hinton & Owen – Finite Element Programming, Academic Press, London 19[illegible]

LEARN program allows either use of existing or adding changes

Definition The data structure is the mode of storage, location and history of data employed.

- compacted column structure - used to keep only non zero elements. "skyline," "profile", "active column"
- Dynamic Storage Allocation - arrays are set during problem execution
- Element Groups (for each different material set no. inputed simultaneously).

example of $[k^e]$ which is 8×8 let x be nonzero, 0 " " $\underset{\sim}{k}^e$ will be stored in 1-D array called A.



⌐⌊ - profile or skyline

in LEARN - location of diag is storage in IDIAG
dimensions of IDIAG is NEQ.
Dimension of A is NA = Σ of column heights
1+2+2+2+4+3+2+8

col 2 col 3

Thus A(1) = K11 A(2) = K12 A(3) = K22 A(4) = K32 A(5) = K33 .... A(24) = K88

The heights actually depend on $\underset{\sim}{D}$ and since we know $\underset{\sim}{D}$ we know the structure of $\underset{\sim}{K}$ a priori

4/9/81

ARRAY IDIAG gives last element location of column I ie IDIAG(I) = L ; for the above

IDIAG(1) = 1 IDIAG(2) = 3 IDIAG(3) = 5 IDIAG(4) = 7 IDIAG(5) = 11
IDIAG(6) = 14 IDIAG(7) = 16 IDIAG(8) = 24

This allows us to compact the LHS of the equation $\underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{f}$

The rhs of the equation is stored in B ∴ $f_i = B(I)$

Let $\underset{\sim}{K}$ be symmetric. Then ∃ a nonsingular upper triangular matrix $\underset{\sim}{U}$ with unit diagonal entries and a diagonal matrix $\underset{\sim}{D}$ ∋.

$$\underset{\sim}{K} = \underset{\sim}{U}^T \underset{\sim}{D}\, \underset{\sim}{U}$$

The number of negative, zero & positive eigenvalues of $\underset{\sim}{K}$ is equal to the no. of negative, zero, positive diagonal entries of $\underset{\sim}{D}$ respectively.

Storage in blank common

Except for element group data, which is stored on a disk file and read in and out of core as needed, all of the main arrays are stored in blank common. This includes both integer and floating point data. In the main program and subroutine LEARN, blank common is denoted by the one-dimensional array A.

The locations and lengths of arrays in blank common are as follows:

Storage in Blank Common - Solution Phase

First Word	Array	
N1 = 1	ID(NDOF, NUMNP)	
N2 = N1 + NDOF*NUMNP	X(NSD, NUMNP)	takes into account prescribed bdry condition
N3 = N2 + NSD*NUMNP*IPREC	F(NDOF, NUMNP) (=D(NDOF, NUMNP))	
N4 = N3 + NDOF*NUMNP*IPREC	IDIAG(NEQ)	
N5* = N4 + NEQ	B(NEQ)	
N6 = N5 + NEQ*IPREC	A(NA)	N_A is sum of column heights
N7 = N6 + NA*IPREC	E(MAX)	

N8 = N7 + MAX-1 (= total required storage; must be .LT. MTOT for execution.)

*During input phase, N1 to N5 are the same as above and element group data begins in N5. Individual element routines determine necessary storage for element groups.

where

ID(M,N)	=	equation number for degree of freedom M, node N
X(M,N)	=	Mth coordinate of node N
F(M,N), D(M,N)	=	Mth direction force/displacement component of node N ($1 \le N \le NUMNP$, $1 \le M \le NDOF$)
IDIAG(N)	=	location of Nth diagonal element in coefficient array A.*
B(N)	=	Nth force/displacement component
A(N)	=	Nth element of coefficient matrix stored in compacted column form
E(N)	=	Nth (single precision) word of element group array
NUMNP	=	number of nodes
NSD	=	number of space dimensions
NDOF	=	number of degrees of freedom
NEQ	=	number of equations
NA	=	number of terms in A
MAX	=	number of single-precision words in longest element group data file
IPREC	=	precision needed (1/2 for single/double precision math).

* This array name is used for the coefficient matrix in various subroutines and should not be confused with the array A, denoting blank common.

Storage Requirements of Plane Stress/Strain Element (NTYPE=3)

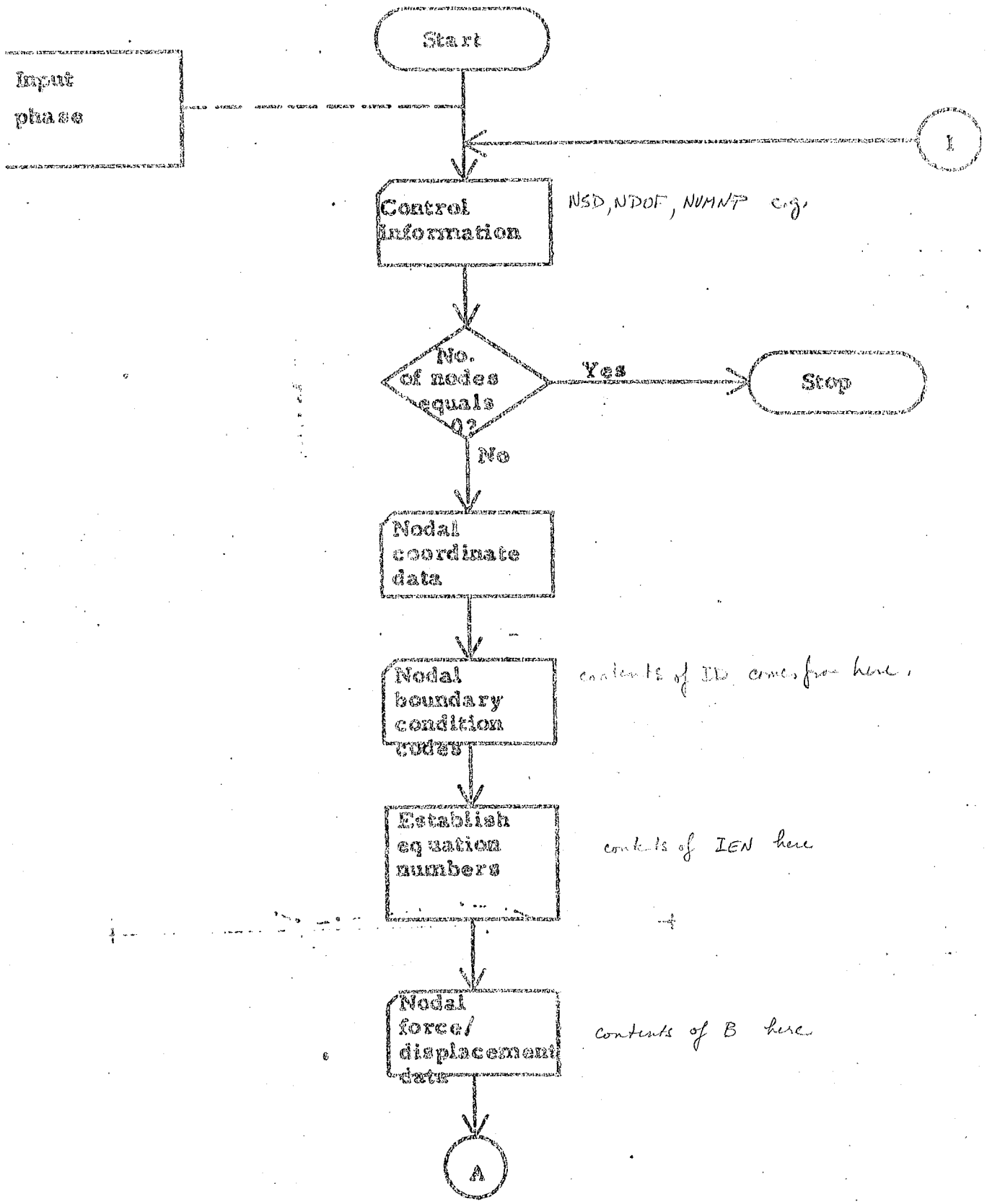
Fixed Word		Array	
N101 = NF*		E(NUMAT)	Young's Mod
N102 = N101 + NUMAT*IPREC		POIS(NUMAT)	ν
N103 = N102 + NUMAT*IPREC		WT(NUMAT)	weight density for gravity calculations
N104 = N103 + NUMAT*IPREC		TH(NUMAT)	thickness
N105 = N104 + NUMAT*IPREC		GRAV(2)	
N106 = N105 + 2*IPREC		MAT(NUMEL)	material set no.
N107 = N106 + NUMEL		IEN(4, NUMEL)	4 node quad.
N108 = N107 + 4*NUMEL		LM(2, 4, NUMEL)	NSD = 2
N109 = N108 + 8*NUMEL		IELNO(NUMPR)	
N110 = N109 + NUMPR		ISIDE(NUMPR)	
N111 = N110 + NUMPR		PRES(NUMPR)	pressure on side
NL = N111 + NUMPR*IPREC			

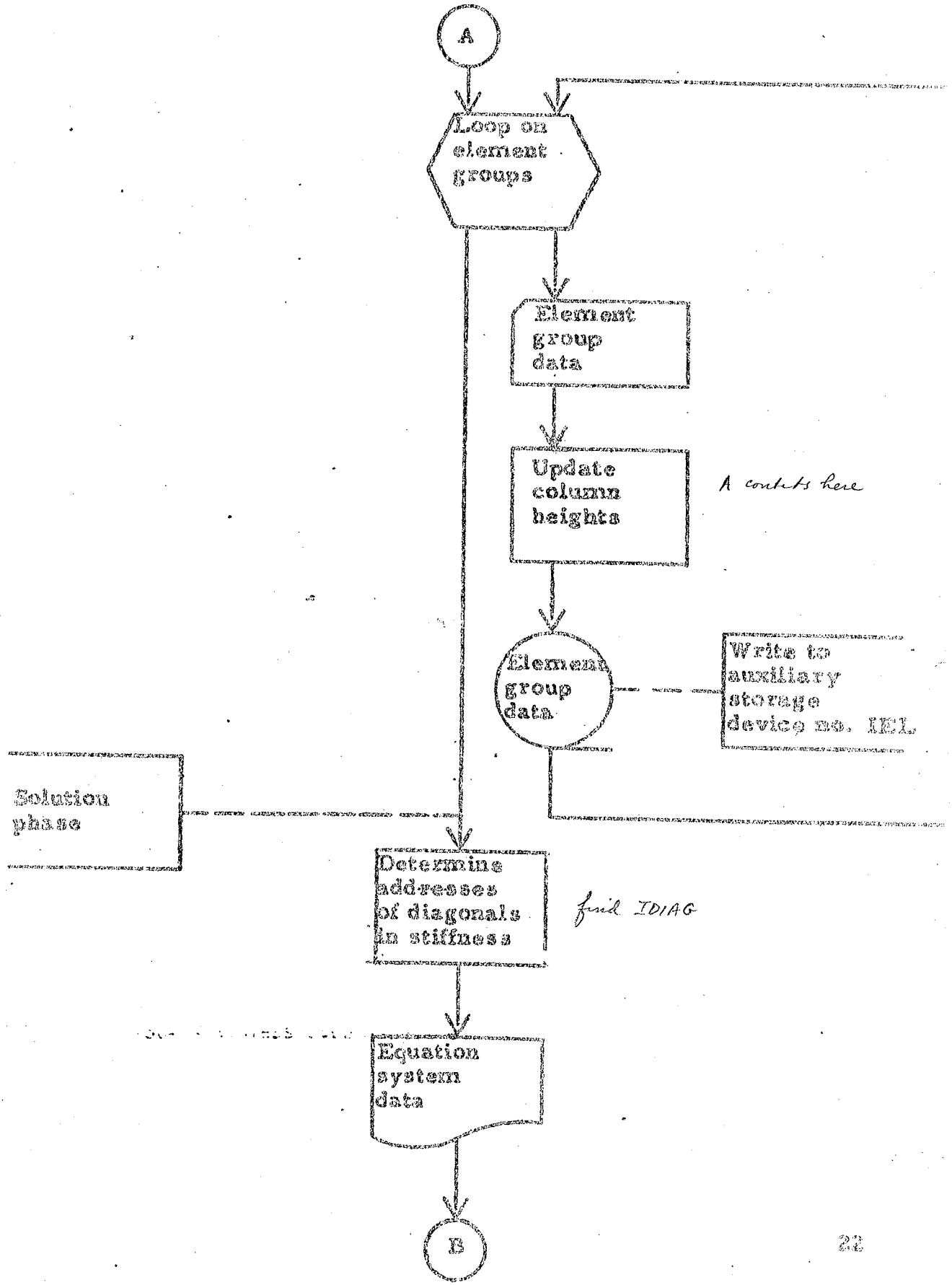
LENGTH = NL - NF (=total required storage for element group)

The meaning of the arrays here may be deduced from the description in the section entitled "User's Manual."

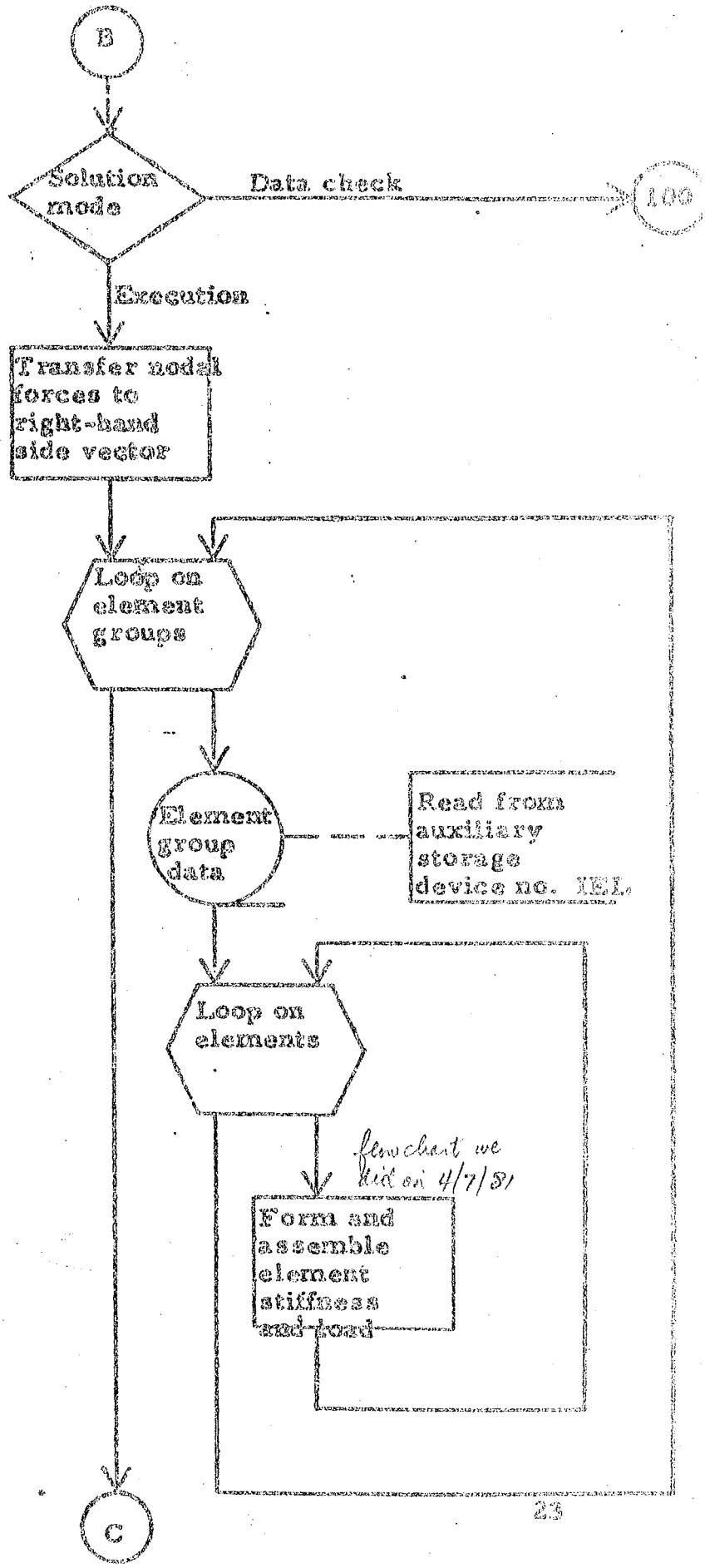
$MAX = \max_i (NL - NF)_i$

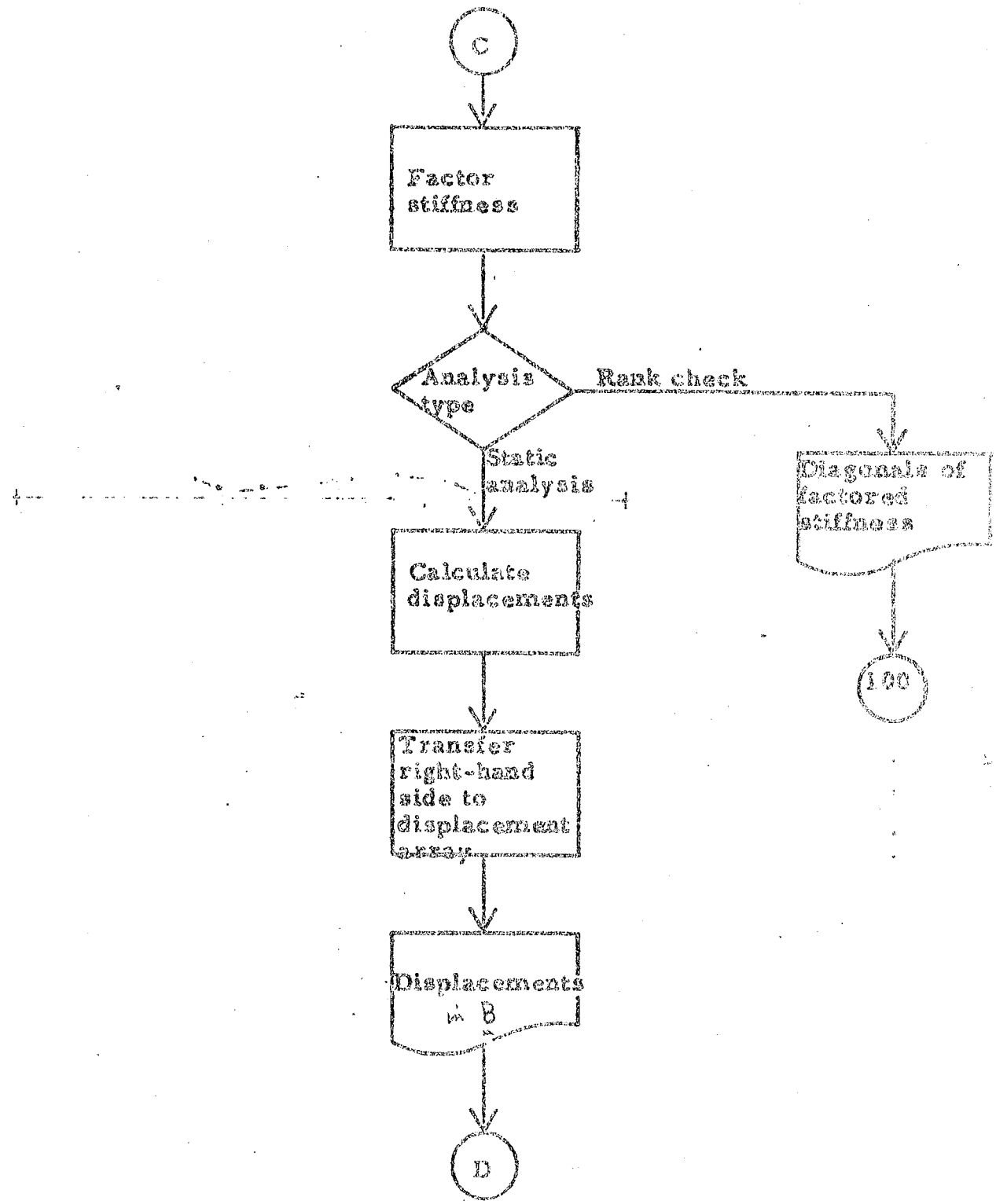
*NF = N5 during input phase and NF = N7 during solution phase.

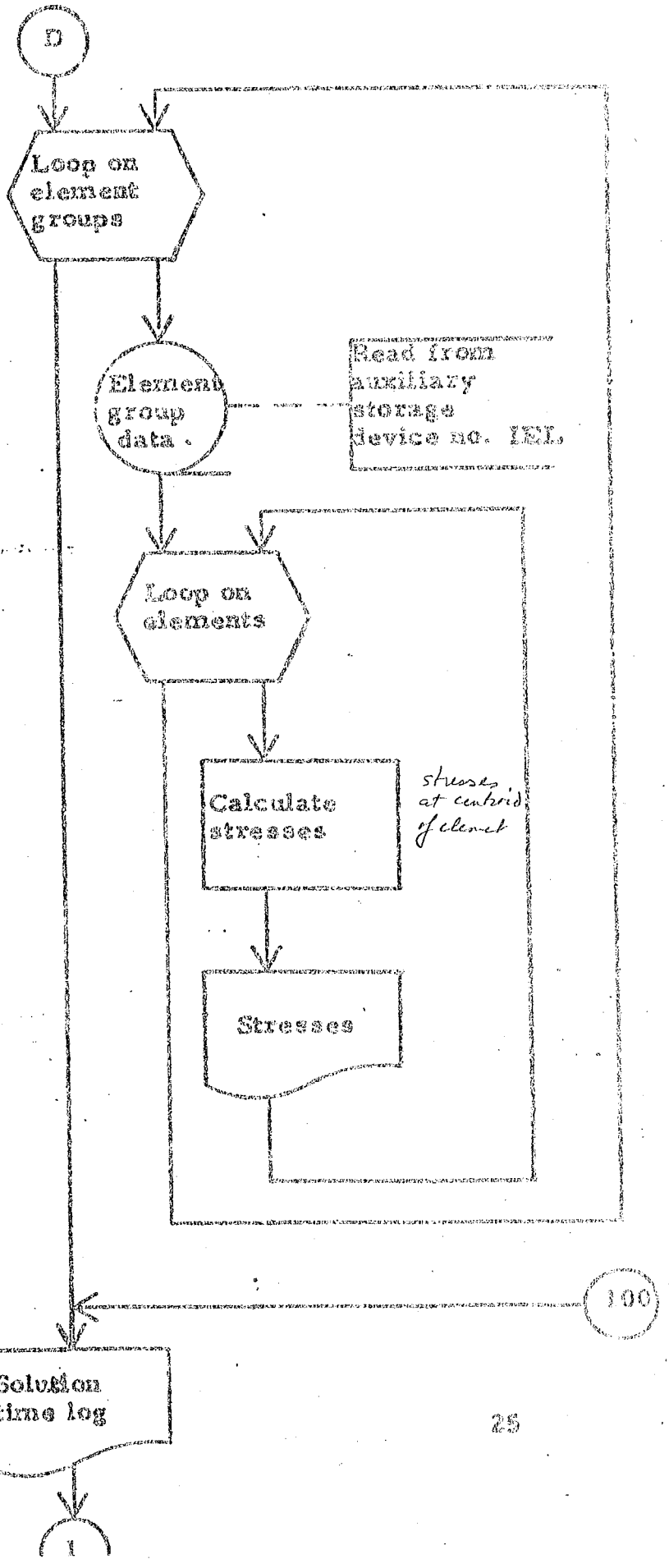

Start
Input phase
1
Control information
NSD, NDOF, NUMNP e.g.
No. of nodes equals 0?
Yes
Stop
No
Nodal coordinate data
Nodal boundary condition codes
contents of ID comes from here.
Establish eq nation numbers
contents of IEN here
Nodal force/ displacement data
contents of B here
A


A
Loop on element groups
Element group data
Update column heights
A contents here
Element group data
Write to auxiliary storage device eg. IEL
Solution phase
Determine addresses of diagonals in stiffness
find IDIAG
Equation system data
B

Flowchart (cont'd)




C
Factor stiffness
Analysis type
Rank check
Static analysis
Diagonals of factored stiffness
100
Calculate displacements
Transfer right-hand side to displacement array
Displacements in B
D



gives you phase information — compare efficiencies

if $\underset{\sim}{K}$ is + def. then eigenvalues of $\underset{\sim}{D}$ are >0
$\underset{\sim}{U}$ has same profile of $\underset{\sim}{K}$ $\Rightarrow$ storage of $\underset{\sim}{U}$: can use same as $\underset{\sim}{K}$
storage of $\underset{\sim}{D}$: need to provide this

Factoriz is done via Crout's method.
define intermediate steps

$\underset{\sim}{U}^T \underset{\sim}{x} = \underset{\sim}{F}$ (forward reduction) solve for $\underset{\sim}{x}$

$\underset{\sim}{D}\,\underset{\sim}{y} = \underset{\sim}{x}$ solve for $\underset{\sim}{y}$

$\underset{\sim}{U}\,\underset{\sim}{d} = \underset{\sim}{y}$ (back substitution)

see pg 10-11 of handout - talks about blank common.

suppose we want to pass X (NSD, NUMP) to COORD

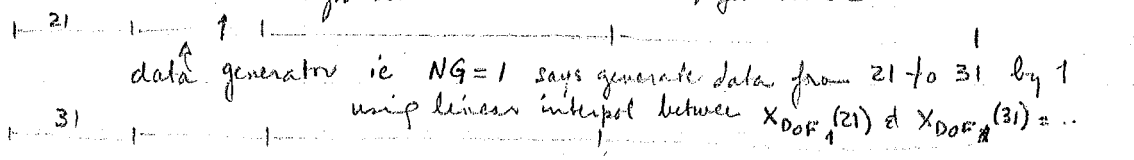
use CALL COORD (A(N2), NSD, NUMP) where X is defined in blank common array A(I) and X begins at A element N2.

∃ an element library for each element we pick
ie SUBROUTINE ELMLIB

```
      :
      GO TO (1,2,.... ), I   <-- must be passed or be in common.
    1 CALL TRUSS
      RETURN
    2 CALL PLANE
      RETURN
    3
      :
```

To input data
1st CARD TITLE card 80 characters like comment
control card (I5 for integer F10. float) This card uses I
all these are found in LEARN

data card



data generator ie NG=1 says generate data from 21 to 31 by 1
using linear interpol between $X_{DOF\,1}(21)$ & $X_{DOF\,1}(31)$ ...

4/14/81

LEARN books will be in THURS

Monday 4/20/81 evening will be a short class on computing on classwork at CIT.

Chapter 4. - Advanced Topics in F.E.

we will be dealing in problems w/ near or complete incompressibility some structural problems.

§ 0. Prelude - why do things work ? (Sometimes).

Look at heat conduction (or elasticity)

Recall weak form

(w) $$a(w,u) = (w,f) + (w,h)_\Gamma \qquad \forall w \in U \qquad (1)$$

(G) $$a(w^h,u^h) = (w^h,f) + (w^h,h)_\Gamma \qquad \forall w^h \in U^h \subset U \qquad (2)$$

we've done this "by the rules"

Now we will look at errors involved.

take (1) and let w^h replace w since $w^h \in U$

then

$$a(w^h,u) = (w^h,f) + (w^h,h)_\Gamma \qquad (3)$$

subtract (2) from (3)

$$a(w^h, u^h - u) = 0 \qquad \text{then } u^h - u = e \Rightarrow e \perp_a w^h \ \forall w^h \in U^h$$

Consider $a(e+w^h, e+w^h) = a(e,e) + 2a(w^h,e)^{\,0} + a(w^h,w^h) \quad \forall w^h \in U^h$

Recall that $a(w,w) > 0$ if $w \neq 0$ & $\forall w \in U$ (with right b.c.).

$$\Rightarrow \quad a(e,e) \leq a(e+w^h, e+w^h) \qquad \forall w^h \in U^h$$

$$a(e+w^h, \dots) = a(u^h - u + w^h, \dots) = a(U^h - u, U^h - u) \quad \text{where } u \in \mathcal{S}$$

where $\mathcal{S} = U + \text{b.c. w/ } U^h = u^h + w^h$;

since $u^h \in \mathcal{S}^h = U^h + \text{b.c.} \Rightarrow U^h \in \mathcal{S}^h$ since $w^h = 0$ as b.c.

now $a(e,e) \leq a(U^h - u, U^h - u) \ \forall U^h \in \mathcal{S}^h \Rightarrow U^h$ can be any fn.

hence any fn picked in $\mathcal{S}^h$ will have a measure $\geq$ the error

hence $a(e,e)$ is the best approximation. All problems looked at previously satisfy this.

$a(e,e) = \int_\Omega e_{,i}\, \kappa_{ij}\, e_{,j}\, d\Omega$. this is the measure used in heat conduction

$a(e,e) = \int_\Omega e_{(i,j)}\, c_{ijkl}\, e_{(k,l)}\, d\Omega$ " " " " " " elasticity

hences we are basically "least squares" best fitting the fn ie we are doing a minimization of the energy. However this result may be misleading

What can possibly go wrong? Approx S^h could be a bad approximation.

1. this can happen in incompressible or nearly incompressible probles

2. also the measure $a(\cdot,\cdot)$ might be inappropriate ie best approx wrt wrong measure. (ie incomp problems)

3. Combination of 1 and 2.

Note that $a(e,e) \le a(U^h - u, U^h - u)$ is not a statement of convergence must let $h \to 0$ to show convergence ie $S = \lim_{h\to 0} \{S^h\}$. $\Rightarrow u = \lim_{h\to 0} \{u^h\}$

Convergence (use best approx + $S^h \tilde{\subset} S$).

Hypotheses

(1) $\exists$ constants $c_1, c_2 > 0$ (not fns of h) $\quad c_1 \|v\|_m^2 \le a(v,v) \le c_2 \|v\|_m^2$ this is an equivalence between $\|v\|_m^2$ & $a(v,v)$

where $\|\cdot\|_m$ = m norm $\quad \|\cdot\|_0 = \int_\Omega (\cdot)^2 d\Omega$; $\|\cdot\|_1^2 = \int ((\cdot)^2 + [\nabla(\cdot)]^2)\, d\Omega$ (Sobolev norm).

(2) $S^h \tilde{\subset} S$ given any $\bar{U} \in S$, $\exists\, U^h \in S^h$ s.t. $\|U^h - \bar{U}\|_m \le c_3 h^\alpha \|\bar{U}\|_s$ where c_3 is a constant (not a fn of h); where $\alpha > 0$ & constant; as $h \to 0$. $s \ge m$

m = order of differentiability of a

for most of our work $m = 1$ (for bernoulli-euler $m = 2$).

Proof of convergence

$c_1 \|e\|_m^2 \le a(e,e)$ using (1)

by best approx $a(e,e) \le a(U^h - u, U^h - u)$ $\forall\, U^h \in S^h$

by (1) $\le c_2 \|U^h - u\|_m^2$

now pick $U^h \Rightarrow$ (2) holds.

$$\therefore \quad c_2 \|U^h - u\|_m^2 \leq c_2 c_3^2 h^{2\alpha} \|u\|_s^2$$

now from $c_1 \|e\|_m^2 \leq c_2 c_3^2 h^{2\alpha} \|u\|_s^2$

$$\|e\|_m \leq \left(\frac{c_2 c_3^2}{c_1}\right)^{1/2} h^\alpha \|u\|_s$$

now $\left(\frac{c_2 c_3^2}{c_1}\right)^{1/2}$ & $\|u\|_s$ are not fns of h & are fixed.

Thus as $h \to 0$ $\|e\|_m \sim h^\alpha \to 0$

if c_1 is a fn of h (c_1 is a measure of positive definiteness) as $h \to 0$ $c_1 \to 0$ i.e. lowest evalue goes thru 0; thus the matrix is rank deficient, or rigid body modes are not taken into account.

if c_2, c_3 are very large then $\Rightarrow h \ll 1$ for the effect of c_2, c_3 to be minimized; this type of effect occurs in structural modes.

Hence for $\|e\|_m \leq \left(\frac{c_2 c_3^2}{c_1}\right)^{1/2} \|u\|_s h^\alpha$ means something if c_2, c_3, c_1 are well behaved

... Prof Hughes will give us the results of all this for some of the elements we have looked at.

We now look at problems where things go wrong.

§ 1. Incompressible elasticity & stokes flow.

(ie rubber) Incompressibility = elasticity w/ modification $u_{i,i} = 0$
stiff bulk mod wrt shear mod.

Now $\sigma_{ij} = -p\delta_{ij} + C'_{ijkl}\varepsilon_{kl}$ (shear deviator). we will assume C'_{ijkl} have same properties as C_{ijkl} of last quarter.

p is unknown & must add one more equation ($\nabla \cdot u = 0$)

Now (S) $\sigma_{ij,j} + f_i = 0$ w/ $\nabla \cdot u = 0$ in Ω

and $g_i = u_i$ on Γ_{g_i}

$\sigma_{ij} n_j = h_i$ on Γ_{h_i} p appears in the body & governing

if we interpret u_i to be velocity, then eqns. are same as isotropic incompress elasticity
ie $c_{ijkl} = \mu(\delta_{ik}\delta_{jl} + \delta_{ij}\delta_{kl})$. where μ is dynamic viscosity.

then $\int_\Omega u_{i,i}\, d\Omega = \int_\Gamma u_i n_i d\Gamma = \int_\Gamma g_i n_i d\Gamma$ if $\Gamma_{g_i} = \Gamma$

since $u_{i,i} = 0 \Rightarrow \int_\Gamma g_i n_i d\Gamma = 0$

16 April 1981

Orientation lecture of CIT Computing Monday 7:30–9:30 PM 111 Polya
Jim Wilczak

Exples of Sobolev norms for our probls. ie $\|u\|_1^2 = \int_0^1 (v^2 + v_{,x}^2)\,dx$; $\|u\|_2^2 = \int_0^1 (u^2 + u_{,x}^2 + u_{,xx}^2)\,dx$

1. linear element 1-D ——— S^h (2 node elements)

$\|u - U^h\|_1 \le ch^1 \|u\|_2$
$\|u - U^h\|_0 \le ch^2 \|u\|_2$ if we do not make any restrictions of the derivs then we approximate u better. (

2. Quad elements ——•——• U^h is quadratic 3 node element.

$\|u - U^h\|_1 \le ch^2 \|u\|_3$
$\|u - U^h\|_0 \le ch^3 \|u\|_3$

All these results are independent of finite elements. The Sobolev norms tell you that $\exists$ a U^h that approximates u very well but doesn't tell you how to get it. The FE method is then the mechanism to find this U^h!

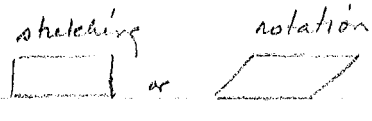
for higher node elements = n $\quad \|u - U^h\|_1 \le c h^{n-1} \|u\|_n$
$\|u - U^h\|_0 \le c h^n \|u\|_n$

3. for hermite cubics (displ & 1st deriv at nodes needed) ——•——• $a(\cdot,\cdot)$ involves 2nd derivs (ie beam problem)
then that is equiv. to $\|\ \|_2$

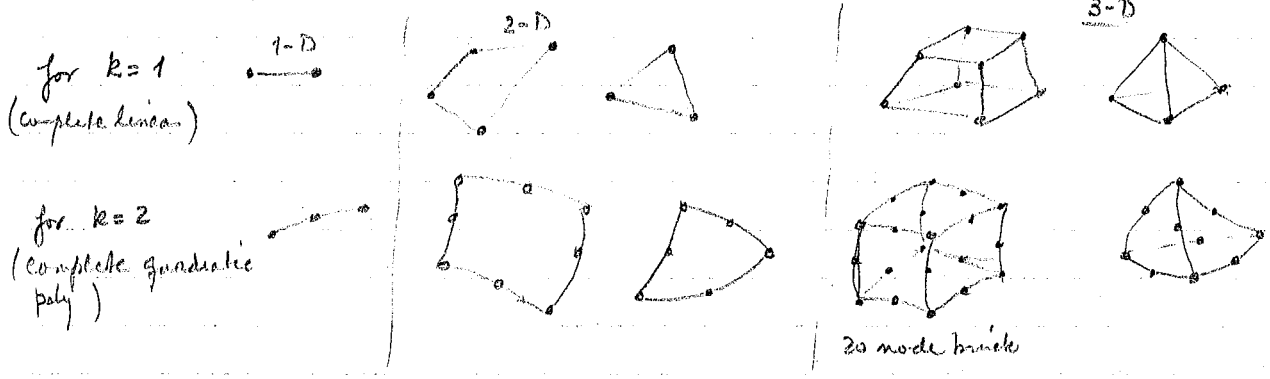
$\|u - U^h\|_2 \le ch^2 \|u\|_4$
$\|u - U^h\|_1 \le ch^3 \|u\|_4$
$\|u - U^h\|_0 \le ch^4 \|u\|_4$

4. Isoparametric elements (since we have slope discont. then m is 1 or 0 only
Always $\|u - U^h\|_1 \le ch^1 \|u\|_2$ parent domain satisfy completeness
$\|u - U^h\|_0 \le ch^2 \|u\|_2$ of linear polyn

if $x(\xi)$ is linear then $\|u - U^h\|_1 \le ch^k \|u\|_{k+1}$ where k is the degree of complete
$\|u - U^h\|_0 \le ch^{k+1} \|u\|_{k+1}$ } both

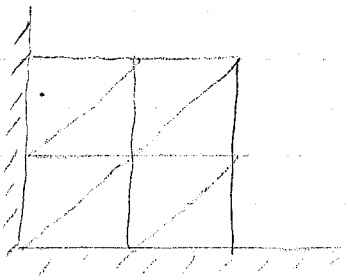
linear mapping ie 

if $\underline{x}(\underline{\xi})$ is a "perturbation" of a linear mapping then the above is still true
"perturb" $\Rightarrow$ not too distorted

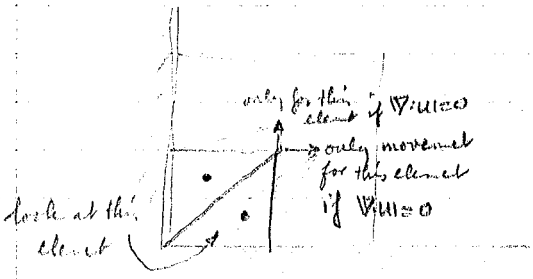


Now back to incompressibility and Stokes Flows.
From last time we had unknowns u_i, and new one p $(\frac{1}{3}\sigma_{ii})$
and the normal eqns + the new one $\nabla \cdot \underline{u} = 0$.

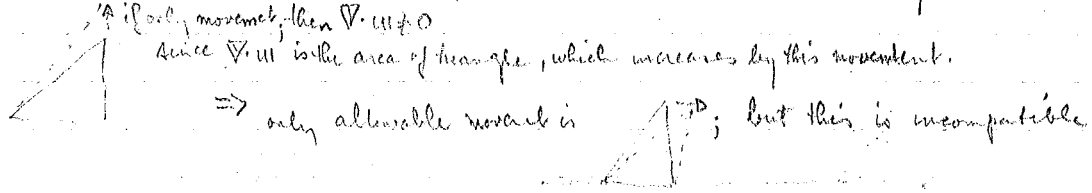
Suppose we have the following problem.



linear triangles.
and zero displ prescribed on bdry.



then this means the point is fixed $u = 0$
if we go to the next set - the same is true $\Rightarrow u = 0$
$\Rightarrow \forall$ nodes $u = 0$
the phenomena is known as locking



or/adjacent element $\Rightarrow u = 0$ at node.

Suppose we let $\hat{u_i} \approx u$, $p^h \approx p$ and we put this into our variational principle we get

$$a'(w,u)\left(=\int_\Omega w_{(i,j)}\, C_{ijkl}\, u_{(k,l)}\, d\Omega\right) - (\nabla\cdot w, p) = \text{usual RHS}$$

for the $\nabla\cdot u = 0$ eq. the variational principle leads to $(q, \nabla\cdot u) = 0$ where q is the weighting fn corresponding to p.

also $(\nabla\cdot w, p) = \int_\Omega w_{i,i}\, p\, d\Omega$

in Galerkin method ‖ replace w by w^h; u by u^h; p by p^h; q by q^h where $q^h, p^h \in P^h$ (space of fns).

Now after the matrix construction we get

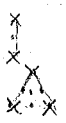
$$\begin{bmatrix} \underset{\sim}{K}_1 & \underset{\sim}{K}_2 \\ \underset{\sim}{K}_2^T & \underset{\sim}{0} \end{bmatrix} \begin{bmatrix} \underset{\sim}{d} \\ \underset{\sim}{p} \end{bmatrix} = \begin{bmatrix} \underset{\sim}{F} \\ \underset{\sim}{0} \end{bmatrix}$$

$\underset{\sim}{F}$ is the RHS.

where $\underset{\sim}{K}_1$ corresponds to C_{ijkl} and $\underset{\sim}{K}_2^T \underset{\sim}{d} = \underset{\sim}{0}$ represents $\nabla\cdot u = 0$
now we must ask what is P^h? that is the problem.

1. Generally we get disastrous results if $P^h = \Delta^h$ (ie same type interpol as Δ^h).
2. this has led to P^h having lower order than Δ^h ie continuous fns.

X - press node
· - u node



example: 9 node elements w/ bilinear pressures (biquadratic displacements)
7 noded triangle w/ linear pressures.

3. Since $\nabla\cdot u$ is discont. from elmt to elmt we can take p discontinuous also

example 9 node displ + constant p.  u biquad u ·, p const. p ×

P - linear 9 node displ  u quadratic, p linear this is better for optimality
$p = \alpha_0 + \alpha_1 x + \alpha_2 y$ & discont. from elements

optimality in sense $\exists$ $2u_i$ eqns + $1p$ eqn $\therefore$ ratio 2/1 in 2-D

if we pick constant p and bilinear u  but has pressure oscill w/ some problems ie "Checkerboard pressure"

PENALTY FUNCTION FORMULATION

4/21/81

This will provide an approximation for incompressible elasticity / Stokes flow which can be used in practice

(1) Replace $p = -\frac{1}{\varepsilon} \operatorname{div} u_i^{(\varepsilon)}$ (ε is a very small parameter)
$= -\frac{1}{\varepsilon} \nabla \cdot u$

(2) Drop $u_{i,i} = 0$

This leads to the following

Problem: given f_i, g_i, h_i, etc,

$(S^{(\varepsilon)})$: $\sigma_{ij,j}^{(\varepsilon)} + f_i = 0$

$u_i^{(\varepsilon)} = g_i$ on Γ_{g_i}

$\sigma_{ij}^{(\varepsilon)} n_j = h_i$ on Γ_{h_i}

$\sigma_{ij}^{(\varepsilon)} = -\frac{1}{\varepsilon} u_{k,k}^{(\varepsilon)} \delta_{ij} + C'_{ijkl} u_{(k,l)}^{(\varepsilon)} = p\delta_{ij} + C'_{ijkl}\varepsilon_{kl}$

$= C_{ijkl} u_{(k,l)}$

where $C_{ijkl} = C'_{ijkl} + \frac{1}{\varepsilon}\delta_{ij}\delta_{kl}$

Note as $\varepsilon \to 0$, $(H^1) u_i^{(\varepsilon)} \to u_i$ = soln. of incomp. problem

$(L_2) p^{(\varepsilon)} \to p$ = soln. of incomp. problem

We have changed the incompressible problem to a slightly compressible problem

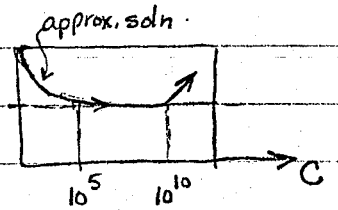
If we assume the isotropic case, note that $\frac{1}{\varepsilon}$ corresponds to λ. λ acts like a bulk modulus in the incompressible limit

Rule of thumb for size of ε:

$\frac{1}{\varepsilon} = \lambda = C_{max} \underbrace{C'_{ijkl}}_{2\mu}$

where C is a constant

so problem is relatively insensitive to C in the range $10^5 - 10^{10}$



"1" = actual soln.

Double prec. IBM - 64 bit
Single prec. CDC - 60 bit
} computer info - why CDC is better

The definitions of λ & μ are

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)} \qquad \text{As } \nu \to \tfrac{1}{2},\ \lambda \to \infty$$

$$\mu = \frac{E}{2(1+\nu)}$$

For Plane stress – there is no incompressible problem

$$\bar{E} = E/(1-\nu^2)$$
$$\bar{\nu} = \nu/(1-\nu)$$
> As $\nu \to \tfrac{1}{2}$, $\bar{E}$ & $\bar{\nu}$ are finite

For Plane strain – problem still blows up

Recall the stiffness calculation

$$\underset{\sim}{k}^e_{ab} = \int_{\Omega^e} \underset{\sim}{B}_a^T \underset{\sim}{D} \underset{\sim}{B}_b \, d\Omega \qquad \underset{\sim}{D} = \underset{\sim}{D}_\lambda + \underset{\sim}{D}_\mu$$

$\underset{\sim}{D}_\lambda$ is locking up the Kinematic in the incompressible mode

$$= \int_{\Omega^e} \underset{\sim}{B}_a^T \underset{\sim}{D}_\lambda \underset{\sim}{B}_b \, d\Omega + \int_{\Omega^e} \underset{\sim}{B}_a^T \underset{\sim}{D}_\mu \underset{\sim}{B}_b \, d\Omega$$

must do something special here to avoid disaster — no problem encountered by this calculation

For $n_{s.d.} = 2$ (plane strain):

$$\underset{\sim}{D}_\lambda = \lambda \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\underset{\sim}{D}_\mu = \mu \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For $n_{s.d.} = 3$:

$$\underset{\sim}{D}_\lambda = \left[\begin{array}{ccc|c} 1 & 1 & 1 & \\ 1 & 1 & 1 & \mathbf{0} \\ 1 & 1 & 1 & \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right]$$

$$\underset{\sim}{D}_\mu = \begin{bmatrix} 2 & & & & & \\ & 2 & & & \mathbf{0} & \\ & & 2 & & & \\ & & & 1 & & \\ & \mathbf{0} & & & 1 & \\ & & & & & 1 \end{bmatrix}$$

In the non-isotropic case,

$$(D_\mu)_{IJ} = C'_{ijkl}$$

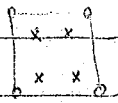
One possible way to develop an improved formulation:

REDUCED/SELECTIVE INTEGRATION

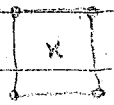
- use different rules on different terms

$$\underset{\sim}{k}^e_{ab} \cong \sum_{l^{(1)}=1}^{n^{(1)}_{int}} \left(\underset{\sim}{B}_a^T \underset{\sim}{D} \underset{\sim}{B}_b j\right)_{l^{(1)}} W^{(1)}_{l^{(1)}} + \sum_{l^{(2)}=1}^{n^{(2)}_{int}} \left(\underset{\sim}{B}_a^T \underset{\sim}{D}_\mu \underset{\sim}{B}_b j\right)_{l^{(2)}} W^{(2)}_{l^{(2)}}$$

- rule 1 is lower than rule 2
- see PLANE 1, pp. 67-68 of LEARN manual



There is a normal rule that goes with an element, i.e. 2x2 for 4-node bilinear el.

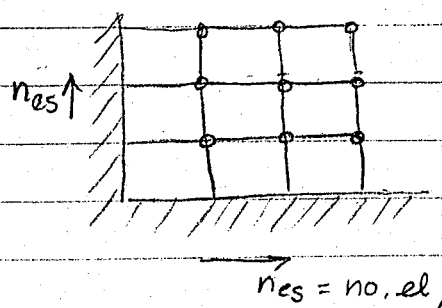


A reduced rule means a lower order, i.e. 1 pt. rule for 4-node quad.

Uniform reduced integration involves a 1-pt. rule on λ & μ terms (in 4-node example)

Selective reduced integration would involve a 1-pt. rule on λ and 2x2 on μ

Constraint counting:



n_{eq} = no. equations

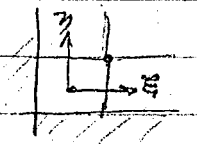
n_c = no. constraints imposed by incompressibility

$$\text{Constraint index} = \frac{n_{eq} - n_c}{n_{es}^2} \quad \text{(CI)}$$

$$\text{Constraint ratio} = \frac{n_{eq}}{n_c} \quad \text{(CR)}$$

All successful elements have constraint ratio > 2 (= 2 for 2-D continuum)

Midterm Tues May 1% in class



$n_{eq}/n_{el} = 2$ $\quad x = x(\xi,\eta) = x(\xi)$

$y = y(\xi,\eta) = y(\eta)$

eg. $\frac{1}{2}(1-\eta)y_1 + \frac{1}{2}(1+\eta)y_2$ ← a linear function physically, just from stretching

$u_i^h = u_i^h(\xi,\eta)$ bilin.

$= u_i^h(x,y)$ also bilin.

Take $0 \cong u^h_{i,i}$ $\quad \alpha_{ji}$ = constants = linear combos. of nodal values d_{ic}

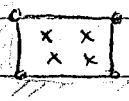
$u_i^h = \alpha_{0i} + \alpha_{1i}x + \alpha_{2i}y + \alpha_{3i}xy$

$u^h_{i,i} = u^h_{1,1} + u^h_{2,2} = \alpha_{11} + \alpha_{22} + \alpha_{31}y + \alpha_{32}x \cong 0$

$= 0 \Rightarrow \alpha_{11}+\alpha_{22}=0$, $\alpha_{31}=0$, $\alpha_{32}=0$ } 3 constraints on nodal values at each point

Thus $n_c/n_{el} = 3$

4/23/81



CI = −1 {2 eq/el, 3 constr.

opt. "CR = 2" 4 Nod/sided

2×2 ⇒ bilin u_i, linear $u_{i,i}$ ⇒ 3 const.

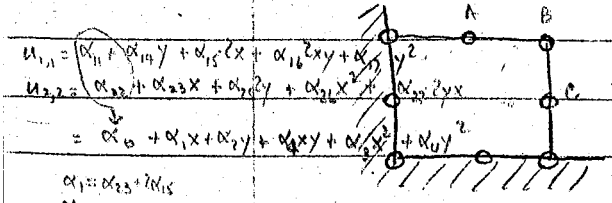
CI = 2 − 1 = +1

CR = 2/1 = 2

(λ-term 1pt ⇒ $u^h_{i,i}$ = const ⇒ 1 const.

EXAMPLES: (WE ALWAYS LOOK AT THE NODE IN THE CORNER SINCE THE OTHERS WILL BE THE SAME)

(1) 8-node - standard quadrature is 3×3



$u_{1,1} = (\alpha_{11} + \alpha_{14}y + \alpha_{15}2x + \alpha_{16}2xy + \alpha_{17}y^2 \ldots$

$u_{2,2} = (\alpha_{22} + \alpha_{23}x + \alpha_{24}2y + \alpha_{26}x^2 + \alpha_{27}2yx$

$= \alpha_0 + \alpha_1 x + \alpha_2 y + \alpha_4 xy + \alpha_5 x^2 + \alpha_6 y^2$

$\alpha_1 = \alpha_{23} + 2\alpha_{15}$

$\alpha_4 = \alpha_{14} + 2\alpha_{27}$

$\alpha_5 = 2\alpha_{27} + 2\alpha_{16}$

$\alpha_3 = \alpha_{24}$

$\alpha_0 = \alpha_{17}$

6 eqns. (2 at each A/B/C). This is a serendipity element

terms involved are $1, x, y, xy, x^2, y^2, x^2y, y^2x$

$0 \cong u^h_{i,i} = \alpha_0 + \alpha_1 x + \alpha_2 y + \alpha_3 x^2 + \alpha_4 xy + \alpha_5 y^2$

we need to require all $\alpha_i = 0$ for $u^h_{i,i} = 0$

∴ 6 constraint eqns. $\quad$ C.I. = 6 − 6 = 0

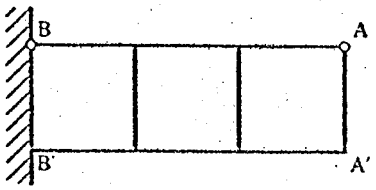
CR = 6/6 = 1

Now try reduced/selective integration. Use 2×2 on λ-term

$n_c = 4$ (at most) since 2×2 is bilinear and has 1 unknown at each of the quad pts

CI = 4 − 2 = 2 $\quad$ CR = 6/4 = 1.5 ← better but not up to 2 yet

Probably the most serious economic problem of complex curvilinear elements is the computer time necessary for performing the numerical integrations. Here some economic limit on the accuracy required in this integration must obviously be imposed.



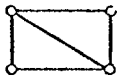
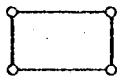
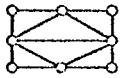
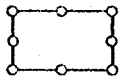
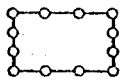
Type of element	Vertical Load of A		Couple at AA'	
	Max. defl. at AA'	Max. stress BB'	Max. defl. at AA'	Max. stress BB'
	0·26	0·19	0·22	0·22
	0·65	0·56	0·67	0·67
	0·53	0·51	0·52	0·55
	0·99	0·99	1·00	1·00
	1·00	1·00	1·00	1·00
EXACT	1·00	1·00	1·00	1·00

Fig. 9.1 A cantilever in plane stress analysed by various elements. Accuracy improvement with higher order elements

9.2 Required Accuracy of Numerical Integration

In the previous chapter it has been indicated how the element matrices can be formed using numerical integration in terms of n Gauss points. The effort of this integration over plane area is roughly proportional to n^2—the number of points at which the function has to be found—while in a three-dimensional situation it is proportional to n^3. The determination of an *adequate minimum number* of Gauss points is thus of some importance.

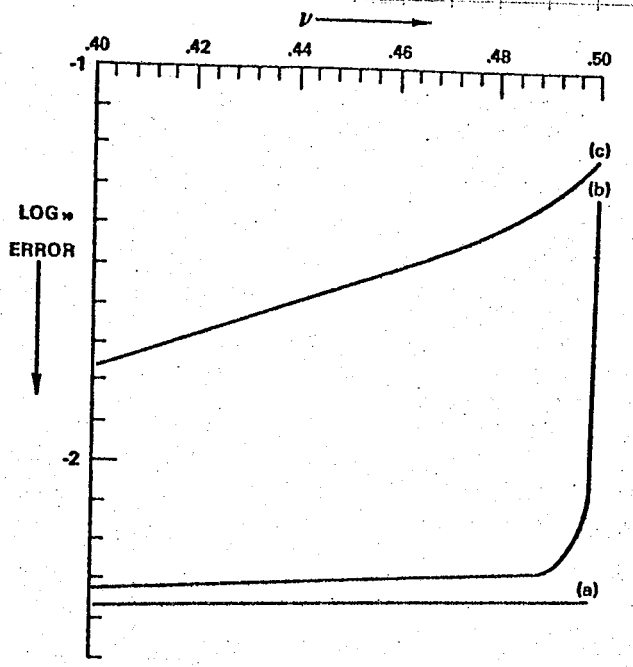
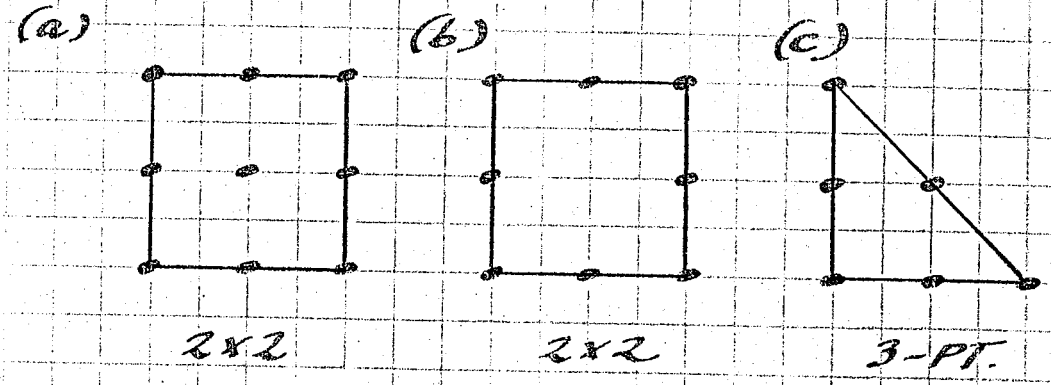


Fig. 3. $\|u^h - u\|$—error in approximating the exact solution with the artificial ν as this $\nu \to \frac{1}{2}$, for three types of quadratic elements, integrated with the formulas of Table 1: (a) Lagrange, (b) serendipity, (c) triangle.



From D.S. Malkus, 'A Finite Element Displacement Model Valid for any value of the Compressibility', Int. J. Solids Structures, Vol. 12, 731-738 (1976).

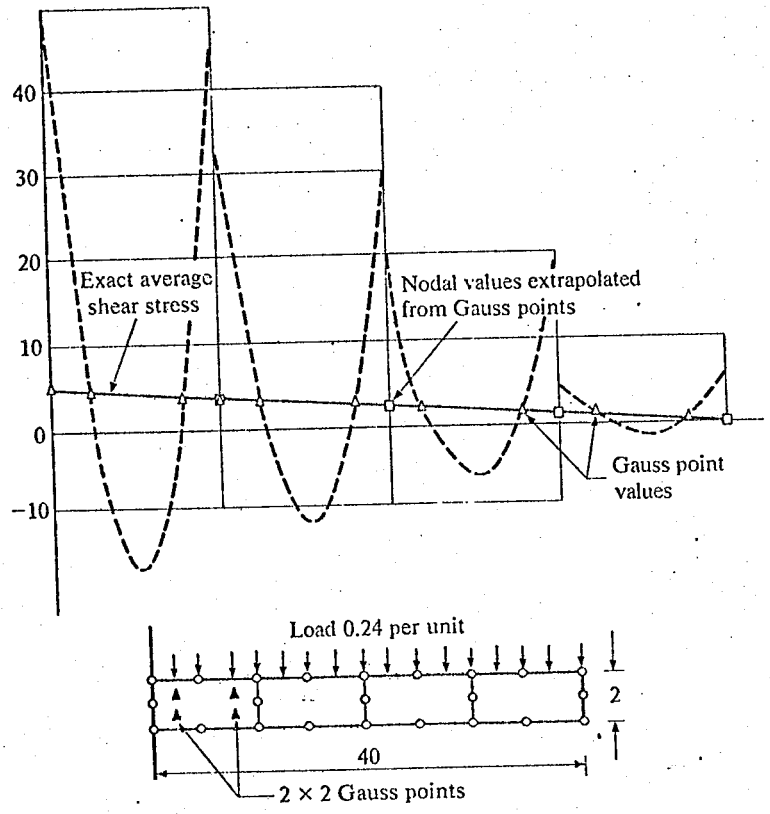
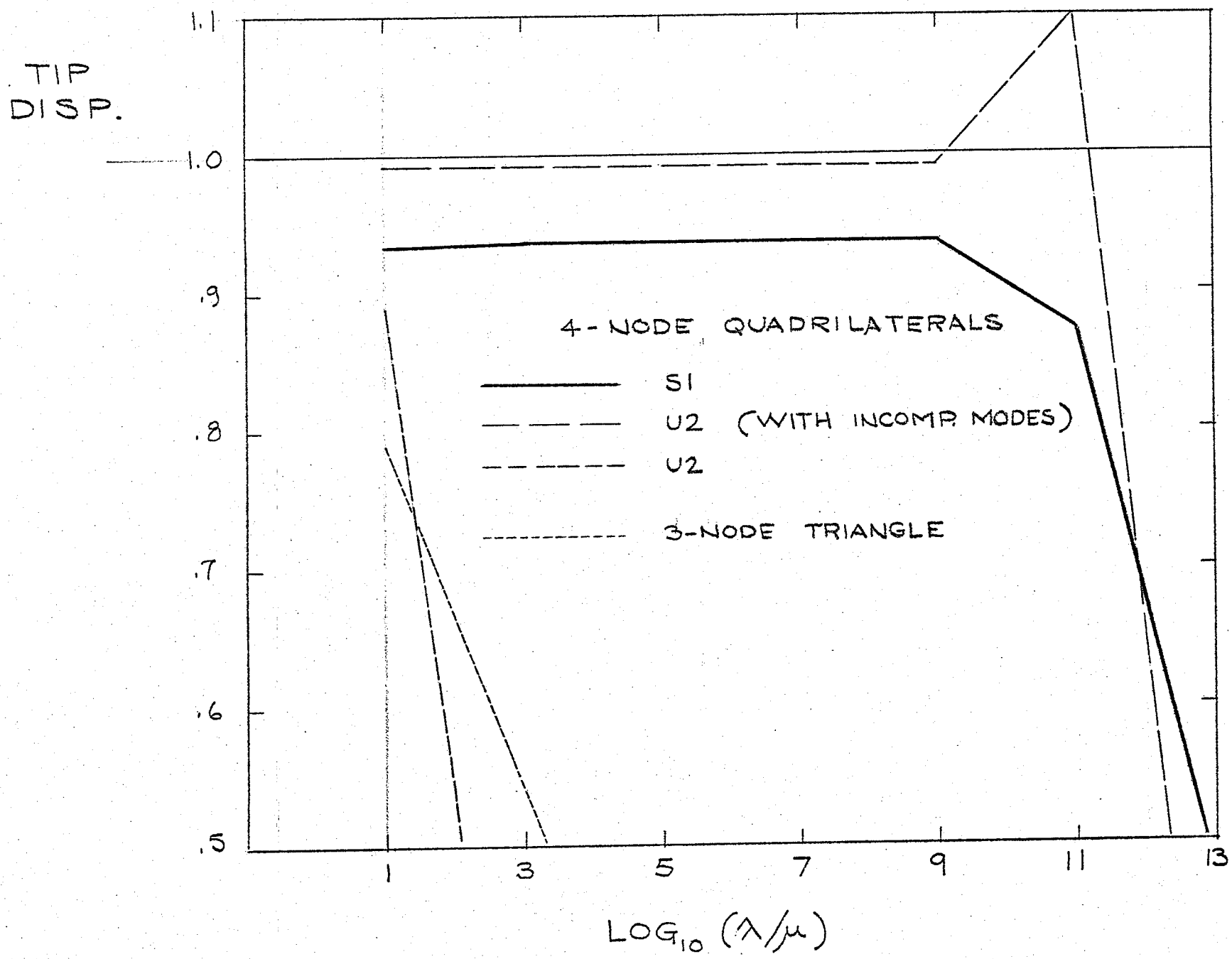


Fig. 11.9 A cantilever beam with four parabolic elements. Stress sampling at Gauss points and linear extrapolation to nodes

In Fig. 11.9 we show, for instance an analysis of a cantilever beam using four quadratic, 'serendipity' type, elements. Whilst the results for deflections and axial stresses are excellent, the shear stresses show a parabolic 'variation' in each element which provides an extremely poor representation of the actual stresses. However, the values sampled at the nodes are an excellent representation of the correct mean shear stresses.

Similar improvements can be shown in the context of other elements and problems, although (fortunately) the discrepancies are not always so large.

The example just quoted suggests that in quadratic C_0 elements, whether two- or three-dimensional, the stresses (or similar quantities) *should never be calculated at nodes*. If nodal values are desired, then a simple bilinear extrapolation from Gauss points should be made. Such values are again shown to be excellent in Fig. 11.9. Further examples of


TIP
DISP.
1.1
1.0
.9
.8
.7
.6
.5
1
3
5
7
9
11
13
4-NODE QUADRILATERALS
S1
U2 (WITH INCOMP. MODES)
U2
3-NODE TRIANGLE
LOG10 (λ/μ)

32 ELEMENTS Q4

λ	EXACT	2X2 INCOMP		2X2	2X2	2X2	1X1
		DISP	DISP/EXACT	DISP	DISP/EXACT	DISP	DISP/EXACT
10	74.27	73.77	0.993	66.02	0.889	69.13	0.931
10^3	68.56	68.03	0.992	15.34	0.224	64.21	0.937
10^5	68.50	67.96	0.992	4.14	0.06	64.16	0.937
10^7	68.50	67.96	0.992	4.00	0.058	64.16	0.937
10^9	68.50	67.94	0.992	4.00	0.058	64.14	0.937
10^{11}	68.50	75.22	1.098	5.09	0.074	60.01	0.876
10^{13}	68.50	10.87	0.158	17.09	0.25	5.95	0.087
10^{15}	68.50	0.57	0.008	0.39	0.006	0.787	0.011

λ	E	ν
10	2.909090909090909090	0.454545454545454545
10^3	2.999000999000999000	0.499500499500499500
10^5	2.999999000009999999	0.499999500000499999
10^7	2.999999990000000 1	0.499999995000000 5
10^9	2.999999999000000	0.499999999500000
10^{11}	2.999999999999000	0.499999999995000
10^{13}	2.999999999999999 00	0.499999999999995 0
10^{15}	2.999999999999999990	0.499999999999999950

64 ELEMENTS TRIANGLES

λ	EXACT	2X2 INCOMP		2X2	2X2	2X2	1X1
		DISP	DISP/EXACT	DISP	DISP/EXACT	DISP	DISP/EXACT
10	74.27	58.555	0.788	58.555	0.788	58.555	0.788
10^3	68.56	36.905	0.538	36.905	0.538	36.905	0.538

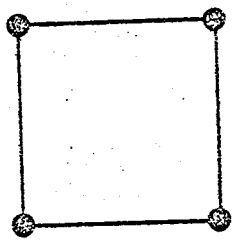
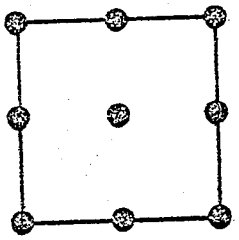
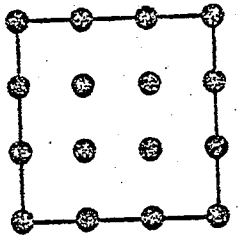
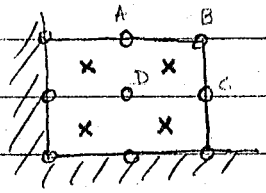
			
shape functions	bilinear	biquadratic	bicubic
λ-term	1 point	2×2	3×3
μ-term	2×2	3×3	4×4

Figure 3. Selective Gauss-Legendre integration rules for 2-dimensional isoparametric Lagrange elements.

(2) 9-node element ("everybody's darling")



8 eqns. (2 at each A, B, C, D)

8 fns. from serendipity element +

x^2y^2 from bubble

$0 \cong u^h_{i,i} = \underbrace{\text{QUADRATIC}}_{\text{6 param's}} + \alpha_6 xy^2 + \alpha_7 y^2 x$

$CI = 8 - 8 = 0$

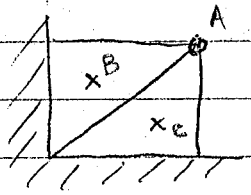
$CR = 8/8 = 1$

RED/SEL INT: 2×2 on λ $\quad n_c = 4$ linear in p ie $u_{i,i}$ involves $1, x, y, xy$ fns

$CI = 8 - 4 = 4$

$CR = \frac{8}{4} = 2$

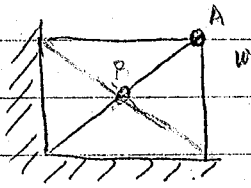
(3) TRIANGLE (LINEAR) 4 (2 from each triangle) less 2 (constraint from pt p same) = 2



why (a) $n_{eq} = 2$ (2 each at A) $\quad CI = 2 - 2 = 0$

$n_c = 2$ (1 each at B, C) $\quad CR = 1$? $= 2/2$ wrt rectangular

$u^h_{i,i}$ is constant in each element - "lock-up"



why (b) $n_{eq} = 4$ (2 each at A, B)

$n_c = 3$ { because of linear dependence of the original 4

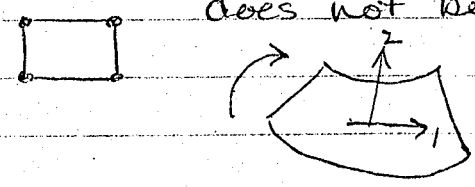
$CI = 1$

$CR = \frac{4}{3}$

this "more-or-less" works

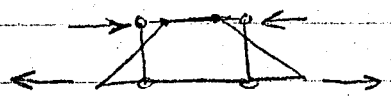
(4) INCOMPATIBLE MODES

(a) does not bend well - usual isoparametric element



$u_1 = cxy$

$u_2 = \frac{c_1}{2}(a^2 - x^2) + c_2(b^2 - y^2)$? is this the actual solution



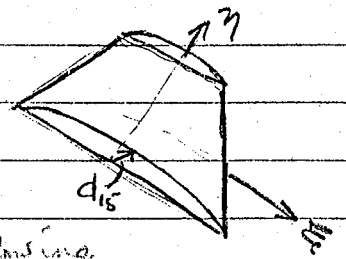
bilinear element response

Wilson et. al. (Berkeley) have suggested that you add the ξ^2 and η^2 deformation patterns

$$u_i^h = \sum_{a=1}^{4} N_a(\xi,\eta)\, d_{ia}^e + \sum_{a=5}^{6} N_a \hat{d}_{ia}^e \qquad N_5 = 1-\xi^2 \quad N_6 = 1-\eta^2$$

$\hat{d}$ ← not nodal parameters

$$u_1^h = \hat{d}_{15}(1-\xi^2)$$



This element is not continuous (that's what incompatible means)

this changes the following

$$\underset{\sim}{k}^e = \int_{\Omega^e} \underset{\sim}{B}^T \underset{\sim}{D}\, \underset{\sim}{B}\, d\Omega \quad \text{where} \quad \underset{\sim}{B} = [\underset{\sim}{B}_1 .. \underset{\sim}{B}_4 \,|\, \underset{\sim}{B}_5 \; \underset{\sim}{B}_6]$$

$\underset{\sim}{k}^e$: 12×12 ; $\underset{\sim}{B}_1$: 3×2

$$\underset{\sim}{k}^e = \left[\begin{array}{c|c} \underset{\sim}{k} & \hat{\underset{\sim}{k}} \\ \hline \hat{\underset{\sim}{k}}^T & \hat{\hat{\underset{\sim}{k}}} \end{array}\right] \quad \begin{matrix} 8\times8 & 8\times4 \\ 4\times8 & 4\times4 \end{matrix}$$

$\hat{\hat{k}}$ is whatever is in lower box

Assume that the generalized forces corresponding to the incompressible modes ($\hat{d}_{i5}, \hat{d}_{i6}$) are zero

$$\underset{\sim}{\hat{k}}^T \underset{\sim}{d}^e + \hat{\hat{\underset{\sim}{k}}}\, \hat{\underset{\sim}{d}} = 0$$

4×8 · 8×1 · 4×4 · 4×1 = 4×1

Solve the eqn. for $\hat{d}$:

$$\hat{\underset{\sim}{d}} = -\hat{\hat{\underset{\sim}{k}}}^{-1}\, \hat{\underset{\sim}{k}}^T \underset{\sim}{d}^e$$

Substitute in: $\underset{\sim}{k}\, \underset{\sim}{d}^e + \hat{\underset{\sim}{k}}\, \hat{\underset{\sim}{d}} = \underset{\sim}{f}^e$

$$\underbrace{\left(\underset{\sim}{k} - \hat{\underset{\sim}{k}}\, \hat{\hat{\underset{\sim}{k}}}^{-1} \hat{\underset{\sim}{k}}^T\right)}_{\text{stiffness with incompressible modes}} \underset{\sim}{d}^e = \underset{\sim}{f}^e$$

This proc. is called STATCON (static condensation) and is included in LEARN

1. Coding - add N_5, N_6 to shape fn. routine
2. Increase nodal loops from 4 to 6
3. Static condensation routine to do the above.

??!

$\sigma_{ij,j} + f_i = 0 \quad \Rightarrow \quad \sigma_{ii,i} + \sigma_{ij,j} + f_i = 0$

$\sigma_{ii,i} = C_{iiii}\, u_{i,ii}$

$\int w_i(\sigma_{ij,j})\, d\Omega \qquad \int w_i f_i\, d\Omega = 0$

$\nabla\cdot(w\sigma) - \sigma\cdot\nabla w$

$\int w_i(\sigma_{ii,i}) + \int w_i(\sigma_{ij,j})$

$\int[(\sigma_{ii} w_i)_{,i} - \sigma_{ii} w_{i,i}]d\Omega \int w_i(\sigma_{ij,j})\, d\Omega$

$\int \sigma_{ii} w_i n_i\, d\Gamma - \int \sigma_{ii}\nabla\cdot w$

$\int w_i \sigma_{ij,j} + \int w_i f_i\, d\Omega = 0$

$\int (w_i\sigma_{ij})_{,j} - \int (w_{i,j}\sigma_{ij}) + \int w_i f_i d\Omega = 0 \qquad$ using $-p\delta_{ij} + C_{ijkl}\varepsilon_{kl} = \sigma_{ij}$

$\cdots + \int w_{i,j} p\,\delta_{ij} d\Omega - \int w_{i,j} C_{ijkl}\, u_{(k,l)}\, d\Omega + \cdots = 0$

$\int (w_i \sigma_{ij} n_j)\, d\Gamma + \int w_{i,i} p\, d\Omega - \int w_{(i,j)} C_{ijkl}\, u_{(k,l)}\, d\Omega + \cdots = 0$

$\Rightarrow$

$\int_\Omega w_{(i,j)} C_{ijkl}\, u_{(k,l)} d\Omega - \int_\Omega \nabla\cdot w\, p\, d\Omega = \int_\Omega w_i f_i d\Omega + \int_{\Gamma_{h_i}} w_i h_i d\Gamma \qquad h_i = \sigma_{ij} n_j \text{ on } \Gamma_{h_i}$

if $\nabla\cdot u = 0 \Rightarrow \int_\Omega q u_{i,i}\, d\Omega = 0 \Rightarrow \int[(q u_i)_{,i} - (q_{,i} u_i)]d\Omega = 0 \Rightarrow (q, \nabla\cdot u) =$

$\Rightarrow \int q u_i n_i\, d\Gamma = \int u\cdot\nabla q\, d\Omega$

$\int_{\Gamma_{g_i}} q\, g_i n_i\, d\Gamma = \int u\cdot\nabla q\, d\Omega$

Note that $\int_\Omega w_{(i,j)} C_{ijkl}\, u_{k,l} d\Omega = \int \underline{B}_a\, \underline{D}_\mu\, \underline{B}_b\, d\Omega$

$-\int_\Omega \nabla\cdot w \left(-\frac{1}{\varepsilon}\nabla\cdot u\right) d\Omega = \int \underset{\sim}{B}_a \underset{\sim}{D}_\lambda \underset{\sim}{B}_b) d\Omega \qquad$ where $\frac{1}{\varepsilon} = \lambda$

$\nabla\cdot w = \begin{pmatrix} N_{a,x} & 0 \\ 0 & N_{a,y} \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \delta_{11} \\ \delta_{12} \end{pmatrix}$ since $\nabla\cdot w = w_{1,1} + w_{1,2}$

1. P.L. Wilson, R.L. Taylor, W.P. Doherty and J. Ghaboussi, "Incompatible Displacement Models" pp. 43-57 in Numerical and Computer Methods in Structural Mechanics, Editors S. J. Fenves, N. Perrone, A.R. Robinson and W.C. Schnobrich, Academic Press, 1973.

2. P. Lancaster, "On the Convergence of Wilson's Non-conforming Element for Solving the Elastic Problem" Computer Methods in Applied Mechanics and Engineering, Vol. 7, 1-16 (1976).

3. R.L. Taylor, P.J. Beresford and E.L. Wilson, "A Non-conforming Element for Stress Analysis," International Journal for Numerical Methods in Engineering, Vol. 10, No. 6, 1211-1219 (1976).

4. D. Kosloff and G.A. Frazier, "Treatment of Hourglass Patterns in Low Order Finite Element Codes," Numerical and Analytical Methods in Geomechanics, Vol. 2, 57-72 (1978).

see handout of next lesson

4/28/81 Comments on incompatible modes

1) The element fns. are no longer continuous

what does this mean → 2) Elements converge for linear transformations

3) Taylor's method - change evaluation of shape fns. only for incompatible modes

$$\underset{\sim}{B_a} = \begin{bmatrix} N_{a,1} & 0 \\ 0 & N_{a,2} \\ N_{a,2} & N_{a,1} \end{bmatrix}$$

$$\underset{\sim}{B_5} = \frac{J^\circ}{2} \begin{bmatrix} -\xi(y,\eta) & 0 \\ 0 & \xi(x,\eta) \\ \xi(x,\eta) & -\xi(y,\eta) \end{bmatrix}$$

$$B_6 = \frac{J^\circ}{2} \begin{bmatrix} -\eta(y,\xi) & 0 \\ 0 & -\eta(x,\xi) \\ -\eta(x,\xi) & \eta(y,\xi) \end{bmatrix}$$

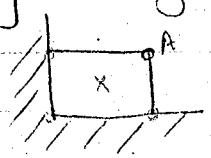
normal method

Taylor - substitute in ○ their values at $\xi = \eta = 0$

This is contained in LEARN in SHAP2ϕ

Remarks

1. This method works well for bending
2. It is a bit experimental
3. It can be generalized to 3-D with $1-\xi^2$, $1-\eta^2$, $1-\zeta^2$
4. The axisymmetric case does not converge
5. We get a good constraint index



$2+4 = 6$ eqns/el. how I can get 2 (from N) where 4 do they come from? from $\hat{d}$ in taylor formula (one pt quad eq)

trial $w^h = 1, x, y, x^2, xy, y^2$

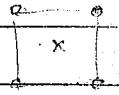
$0 = u_{,x} \Rightarrow$ 3 constraints

C.I. $= 6-3 = 3$

C.R. $= 6/3 = 2$

HOURGLASS STIFFNESS METHOD

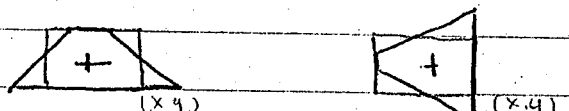
We observe some facts about 1-pt. quadrature on a bilinear element:



1. Optimally fast
2. Gives reasonably good C.I. (=1)
3. The element is singular
   2 - translational modes
   1 - rotational mode
   ∃ 3 energy modes

1-pt. quad. adds 2 spurious zero-energy mode

These modes look like:



$u_1 = c_1 \xi\eta$ $\qquad$ $u_2 = c_2 \xi\eta$

$0 = \underset{\sim}{d}^T \underset{\sim}{k}^e \underset{\sim}{d}$

$= \int \cdots$

$$\left.\begin{aligned} u_{1,x} &= c_1 \eta \\ u_{2,y} &= c_2 \xi \end{aligned}\right\} u$$

Kosloff adds to the one-pt. stiffness matrix a stiffness associated with the hourglass modes

Remarks

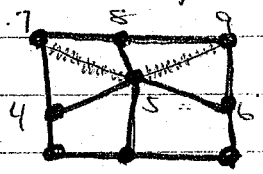
1. Element is exactly the same as incompressible modes in rectangular configurations
2. Method appears to behave better in arbitrary configurations
3. Method is much cheaper because it does not involve static condensation
4. Error in original Kosloff paper ③ → ⑫ corrected in later paper by Taylor
5. Method is coded in subroutine PLANE1, lines 165-180 FORMS1, FORMS2

ENGINEERING CRITERIA - how good are the results?

1. Invariance - results must be independent of reference configuration
   - this is usually satisfied (we will not go into when it is not)
2. Patch test - implies that the element is complete
   - satisfied by most standard integration rules

a) Engr. version - (Irons 1965)
   done on computer
   implies completeness
   if element can represent deformation states 1, x, y in an arbitrary patch of elements, then it passes

test element →



(trans)

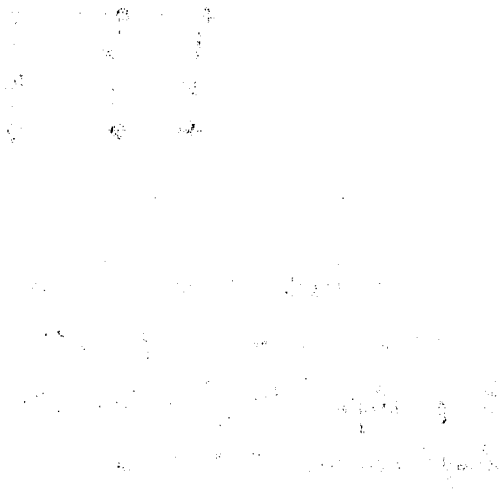
Set boundary nodes

	Test #	d_{1a}	d_{2a}	
1	1	1	0	a: 1→4, 6→9
	2	0	1	
x	3	x_a	0	
	4	0	x_a	
y	5	y_a	0	
	6	0	y_a	

set these values for the boundary nodes & get the analysis to give you the displacement of node 5.

A higher order patch test implies higher order completeness

(also a mathematical version due to Strang)



LINEAR TRANSIENT ANALYSIS

Look at the parabolic case - heat eqn.

(S)

Given: $f: \Omega \times]0,T[\to \mathbb{R}$ $\quad f:(\cdot,\cdot)$

$g: \Gamma_g \times]0,T[\to \mathbb{R}$

$u_0: \Omega \to \mathbb{R}$

Find: $u: \bar{\Omega} \times [0,T] \to \mathbb{R}$

$$\rho c u_{,t} + q_{i,i} = f \quad \text{on } \Omega \times]0,T[$$

$$u = g \quad \text{on } \Gamma_g \times]0,T[$$

$$q_i n_i = -\bar{h} \quad \text{on } \Gamma_h \times]0,T[$$

$$u(x,0) = u_0 \quad x \in \Omega$$

where $q_i = -K_{ij} u_{,j}$ $\quad$ if $K_{ij} = -\kappa \delta_{ij} \Rightarrow q_i = -\kappa u_{,i}$

This converts to the following

(W)

Given: $f, g, \bar{h}, u_0$ as in (S)

Find: $u[0,T] \to S_{g(t)}$ $\quad \forall w \in V$

$$(\rho c \dot{u}, w) + a(u, w) = (f, w) + (\bar{h}, w)_{\Gamma} \qquad (*)$$

$$(u(0) - u_0, w) = 0 \qquad (**)$$

see next page

Exercise: verify the equivalence of (S) & (W)

Note: (*) + usual hypothesis $\Rightarrow$ P.D.E. + 2 b.c.

(**) $\Rightarrow$ i.c.

(G1)

Given: $f, g, \bar{h}, u_0$ as in (S)

Find: $u^h: [0,T] \to S^h_{g(t)}$ $\quad \forall w \in V^h$

$$(\rho c \dot{u}^h, w^h) + a(u^h, w^h) = (f, w^h) + (\bar{h}, w^h)_{\Gamma}$$

$$(u^h(0) - u_0, w^h) = 0$$

G1 & G2 are the semi-discrete Galerkin eqns (t is still continuous)

(G2)

$v^h: [0,T] \to V^h$

$u^h = v^h + g^h \quad g^h: [0,T] \to S^h$

$$(\rho c \dot{v}^h, w^h) + a(v^h, w^h) = (f, w^h) + (\bar{h}, w^h)_{\Gamma} - (\rho c \dot{g}^h, w^h) - a(g^h, w^h)$$

$$(u^h(0) - u_0, w^h) = 0$$

$$\int (\rho c u_{,t} - \underbrace{[\kappa_{ij} u_{,j}]_{,i}}_{q_{i,i}}) w = \int w f \, d\Omega$$

$$\int (\rho c u_{,t} w) d\Omega - \int \{ w \kappa_{ij} u_{,j} \}_{,i} + \{ w_{,i} \kappa_{ij} u_{,j} \} \, d\Omega = \int w f \, d\Omega$$

Green's Thm. $- \int_\Gamma w (\kappa_{ij} u_{,j}) \, d\Gamma \, n_i$

now $q_i n_i = \bar{h}$ on Γ_h — what about Γ_g? it is assumed that $w = 0$ on Γ_g

move

$$(\rho c \dot{u}, w) - \int w \bar{h} \, d\Gamma + a(w,u) = (w,f) + (w,\bar{h})_{\Gamma_h}$$

$$\int w (u(x,0) - u_0) \, d\Omega = 0 \implies (w, u(0) - u_0) = 0$$

and if $u = v + \tilde{g} \Rightarrow$

$$(\rho c \dot{v}, w) + a(w,v) = (w,f) + (w,\bar{h})_{\Gamma_h} - (\rho c \dot{\tilde{g}}, w) - a(w,\tilde{g})$$

and $(w, v(0) - v_0) = (w, \tilde{g}_0 - \tilde{g}(0))$

$$(w, v(0)) = (w, u_0) - (w, \tilde{g}(0))$$

$$(\rho c \textstyle\sum N_a \dot{d}_a, \sum N_B c_B) + a(\sum N_B c_B, \sum N_a d_a) = (\sum N_B c_B, f) + (\sum N_B c_B, \bar{h}) - (\sum N_B c_B, \rho c \sum N_a \dot{g}_a) - a(\sum N_B c_B, \sum N_a g_a)$$

Then for any $\sum c_B$ $\{ (\rho c \sum N_a \dot{d}_a, N_B) + a(N_B, \sum N_a d_a) = (N_B, f) + (N_B, \bar{h})_{\Gamma_h} - (N_B, \rho c \sum N_a \dot{g}_a) - a(N_B, \sum N_a g_a)$

let $\int \rho c N_a N_B \, d\Omega = [m_{aB}]$ $\qquad \int N_B N_A \, d\Omega = [k_{ab}]$

$\underset{\sim}{M} \dot{\underset{\sim}{d}} + \underset{\sim}{K} \underset{\sim}{d} = \underset{\sim}{F}$ where $\underset{\sim}{F}$ is the RHS $- \sum m^e_{ab} \dot{g}_a - \sum k^e_{ab} g_a$

and since $u(0) = u_0$

then if $u(0) = v(0) + g(0) = \sum N_a \underset{\sim}{d}_a(0) = u_0 = \sum N_A d_{A_0}$

$\underset{\sim}{d}(0) = \underset{\sim}{d}_0$

from $(w, v(0)) = (w, u_0) - (w, \tilde{g}(0))$

$\Rightarrow (N_B, \sum N_A \underset{\sim}{d}_{A_0}) = (N_B, u_0) - (N_B, \sum N_A \tilde{g}_A(0))$

$\underset{\sim}{\bar{M}} \underset{\sim}{d}_0 = \int N_B u_0 \, d\Omega - \underset{\sim}{\bar{M}} \tilde{g}(0) = \underset{\sim}{\bar{d}}$

$\underset{\sim}{d}_0 = \underset{\sim}{\bar{M}}^{-1} [\underbrace{\int N_B u_0 \, d\Omega - \underset{\sim}{\bar{M}} \tilde{g}(0)}_{\underset{\sim}{\bar{d}}}]$ $\qquad \underset{\sim}{\bar{M}} \tilde{g}(0) = \sum_{q=1}^{nem} \bar{m}_{Pq} \tilde{g}_q$

FINITE ELEMENTS

LAST TIME: INCOMPATIBLE MODES

REMARK: THE ASSUMPTION $f^e_{ia} = 0$ $a=5,6$ ARISES BECAUSE SINCE WE WOULD LIKE $\sum_{a=1}^{6} f^e_{ia} = \int_{\Omega^e} f_i \, d\Omega$

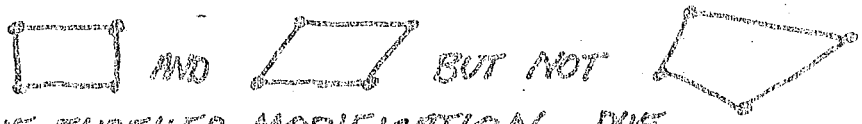
I.E. THE SUM OF THE NODAL FORCES SHOULD EQUAL THE TOTAL FORCE ON THE ELEMENT

BUT IF WE INCLUDE N_5, N_6 IN THE FORCE CALC. WE END UP OVERLOADING THE ELEMENT SINCE $\sum_{a=1}^{4} N_a = 1 \Rightarrow \sum_{a=1}^{6} f^e_{ia} \to \int_{\Omega^e} f_i \, d\Omega$

COMMENTS:

1.) EL. FUNCTIONS ARE NO LONGER CONTINUOUS.

2.) THE EL.S CONVERGE FOR LINEAR TRANSFORMATIONS ONLY

I.E.  AND BUT NOT

3.) NEED ONE FURTHER MODIFICATION DUE TO TAYLOR (REF. 3.) TO OBTAIN CONVERGENCE IN ARBITRARY CONFIGURATIONS

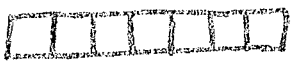
$$B_5 = \begin{bmatrix} N_{a,x} & 0 \\ 0 & N_{a,y} \\ N_{a,y} & N_{a,x} \end{bmatrix} \qquad B_5 = \frac{2}{J} \begin{bmatrix} -\xi \bar{y}_{,\eta} & 0 \\ 0 & \xi \bar{x}_{,\eta} \\ \xi \bar{x}_{,\eta} & -\xi \bar{y}_{,\eta} \end{bmatrix}$$

$$B_6 = \frac{2}{J} \begin{bmatrix} \eta \bar{y}_{,\xi} & 0 \\ 0 & -\eta \bar{x}_{,\xi} \\ -\eta \bar{x}_{,\xi} & \eta \bar{y}_{,\xi} \end{bmatrix}$$

TAYLORS MODIFICATION REPLACES $(\bar{\ })$ QUANTITIES BY THEIR VALUES AT $\xi = \eta = 0$

NOTE: THIS MODIFICATION IS DONE IN SHAPE ϕ

REMARKS: Incompatible Modes

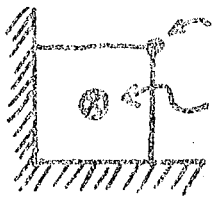
1.) WORKS WELL FOR BENDING: 
WITH JUST ONE ELEMENT THROUGH THE THICKNESS.

2.) ITS A BIT EXPENSIVE

3.) THE 3-D GENERALIZATION IS OBVIOUS,
ADD $1-\xi^2$, $1-\eta^2$, $1-\zeta^2$

4.) THE AXISYMMETRIC CASE DOESN'T CONVERGE.

5.) GOOD CONSTRAINT INDEX,



$\Rightarrow 2+4=6$ eq's/el.

CONSTRAINTS: $w^h = 1, x, y, x^2, xy, y^2$? trial fns

$0 = \varepsilon_{ij} \Rightarrow$ 3 CONSTRAINTS at center 1 pt quad

$CI = 6-3 = +3$
$CR = 6/3 = 2$ } VERY GOOD !!

HOURGLASS STIFFNESS: ? uniform reduced integ 1 pt.

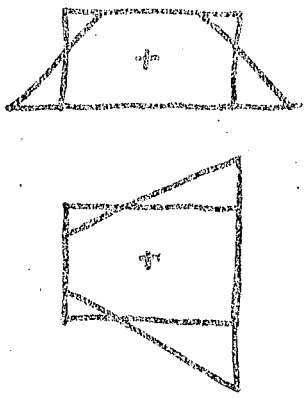
OBSERVATIONS — 1 PT. QUAD. (U_1) ON BILIN. EL.

IS 1.) OPT. FAST

2.) GIVES FAVORABLE $CI = 2-1 = 1$

3.) EL. IS SINGULAR i.e. THERE ARE TWO additional SPURIOUS ZERO ENERGY MODES OF k^e.
k^e USUALLY HAS THREE ZERO ENERGY MODES CORRESPONDING TO
a) TWO RIGID BODY TRANSLATIONS.
b) ONE RIGID BODY ROTATION.

THE TWO ADDITIONAL MODES ARE TERMED "HOURGLASS MODES"



$u_1 = c \begin{bmatrix} \xi \eta \\ x y \end{bmatrix}$

$u_2 = c \begin{bmatrix} \xi \eta \\ x y \end{bmatrix}$

EQUIVALENT WITH PROPER EL. ORIENTATION

$$0 = \underline{d}^T \underline{k}^e \underline{d} = \int_{\Omega^e}^{} c_{ijkl} u^h_{(k,l)} u^h_{(i,j)} \, d\Omega$$

$$\left.\begin{aligned} u_{i,x} &= u_{i,\xi} = c_1 \eta \\ u_{i,y} &= u_{i,\eta} = c_2 \xi \end{aligned}\right\} \Rightarrow u_{i,xy}(0,0) = 0$$

KOSLOFF: ADDS TO IT QUAD. STIFFNESS & STIFFNESS ASSOCIATED WITH THESE MODES.

REMARKS:

1.) IT TURNS OUT THIS EL. IS EXACTLY THE INCOMPATIBLE MODES EL. IN A RECTANGULAR CONFIGURATION.

2.) APPEARS TO BEHAVE BETTER THAN INCOMP. MODES IN ARBITRARY CONFIGURATIONS.

3.) MUCH CHEAPER TO FORMULATE THAN INCOMP. MODES.

4.) ERROR IN KOSLOFFS PAPER

5.) CODED IN LEGRN SUBROUTINE PLANE1 LINES 465→480 USES FORMS1 AND FORMS2

NOTE: COMPARE ELEMENT [redacted] IN HW PROB ?

formation time

ENGINEERING CRITERIA FOR ELEMENTS

RULES OF THUMB FOR REDUCED/SELECTIVE INTEGRATION AND INCOMPATIBLE MODES (SO CALLED NONCONFORMING ELEMENTS)

1.) INVARIANCE CONDITION – { RESULTS MUST BE INDP. OF REFERENCE FRAME.
2.) PATCH TEST – { VERIFICATION OF COMPLETENESS

THE REDUCED/SELECTIVE INTEGRATION EL. AND INCOMPATIBLE MODES (NON-CONFORMING) EL. BREAK THE "GALERKIN RULES". IN FACT SO DO ALL NUMERICALLY INTEGRATED ELEMENTS IF THE INTEGRATION RULE IS NOT EXACT. FOR ISOPARAMETRIC EL.S THE INTEGRATION RULE, NO MATTER HOW HIGH, IS NEVER EXACT FOR EL. DOMAINS WHICH ARE NOT □ ▱ (i.e. ONES FOR WHICH $x = N_a x_a$, $y = N_a y_a$, $z = N_a z_a$ ARE NOT CONSTANTS) BECAUSE CHAIN RULES PUT J's IN THE DENOMINATOR. THEREFORE WE NEED TO INTEGRATE RATIONAL FUNCTIONS, NOT POLYNOMIALS.
HOW CAN WE ASCERTAIN THE VALIDITY, OR LACK OF SAME, FOR THESE ELEMENTS?

THE PATCH TEST

VERSION	ORIGINATOR
ENGINEERING	B. IRONS '65
MATHEMATICS	G. STRANG '73 (BASED ON IRONS)

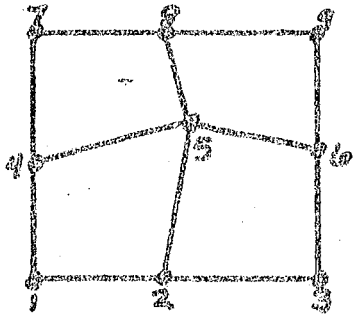
ENGINEERING VERSION OF PATCH TEST

1.) DONE ON THE COMPUTER. [i.e. IT CAN BE USED TO CHECK EL. PROGRAM]

2.) IT IMPLIES THE COMPLETENESS CONDITION AND TELLS US WHATEVER "TRICKS" WE EMPLOYED TO DERIVE THIS ELEMENT (INCOMP. MODES/ QUAD "TRICKS" ETC.) ARE IN FACT OK.

3.) THE PATCH TEST IS PASSED FOR AN ELEMENT IF THE DEFORMATION STATES $1, x, y$ (, 3) ARE EXACTLY REPRESENTABLE BY AN ARBITRARY "PATCH" OF ELEMENTS. (WHENEVER THE EXACT SOLUTION BEHAVES ACCORDINGLY)

EXAMPLE:



"ARBITRARY PATCH"

SET BOUNDARY NODES ($a = 1,2,3,4,7,8,9$) TO CORRECT VALUES LOOK AT COMPUTED SOLUTION $a = 5$

TEST #		u_a^h	v_a^h	
1	1	1	0	ALL $1, x, y$ STATES CORRECTLY REPRESENTED $\Rightarrow$ COMPLETENESS
	2	0	1	
x	3	x_a	0	
	4	0	x_a	
y	5	y_a	0	
	6	0	y_a	

THE SOLUTION AT $a=5$ MUST BE EXACT AND SO MUST BE THE DERIVATIVE GRADIENT (STRESSES AND STRAINS) WITHIN THE ELEMENTS.

HIGHER ORDER PATCH TEST $\Rightarrow$ HIGHER ORDER COMPLETENESS (x^2, xy, y^2 ETC.)

LINEAR TRANSIENT ANALYSIS

PARABOLIC CASE : HEAT EQUATION

(S)

GIVEN :

$$f : \Omega \times]0,T[\to \mathbb{R}$$
$$g : \Gamma_g \times]0,T[\to \mathbb{R}$$
$$\bar{h} : \Gamma_h \times]0,T[\to \mathbb{R}$$
$$u_0 : \Omega \to \mathbb{R}$$
$$T \neq \infty \quad (\text{FINITE})$$

FIND :

$$u : \bar{\Omega} \times [0,T] \to \mathbb{R} \ni$$
$$\rho c\, u_{,t} + q_{i,i} = f \quad \text{on } \Omega \times]0,T[$$
$$u = g \quad \text{on } \Gamma_g \times]0,T[\quad \Big\} \text{ BC's}$$
$$q_i n_i = -\bar{h} \quad \text{on } \Gamma_h \times]0,T[$$
$$u(x,0) = u_0(x) \quad x \in \Omega \quad \} \text{ IC}$$
$$q_i = -K_{ij} u_{,j}$$

(W)

GIVEN : $f, g, \bar{h}, u_0$ AS IN (S)

FIND :

$$u : [0,T] \to S_{g(t)} \ni \forall w \in V \quad (u = v + g)$$
$$(\rho c\, \dot{u}, w) + a(u,w) = (f,w) + (\bar{h},w)_{\Gamma} \quad (*)$$
$$(u(0) - u_0, w) = 0 \quad (**)$$

NOTE : $(\cdot,\cdot)$ $a(\cdot,\cdot)$ SAME AS FOR STATIC CASE.

EXERCISE : VERIFY EQUIVALENCE $(W) \Leftrightarrow (S)$

NOTE : $(*)$ & USUAL HYP. $\Rightarrow$ P.D.E. + 2 BC's

$(**) \Rightarrow$ IC

(G_1)

GIVEN: $f, g, \bar{h}, u_0$ AS IN (S)

FIND: $u^h : [0,T] \to S^h_{g(t)} \ni \forall\, w^h \in V^h$

$$(\rho c \dot{u}^h, w^h) + a(u^h, w^h) = (f, w^h) + (\bar{h}, w^h)_\Gamma$$

$$(u^h(0) - u_0, w^h) = 0$$

(G_2)

GIVEN: $f, g, \bar{h}, u_0$ AS IN (S)

FIND: $v^h : [0,T] \to V^h$

$$u^h = v^h + g^h \,, \quad g^h : [0,T] \to S^h$$

$$\ni \forall\, w^h \in V^h$$

$$(\rho c \dot{v}^h, w^h) + a(v^h, w^h) = (f, w^h) + (\bar{h}, w^h)_\Gamma - (\rho c \dot{g}^h, w^h) - a(g^h, w^h)$$

$$(v^h(0), w^h) = (u_0, w^h) - (g^h(0), w^h)$$

(G_1) & (G_2) ARE CALLED <u>SEMI-DESCRETE GALERKIN EQUATIONS</u> (t IS STILL CONT.)

ASSUME:

$$v^h(x,t) = \sum_{A \in \eta_0} N_A(x)\, d_A(t) \in V^h$$

$$g^h(x,t) = \sum_{A \in \eta_g} N_A(x)\, g_A(t) \in S^h$$

WHERE $u^h(x,t) = v^h(x,t) + g^h(x,t)$

$$\sum_{A \in \eta_0} N_A d_A + \sum_{A \in \eta_g} N_A g_A$$

NOTE: THE ABOVE IS JUST A FORM OF APPROXIMATION, NOT SEPARATION OF VARIABLES!

THE N_A's ARE THE USUAL SHAPE FUNCTIONS, THEY HAVE NO TIME DEPENDENCE.

FIND: $d : [0,T] \to \mathbb{R}^{NEQ}$ ∋

$$M \dot{d} + K d = F(t)$$

(MATRIX SYSTEM OF LINEAR O.D.E.'s)

$$d(0) = d_0$$

WHERE:

$$M = \mathop{A}_{e=1}^{NE} (m^e) \quad \text{... MASS OR CAPACITY MATRIX}$$

$$m^e = [m^e_{pq}]$$

$$m^e_{pq} = \int_{\Omega^e} \rho c N_a N_b \, d\Omega$$

$$K = \mathop{A}_{e=1}^{NE} (k^e) \quad \text{... CONDUCTIVITY MATRIX}$$

$$k^e = [k^e_{pq}]$$

$$k^e_{pq} = \int_{\Omega^e} B_a^T D B_b^T \, d\Omega$$

$$F(t) = F^{(e)}_{inact} + \mathop{A}_{e=1}^{NE} (f^e(t)) \quad \text{... FORCE VECTOR}$$

$$f^e = \{ f^e_p \}$$

$$f^e_p = \int_{\Omega^e} N_a \bar{f} \, d\Omega + \int_{\Gamma^e_h} N_a \bar{h} \, d\Gamma - \sum_{q=1}^{NEN} (k^e_{pq} g^e_q + m^e_{pq} \dot{g}^e_q)$$

(M)

AND THE CONSISTENT d_0 (INITIAL VALUE) IS GIVEN BY:

$$d_0 = \bar{M}^{-1} \{ \mathop{A}_{e=1}^{NE} (\bar{f}^e) \}$$

$$\bar{M} = \mathop{A}_{e=1}^{NE} (\bar{m}^e), \quad \bar{m}^e_{pq} = \int_{\Omega^e} N_a N_b \, d\Omega$$

$$\bar{f}^e_p = \int_{\Omega^e} u_0 N_p \, d\Omega - \sum_{q=1}^{NEN} \bar{m}^e_{pq} g^e_q(0)$$

comes from $\int w (u(0) - u_0) d\Omega = 0$

$$\Rightarrow \sum_a \int N_B N_a d_{0_a} d\Omega + \sum_a \int N_B \left(\sum_q N_q g_q(0) \right) d\Omega - \int N_B u_0 d\Omega = 0$$

$$0 = \bar{M} d_0 + \bar{m} g(0) - \int N_B u_0 d\Omega$$

$$\therefore \; d_0 = \bar{M}^{-1} \{ \int N_B u_0 d\Omega - \bar{m} g(0) \}$$

REMARKS:

1.) $\underline{M}$ (AND $\bar{\underline{M}}$) IS SYMMETRIC POSITIVE DEFINITE AND HAS THE SAME BAND PROFILE AS $\underline{K}$

2.) IT IS COMMON PRACTICE TO DIRECTLY SPECIFY $\underline{\dot{d}}_0$ THUS AVOIDING AN ADDITIONAL MATRIX SOLUTION.

3.) RECALL $p = N_{el}(a-1) + c^e$, FOR HEAT $N_{el} = 1$ THUS $\overset{i=1}{\therefore} p = (a-b) + 1 = a$

ONE-STEP ALGORITHMS FOR THE SEMI-DESCRETE HEAT EQUATION: GENERALIZED TRAPEZOIDAL METHOD

GIVEN: (THE FOLLOWING EQ.'S DERIVED IN (M)) F.D.

$$\underline{M}\,\underline{\dot{d}} + \underline{K}\,\underline{d} = \underline{F} \qquad \text{if } \underline{\dot{d}} \Rightarrow d_{n+1} = d_n + \Delta t\, \dot{d}_n \quad (1)$$

$$\underline{d}(0) = \underline{D}_0 \qquad (2)$$

WHERE:

$\underline{M}$ ... CAPACITY MATRIX; SYMM. POS. DEF.

$\underline{K}$ ... CONDUCTIVITY MATRIX, SYMM. POS. SEMI-DEF.

$\underline{F} = \underline{F}(t)\ t \in [0,T]$ ... HEAT SUPPLY VECTOR (PRESC.)

$\underline{d}$ ... TEMPERATURE VECTOR.

$\underline{\dot{d}}$ ... TIME (t) DERIVATIVE OF $\underline{d}$.

$\underline{D}_0$ ... GIVEN INITIAL TEMPERATURE.

APPLYING THE GENERALIZED TRAPEZOIDAL METHOD FOR FIRST ORDER SYSTEMS TO (1), WE GET

$$\underline{M}\,\underline{Y}_{n+1} + \underline{K}\,\underline{d}_{n+\alpha} = \underline{F}_{n+1} \qquad (3)$$

$$\underline{d}_{n+1} = \underline{d}_n + \Delta t\, \underline{Y}_{n+\alpha} \leftarrow \dot{d}\ \text{eqn} \qquad (4)$$

$$\underline{Y}_{n+\alpha} = (1-\alpha)\,\underline{Y}_n + \alpha\,\underline{Y}_{n+1} \qquad (5)$$

$$\underline{\dot{d}}_{n+1} = \underline{\dot{d}}_n + \Delta t\,[1-\alpha]\,\dot{d}_n + \Delta t\,\alpha\,\dot{d}_{n+1}$$

NOTATION:

n_{ts} ... NUMBER OF TIME STEPS

$\Delta t = T/n_{ts}$ ... TIME STEP (ASSUMED CONSTANT)

$t_n = n\Delta t$ ... TIME n

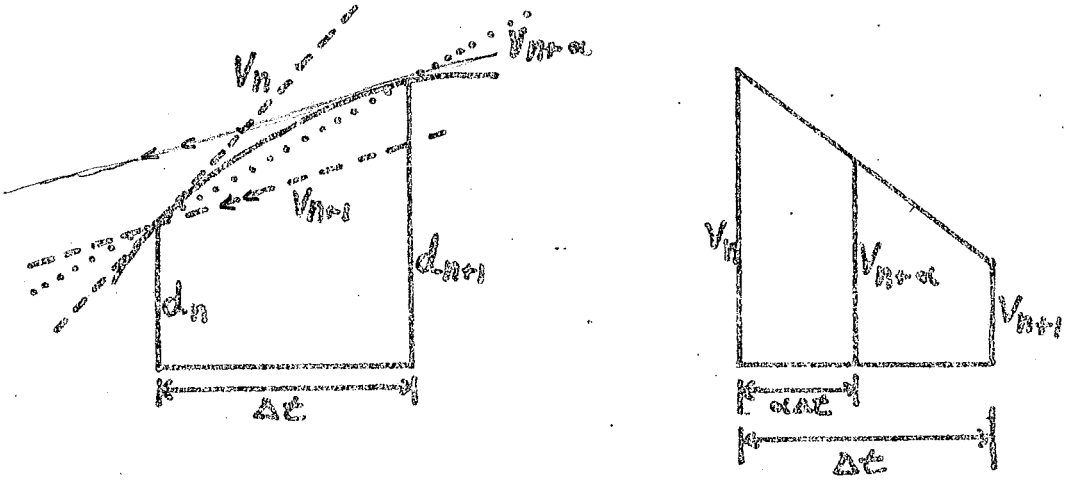
n ... INTEGER STEP NUMBER ($0 \le n \le n_{ts}$)

$d_n \approx d(t_n)$ ... APPROX. TEMP.

$v_n \approx \dot{d}(t_n)$ ... APPROX. TIME DERIVATIVE OF TEMP.

$\alpha \in [0, 1]$ ... PARAMETER

ILLUSTRATION OF $v_{n+\alpha}$ (1 dof)



SOME WELL KNOWN MEMBERS OF THE GENERALIZED TRAPAZOIDAL FAMILY,

α	METHOD
0	FORWARD DIFFERENCES; FORWARD EULER
1/2	TRAPAZOIDAL RULE; MIDPOINT RULE; CRANK-NICOLSON
2/3	GALERKIN
1	BACKWARD DIFFERENCES; BACKWARD EULER

REMARKS:

1) FOR $\alpha = 0$ THE METHOD IS SAID TO BE EXPLICIT. IF $\alpha = 0$ AND M IS LUMPED (DIAGONAL) THEN IT CAN BE SEEN (3)-(5) THAT THE SOLUTION CAN BE ADVANCED WITHOUT THE NECESSITY OF SOLVING MATRIX EQUATIONS.

2) FOR $\alpha \neq 0$ THE METHOD IS SAID TO BE IMPLICIT. IF $\alpha \neq 0$ A SYSTEM OF EQUATIONS WITH COEFFICIENT MATRIX $M + \alpha\Delta t K$ MUST BE SOLVED AT EACH STEP for $\underline{v}_k$ for $\underline{v}_k$ then put into two formulas of Pg 9 to get $\underline{d}_k$

DEFINITION: CONVERGENCE

WE CALL AN ALGORITHM CONVERGENT IF FOR EACH FIXED t_n, $\Delta t = t_n/n$

$$\underline{d}_n \to \underline{d}(t_n) \quad \text{AS} \quad \Delta t \to 0 \quad (n \to \infty)$$

TO BE SHOWN: STABILITY + CONSISTENCY $\Rightarrow$ CONVERGENCE

IN ORDER TO STUDY THE CHARACTERISTICS OF A TIME INTEGRATION ALGORITHM (I.E. ITS STABILITY, CONSISTENCY, ACCURACY AND CONVERGENCE) WE WILL EMPLOY THE MODAL APPROACH (ALSO CALLED SPECTRAL OR FOURIER ANALYSIS)

MODAL ANALYSIS: REDUCTION TO A SINGLE DEGREE-OF-FREEDOM (SDOF) MODEL EQUATION

WE WILL NOW DECOMPOSE (1) INTO n_{eq} UNCOUPLED SCALAR EQUATIONS. AND THUS THE ANALYSIS OF THE ALGORITHM WILL REDUCE TO THE ANALYSIS OF A SIMPLE SINGLE-DEGREE-OF-FREEDOM CASE.

FOR Transient Analysis

let $\underset{\sim}{d} = \underset{\sim}{\psi} e^{-\lambda t}$ $\lambda \geq 0$

$\dot{\underset{\sim}{d}} = \lambda \underset{\sim}{\psi} e^{-\lambda t}$

$\therefore \underset{\sim}{M}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = (-\lambda \underset{\sim}{M} + \underset{\sim}{K}) \underset{\sim}{\psi} e^{-\lambda t}$

look at homogeneous syste. first

$\therefore \not{e}^{-\lambda t} \neq 0 \Rightarrow$

$(\underset{\sim}{K} - \lambda \underset{\sim}{M})\underset{\sim}{\psi} = 0$

EIGENVALUE PROBLEM:

GIVEN: $\underset{\sim}{M}$ ... SYMM. POS. DEF.

$\underset{\sim}{K}$ ... SYMM. POS. SEMI-DEF.

FIND: λ_ℓ, $\underset{\sim}{\psi}_\ell \in \mathbb{R}^{neq}$ $1 \leq \ell \leq neq$ $\ni$

$$(\underset{\sim}{K} - \lambda_\ell \underset{\sim}{M})\underset{\sim}{\psi}_\ell = \underset{\sim}{0} \quad \ell \in \{1, 2, \ldots, neq\} \tag{6}$$

WHERE $$0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{neq} \tag{7}$$

AND $$\underset{\sim}{\psi}_\ell^T \underset{\sim}{M} \underset{\sim}{\psi}_m = \delta_{\ell m} \quad \text{... ORTHONORMALITY} \tag{8}$$

RECALL THAT $\{\underset{\sim}{\psi}_\ell\}_1^{neq}$ FORMS A BASIS FOR $\mathbb{R}^{neq}$ AND THAT $\delta_{\ell m} = \begin{cases} 0 & \ell \neq m \\ 1 & \ell = m \end{cases}$ (KRONECKER DELTA)

FROM (8) (ORTHONORMALITY)

$$\underset{\sim}{\psi}_\ell^T \underset{\sim}{K} \underset{\sim}{\psi}_m = \lambda_\ell \delta_{\ell m} \quad \text{(NO SUM)} \tag{9}$$

USING THE ABOVE WE WILL DECOMPOSE FIRST THE ORIGINAL SYSTEM OF O.D.E.s (1), AND SECOND THE GENERALIZED TRAPEZOIDAL ALGORITHM (3)–(5)

FIRST, LET

$$\underset{\sim}{d}(t) = \sum_{m=1}^{neq} d_{(m)}(t)\, \underset{\sim}{\psi}_m \quad \text{now} \quad \underset{\sim}{\psi}_\ell^T \underset{\sim}{M}\, \underset{\sim}{d}(t) = \sum \overset{d_m(t)}{(8)} \delta_{\ell m} = d_\ell$$

THUS

$$\dot{\underset{\sim}{d}}(t) = \sum_{m=1}^{neq} \dot{d}_{(m)}(t)\, \underset{\sim}{\psi}_m \quad \text{and} \quad \underset{\sim}{\psi}^T \underset{\sim}{M}\, \dot{\underset{\sim}{d}} = \dot{d}_\ell(t) \quad \overset{\text{similarly}}{(11)}$$

WHERE $d_{(m)}(t)$ ARE CALLED FOURIER COEFFICIENTS,

$$d_{(\ell)}(t) = \underset{\sim}{\psi}_\ell^T \underset{\sim}{M}\, \underset{\sim}{d}(t) \tag{12}$$

$$\dot{d}_{(\ell)}(t) = \underset{\sim}{\psi}_\ell^T \underset{\sim}{M}\, \dot{\underset{\sim}{d}}(t) \tag{13}$$

SUBSTITUTION OF (10) & (11) INTO (3) YIELDS

$$\sum_{m=1}^{n_{eq}} \left(\ddot{d}_{(m)} \Psi_\ell^T M \Psi_{(m)} + d_{(m)} \Psi_\ell^T K \Psi_{(m)} \right) = \Psi_\ell^T E \qquad (14)$$

THUS $\ddot{d}_{(\ell)} + \lambda_\ell d_{(\ell)} = F_{(\ell)}$ ($\lambda_\ell \delta_{\ell m}$, $\delta_{\ell m}$) (15)

WHERE $F_{(\ell)}(t) = \Psi_\ell^T E(t)$

THE INITIAL CONDITION REDUCES TO

$$d_{(\ell)}(0) = \Psi_\ell^T M d(0)$$

$$= \Psi_\ell^T M D_0$$

$$\stackrel{\text{DEF.}}{=} D_{0(\ell)} \qquad (16)$$

TO SIMPLIFY SUBSEQUENT WRITING DROP ℓ SUBSCRIPT. THUS OUR TYPICAL MODAL INITIAL-VALUE PROBLEM BECOMES

$$\ddot{d} + \lambda d = F \quad , \quad t \in [0, T] \qquad (17)$$

$$d(0) = D_0 \qquad (18)$$

THIS IS SOMETIMES CALLED THE SINGLE-DEGREE-OF-FREEDOM MODEL PROBLEM.

A SIMILAR DECOMPOSITION OF THE GENERALIZED TRAPAZOIDAL METHOD (3)-(5) YIELDS

$$d_n = \sum_{m=1}^{n_{eq}} d_{n(m)} \Psi_m \qquad (19)$$

$$d_{n+1} = \sum_{m=1}^{n_{eq}} d_{n+1(m)} \Psi_m \qquad (20)$$

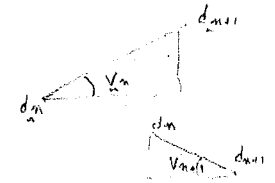
WHERE $d_{n(\ell)} = \Psi_\ell^T M d_n \qquad (21)$

$$d_{n+1(\ell)} = \Psi_\ell^T M d_{n+1} \qquad (22)$$

let $\dot{\underset{\sim}{d}} = \underset{\sim}{V}$

$$\underset{\sim}{M}\,\dot{\underset{\sim}{V}} + \underset{\sim}{K}\,\underset{\sim}{d} = \underset{\sim}{F}$$

then $\underset{\sim}{M}\,\dot{\underset{\sim}{V}}_{n+1} + \underset{\sim}{K}\,\underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1}$



for $\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\,\underset{\sim}{V}_n$ forward diff $\alpha = 0$ $\quad \frac{\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n}{\Delta t} = \underset{\sim}{V}_n$

$\underset{\sim}{V}_{n+\alpha} = (1-\alpha)\,\underset{\sim}{V}_n + \alpha\,\underset{\sim}{V}_{n+1} \Rightarrow \underset{\sim}{V}_n = 1\cdot\underset{\sim}{V}_n + 0\cdot\underset{\sim}{V}_{n+1}$

---

for $\underset{\sim}{d}_n = \underset{\sim}{d}_{n+1} - \Delta t\,\underset{\sim}{V}_{n+1}$ backward diff $\alpha = 1 \Rightarrow \underset{\sim}{d}_{n+1} = \Delta t\,\underset{\sim}{V}_{n+1} + \underset{\sim}{d}_n$; $\frac{\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n}{\Delta t} = \underset{\sim}{V}_{n+1}$

$\underset{\sim}{V}_{n+\alpha} = (1-\alpha)\,\underset{\sim}{V}_n + \alpha\,\underset{\sim}{V}_{n+1} \Rightarrow \underset{\sim}{V}_{n+1} = 0\cdot\underset{\sim}{V}_n + 1\cdot\underset{\sim}{V}_{n+1}$

---

for $\underset{\sim}{V}_{n+1/2}$: $\frac{\underset{\sim}{V}_{n+1} + \underset{\sim}{V}_n}{2} = \alpha\,\underset{\sim}{V}_{n+1} + (1-\alpha)\,\underset{\sim}{V}_{n+1}$ centered diff

$$\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\,\underset{\sim}{V}_{n+1/2} \longleftarrow \underset{\sim}{V}_{n+1/2} = \frac{\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n}{2\cdot\Delta t/2}$$

proof $\quad \underset{\sim}{d}_{n+1} = \underset{\sim}{d}_{n+1/2} + \frac{\Delta t}{2}\,\underset{\sim}{V}_{n+1/2}$; $\underset{\sim}{d}_n = \underset{\sim}{d}_{n+1/2} - \frac{\Delta t}{2}\,\underset{\sim}{V}_{n+1/2}$ $\Big\}$ $\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n = \Delta t\,\underset{\sim}{V}_{n+1/2}$

thus $\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\,\underset{\text{lumped}}{\underset{\sim}{V}_{n+\alpha}} = \underset{\sim}{d}_n + \Delta t\,\{(1-\alpha)\underset{\sim}{V}_n + \alpha\,\underset{\sim}{V}_{n+1}\}$.

if $\alpha = 0$ explicit & $\underset{\sim}{M}$ is diag then from the above $\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\,\underset{\sim}{V}_n$ $\therefore$

$$\underset{\sim}{M}\,\underset{\sim}{V}_{n+1} = \underset{\sim}{F}_{n+1} - \underset{\sim}{K}\,(\underset{\sim}{d}_n + \Delta t\,\underset{\sim}{V}_n) = \text{known rhs} \quad \therefore \quad (V_i)_{n+1} = (F_{n+1} - K\,d_{n+1})_i \;/\,[M_{ii}]$$

---

if $\alpha \neq 0$ then since $\underset{\sim}{K}$ is not diag then $\underbrace{\{\underset{\sim}{M} + \alpha\Delta t\,\underset{\sim}{K}\}}_{\text{not diag}}\underset{\sim}{V}_{n+1} = \underset{\sim}{F}_{n+1} - \underset{\sim}{K}\,\underbrace{\{\underset{\sim}{d}_n + \Delta t\,(1-\alpha)\underset{\sim}{V}_n\}}_{\text{known rhs}}$

this is implicit

solve for $\underset{\sim}{V}_{n+1}$; put into $\underset{\sim}{V}_{n+\alpha} = (1-\alpha)\underset{\sim}{V}_n + \alpha\,\underset{\sim}{V}_{n+1}$ to get $\underset{\sim}{V}_{n+\alpha}$; put into $\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\,\underset{\sim}{V}_{n+\alpha}$ to get $\underset{\sim}{d}_{n+1}$

Now $\alpha\,\underset{\sim}{M}\,\underset{\sim}{V}_{n+1} + \alpha\underset{\sim}{K}\,\underset{\sim}{d}_{n+1} = \alpha\,\underset{\sim}{F}_{n+1}$

$(1-\alpha)\,\underset{\sim}{M}\,\underset{\sim}{V}_n + \underset{\sim}{K}(1-\alpha)\,\underset{\sim}{d}_n = (1-\alpha)\,\underset{\sim}{F}_n$

$$\underset{\sim}{M}\left\{\frac{\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n}{\Delta t}\right\} + \underset{\sim}{K}\left\{(1-\alpha)\underset{\sim}{d}_n + \alpha\,\underset{\sim}{d}_{n+1}\right\} = \alpha\,\underset{\sim}{F}_{n+1} + (1-\alpha)\underset{\sim}{F}_n$$

$$\therefore \quad (\underset{\sim}{M} + \alpha\underset{\sim}{K}\Delta t)\,\underset{\sim}{d}_{n+1} = \{\underset{\sim}{M} - \Delta t\,\underset{\sim}{K}(1-\alpha)\}\underset{\sim}{d}_n + \Delta t\,\underset{\sim}{F}_{n+\alpha}$$

$\underset{\sim}{F}_{n+\alpha} = \alpha\,\underset{\sim}{F}_{n+1} + (1-\alpha)\,\underset{\sim}{F}_n$

Now let $\underset{\sim}{\psi}_i^T$ ( ) be applied to both sides of eqn w/ $\underset{\sim}{d} = \underset{m}{d}^T\underset{\sim}{\psi}_m$ & $\underset{\sim}{\psi}_i^T\underset{\sim}{M}\,\underset{\sim}{\psi}_m = \delta_{im}$ $\underset{\sim}{\psi}_i^T\underset{\sim}{K}\,\underset{\sim}{\psi}_m = \lambda_i$

then $\quad [1 + \alpha\lambda_i\Delta t]\,d_{i_{n+1}} = [1 - \Delta t\lambda_i(1-\alpha)]\,d_{i_n} + \Delta t\,\underset{\sim}{\psi}_i^T\,\underset{\sim}{F}_{n+\alpha}$

w/ $\quad d(0) = \underset{\sim}{\psi}_i^T\underset{\sim}{M}\,\underset{\sim}{d}(0)$

$$\sum_{m=1}^{n_{eq}} \left[d_{n+1(m)} \psi_\ell^T (M + \alpha \Delta t K) \psi_m - d_{n(m)} \psi_\ell^T ((1-\alpha) \Delta t K) \psi_m \right] = \Delta t \psi_\ell^T E_{n+\alpha} \quad (23)$$

$$(1 + \alpha \Delta t \lambda_\ell) d_{n+1(\ell)} = (1 - (1-\alpha) \Delta t \lambda_\ell) d_{n(\ell)} + \Delta t F_{n+\alpha(\ell)} \quad (24)$$

$$d_{0(\ell)} = D_{0(\ell)} \quad (25)$$

WHERE $E_{n+\alpha} = (1-\alpha) E_n + \alpha F_{n+1}$

$F_{n+\alpha(\ell)} = \psi_\ell^T E_{n+\alpha}$

DROP THE ℓ^{TH} MODAL SUBSCRIPT

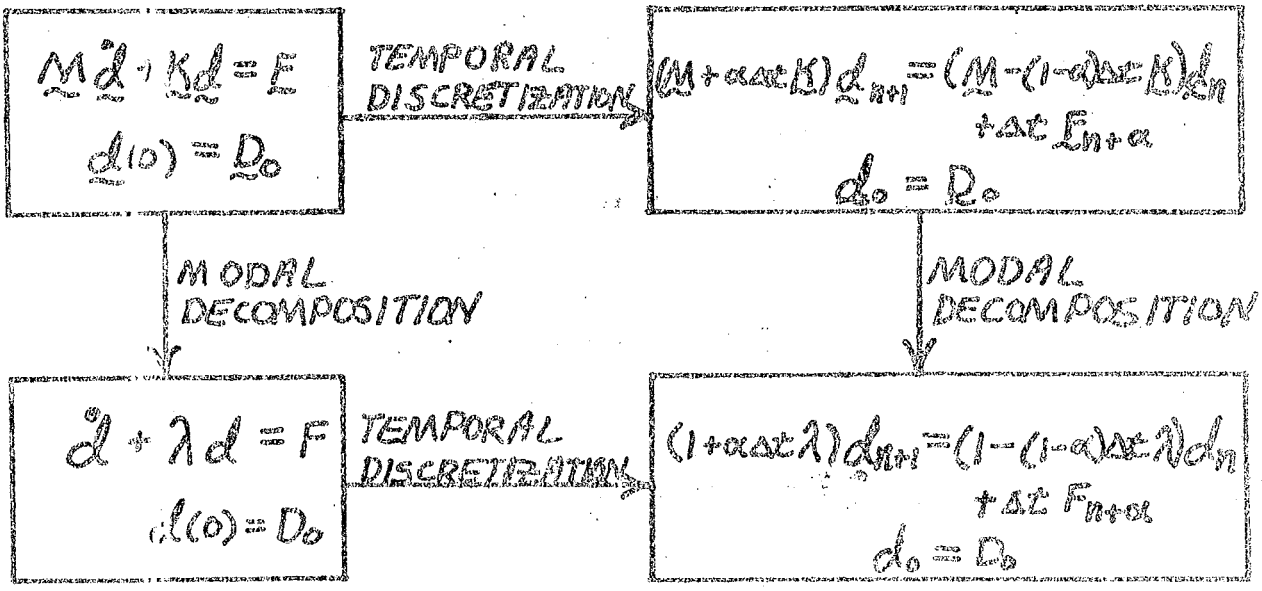
$$(1 + \alpha \Delta t \lambda) d_{n+1} = (1 - (1-\alpha) \Delta t \lambda) d_n + \Delta t F_{n+\alpha} \quad (26)$$

$$d_0 = D_0 \quad (27)$$

THIS IS SOMETIMES CALLED THE TEMPORALLY DISCRETIZED SDOF MODEL PROBLEM.

REMARKS:

1.) NOTE THAT DIRECTLY APPLYING THE GEN. TRAP METHOD TO (17) ALSO RESULTS IN (24). THIS FACT IS DEPICTED IN THE COMMUTATIVE DIAGRAM BELOW.



2.) THE CONVERGENCE OF $\underline{d}_n$ TO $\underline{d}(t_n)$ CAN BE ESTABLISHED BY SHOWING THE FOURIER COEFFICIENTS CONVERGE (i.e. $d_{n(\ell)}$ CONVERGES TO $d_{(\ell)}(t_n)$ FOR EACH $\ell \in \{1, 2, \ldots n_{eq}\}$).

LET $\underline{e}(t_n) = \underline{d}_n - \underline{d}(t_n)$ ... ERROR IN $\underline{d}_n$

$e_{(\ell)}(t_n) = d_{n(\ell)} - d_{(\ell)}(t_n)$ ... ℓ^{TH} FOURIER COEFF.

USING $\underline{M}$ (SYMM. POS. DEF.) TO DEFINE A NORM.

$$\underline{e}(t_n)^T \underline{M}\, \underline{e}(t_n) = \sum_{\ell,m=1}^{n_{eq}} \{e_{(\ell)}(t_n)\underline{\psi}_\ell\}^T \underline{M} \{e_{(m)}(t_n)\underline{\psi}_m\}$$

$$\underline{e}(t_n) = \sum_{\ell=1}^{n_{eq}} \{e_\ell(t_n)\underline{\psi}_\ell\}$$

$$= \sum_{\ell,m=1}^{n_{eq}} e_{(\ell)}(t_n) e_{(m)}(t_n) \underline{\psi}_\ell^T \underline{M}\, \underline{\psi}_m$$

$$= \sum_{\ell,m=1}^{n_{eq}} e_{(\ell)}(t_n) e_{(m)}(t_n)\, \delta_{\ell m}$$

$$= \sum_{\ell=1}^{n_{eq}} (e_{(\ell)}(t_n))^2$$

SO $\underline{e}(t_n)^T \underline{M}\, \underline{e}(t_n) \to 0$ IF AND ONLY IF $e_{(\ell)}(t_n) \to 0$ FOR EACH $\ell \in \{1, 2, \ldots n_{eq}\}$. SINCE $\underline{M}$ IS POSITIVE DEFINITE $\underline{e}(t_n)^T \underline{M}\, \underline{e}(t_n) \to 0$ IF AND ONLY IF $\underline{e}(t_n) \to 0$.

OR $e_{n(\ell)} \to 0 \Leftrightarrow \underline{e}(t_n) \to 0 \Leftrightarrow \underline{d}_n \to \underline{d}(t_n) \Rightarrow$ CONVERGENCE

THUS

CONVERGENCE OF SDOF PROB. $\Leftrightarrow$ CONVERGENCE OF SYSTEM

Hand it in (HW) a week from exam.

In a dynamic syste the nodal values become fns of time. However we are not assuming separation of variables.

Most common is trapezoidal method. Solve proble by working w/ the EV, $\overrightarrow{EV}$ to help decouple time/space coord.

① now we get $\underset{\sim}{M}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}$ w/ $\underset{\sim}{d}(0) = \underset{\sim}{D}_0$ from discretiz of FE proble

② Under temporal discretiz we get $\left(\underset{\sim}{M} + \alpha\Delta t \underset{\sim}{K}\right)\underset{\sim}{d}_{n+1} = \left(\underset{\sim}{M} - (1-\alpha)\Delta t \underset{\sim}{K}\right)\underset{\sim}{d}_n + \Delta t \underset{\sim}{F}_{n+\alpha}$
where $\underset{\sim}{F}_{n+\alpha} = (1-\alpha)\underset{\sim}{F}_{n+1} + \alpha \underset{\sim}{F}_n$

ⓐ Step $n+1$ based on results of step n

③ Using modal decomp on the original proble.

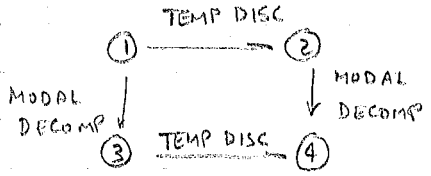
$$\dot{d} + \lambda d = F \qquad \forall\ \lambda \text{ of the syste.}$$

④ TEMP DISC

$$d_0 = D_0$$

If we use generaliz "trapez" algorith of the temporal discretiz we get

$$(1 + \alpha\Delta t\lambda)\, d_{n+1} = \left(1 - (1-\alpha)\Delta t\lambda\right) d_n + \Delta t\, F_{n+\alpha}$$
$$\text{w/}\quad d_0 = D_0$$



non linear problems will not necessarily satisfy this diagram.
if ① → ② → ④ ≠ ① → ③ → ④ then numerical problem may not converge to initial proble.

Convergence let $\underset{\sim}{d}_n$ be soln to matrix proble & $\underset{\sim}{d}(t_n)$ be soln to actual proble.
and define $\underset{\sim}{e}(t_n) = \underset{\sim}{d}_n - \underset{\sim}{d}(t_n)$ and look at temporal convergence.
We had shown that static problems are convergent, we will now look at discretized time and see whether the "quasi static" results are now convergent in time

now $\underset{\sim}{e}(t_n) \to 0$ if $e_{(i)}(t_n) = d_{n(i)} - d_{(i)}(t_n) \to 0$

Proof: Recall $\underset{\sim}{M}$ is $= \underset{\sim}{M}^T$ & $\underset{\sim}{c}^T \underset{\sim}{M}\, \underset{\sim}{c} > 0$ for $\underset{\sim}{c} \neq 0$

Now take $\underset{\sim}{e}^T(t_n)\underset{\sim}{M}\,\underset{\sim}{e}(t_n) = \sum^{neq} \left(e_{(i)}(t_n)\,\underset{\sim}{\psi}_{(i)}\right)^T \underset{\sim}{M} \left(e_{(m)}(t_n)\,\underset{\sim}{\psi}_{(m)}\right)$

$$= \sum_{\ell,m=1}^{neq} e_\ell(t_n)\, e_m(t_n)\, \underset{\sim}{\psi}_\ell^T \underset{\sim}{M}\, \underset{\sim}{\psi}_m = \sum_{\ell,m=1}^{neq} e_\ell\, e_m\, \delta_{\ell m} = \sum_{\ell=1}^{neq} e_\ell^2 > 0$$

if $\sum e_{(\ell)}^2 = 0 \Rightarrow e_{(\ell)} = 0 \;\therefore\; \underset{\sim}{e} = 0 \;\&\; \underset{\sim}{d}_n = \underset{\sim}{d}(t_n)$

QED

Stability of algorithm + consistency $\Rightarrow$ convergence.

Consistency is equivalent to accuracy. Stability will now be looked at

look at homogeneous syste $\dot{d} + \lambda d = 0 \Rightarrow d = ce^{-\lambda t}$ w/ $d = d(t_n)$ when $t = t_n$

$d(t_n) = ce^{-\lambda t_n}$ $\quad c = d(t_n)e^{\lambda t_n} \;\therefore\; d = d(t_n)e^{-\lambda(t-t_n)}$

EXACT sol.. $\quad d(t_{n+1}) = \exp\left\{-\lambda \underbrace{(t_{n+1} - t_n)}_{\Delta t}\right\} d(t_n)$ $\quad$ where $\lambda \geq 0$.

if $\Delta t > 0 \quad |d(t_{n+1})| < |d(t_n)|$ if $\lambda > 0$

if $\lambda = 0 \quad d(t_{n+1}) = |d(t_n)|$ since eqn becomes $\dot{d} = 0$ $\quad d = \text{const.}$

look at the homogeneous version of this discrete results

$$(1 + \alpha \Delta t \lambda)\, d_{n+1} = (1 - (1-\alpha)\Delta t \lambda)\, d_n$$

$\lambda \geq 0 \quad \Delta t > 0 \quad \alpha \in [0,1] \quad \therefore \quad 1 + \alpha \Delta t \lambda > 0$

$\therefore \quad d_{n+1} = A\, d_n \qquad A = \dfrac{(1 - (1-\alpha)\Delta t \lambda)}{1 + \alpha \Delta t \lambda}$ is the amplif factor

$A = 1 - \dfrac{\Delta t \lambda}{1 + \alpha \Delta t \lambda}$

A approxim $\exp(-\lambda \Delta t)$

we now ask that $|d_{n+1}| < |d_n|$ if $\lambda > 0$ as analog of contin. syste.

$|d_{n+1}| = |d_n|$ $\lambda = 0$

if $\lambda = 0 \quad A = 1 \;\therefore\; d_{n+1} = d_n$

if $\lambda > 0 \quad |A| < 1 \quad$ but $\quad A = 1 - \dfrac{\Delta t \lambda}{1 + \alpha \Delta t \lambda} \quad \therefore \quad \dfrac{\Delta t \lambda}{1 + \alpha \Delta t \lambda} > 0$

$-1 < A < 1$

leads to $\Rightarrow$ & $1 > \dfrac{\Delta t \lambda}{2(1 + \alpha \Delta t \lambda)} > 0$

$\therefore \quad 2 + 2\alpha \Delta t \lambda > \Delta t \lambda$

$2 > \Delta t \lambda (1 - 2\alpha)$

$\dfrac{2}{\Delta t \cdot \lambda} > (1 - 2\alpha) \quad \therefore \; \frac{1}{2}\left(1 - \dfrac{2}{\Delta t \lambda}\right) < \alpha \quad \therefore$

for stability $\alpha \geq \frac{1}{2}$ if $\Delta t, \lambda > 0$

if $\alpha < \frac{1}{2}$. $\quad \Delta t < \dfrac{2}{\lambda(1 - 2\alpha)}$ $\quad$ Δt depends on largest λ of syste ie λ_{max}.

Unconditional stability NO Δt limit (EG $\alpha \geq 1/2$)
Conditional stability $\exists$! a Δt restrict EG $\alpha < 1/2$

which is $\Delta t < \frac{2}{\lambda(1-2\alpha)}$

if $d_{n+1} = A d_n$ $d_n = A d_{n-1}$ $\therefore d_{n+1} = A^2 d_{n-1} = \dots A^{n+1} d_0$

if $|A| < 1 \Rightarrow A^n \to 0$; if $|A| > 1$ unstable.

where A^n grows

A \ n	100	1000
.99	.37	9.32×10^{-5}
1.01	2.7	2.09×10^{4}
.9	2.66×10^{-5}	1.75×10^{-46}
1.1	1.39×10^{41}	2.47×10^{41}

5/7/81

Midterm changed to Thursday 14 May

To summarize - stability for general trapezoidal methods is dependent on the amplification factor

$$A = \frac{1-(1-\alpha)\lambda\Delta t}{1+\alpha\lambda\Delta t} \quad \text{for } d_{n+1} = A d_n$$

The system is stable when $|A| < 1$ and $\lambda = \lambda_{neg}$

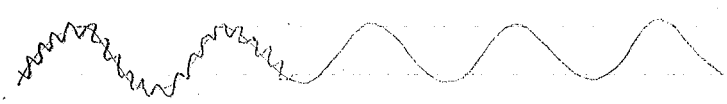
it is unconditionally stable for $\alpha \geq 1/2$
" " conditionally " " $\alpha < 1/2$ ie $\Delta t < 2/[(1-2\alpha)\lambda_{neg}]$

See handout:

1. if $\alpha = 1$ and let $\Delta t \to \infty$ this is backward difference

$\underset{\sim}{K}\, \underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1}$ (ie we tend to equilib)

2. if $\lambda\Delta t \gg 1$ for $\alpha = 1/2$ (trapezoid) high mode $A_\infty = -1$ since $\frac{1-\frac{1}{2}\Delta t\lambda}{1+\frac{1}{2}\Delta t\lambda} \to -1$
$\therefore d_{n+1} = -d_n$ $\therefore d_n = (-1)^n d_0$



wwww - superposed due to the flip-flop of the high mode

to filter the oscill out report only the average $\frac{d_{n+1}+d_n}{2} = \frac{(-1)^{n+1}+(-1)^n}{2} = 0$
this technique removes high mode problems.
the midpoint defines the oscillation limit ~~~~

Consistency and convergence

we had looked at homog system. Now must included load

$\therefore \quad d_{n+1} = A d_n + L_n \qquad (1)$

$L_n = F_{n+\alpha}\Delta t/(1+\alpha\Delta t\lambda)$

this is the algorithm

the exact can be represented by $d(t_{n+1}) = A\,d(t_n) + L_n + \tau(t_n) \qquad (2)$

where we are fitting the exact algorithm to the algorithm and looking at the difference between (1) & (2), which defines the truncation error.

Now if $|\tau(t)| \le c\,\Delta t^{k+1} \qquad t\in[0,T]$ with $c \neq c(\Delta t) \qquad k>0$

then the algorithm is said to be consistent and k is defined as the order of accuracy of algorithm. ie rate of convergence.

Facts: for generalized trapez family $k=1 \quad \alpha\in[0,1]$
for $\dot d+\lambda d = F$ $k=2 \quad \alpha = \frac12$ (trapez rule).

To show this take $d(t_{n+1})$, $d(t_n)$, $F(t_{n+1})$ } expand about $t_{n+\alpha}$ using Taylor exp

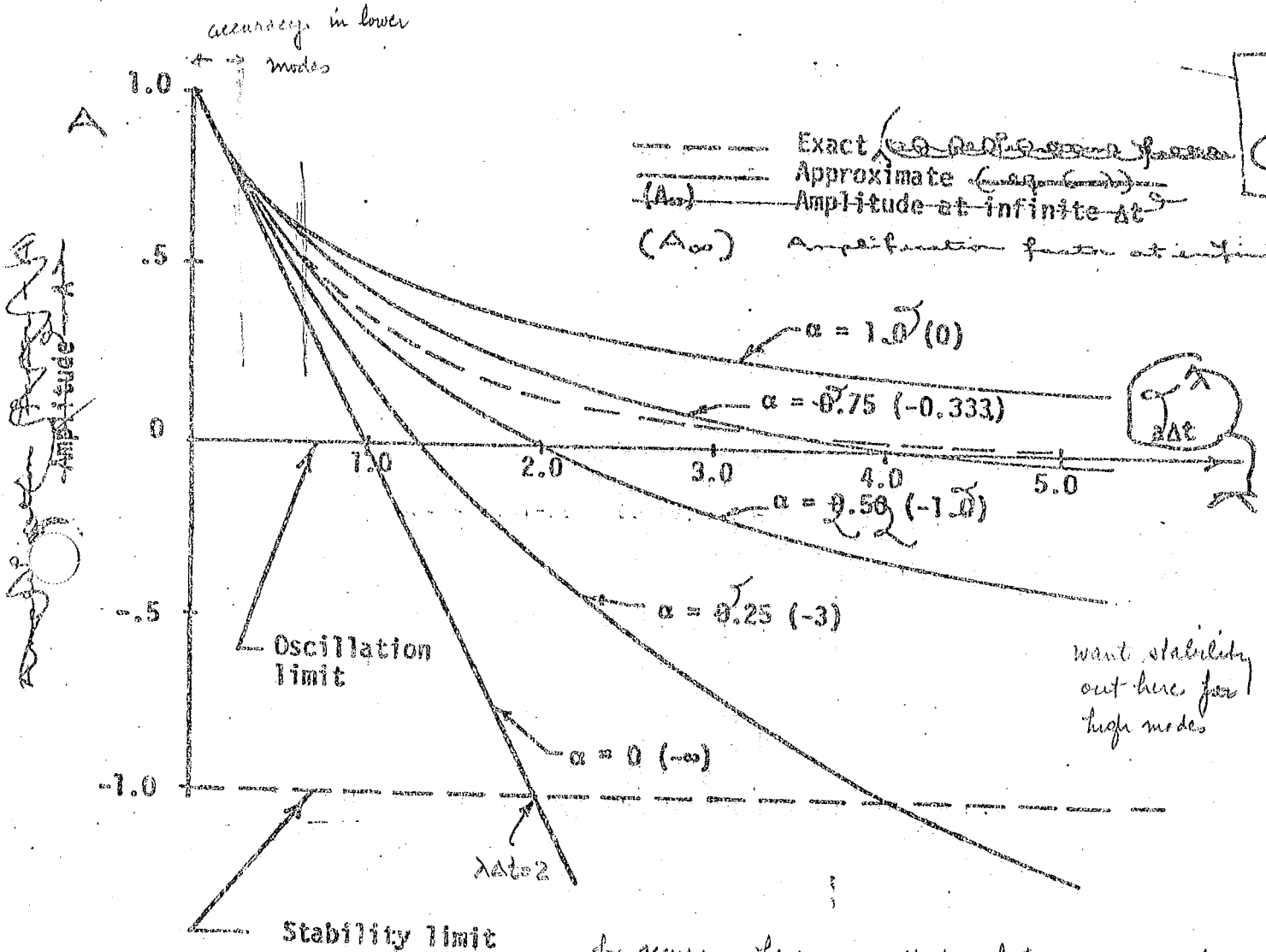
and use exact ie $(\dot d+\lambda d = F)$ at $t_{n+\alpha}$ also use time deriv of it.
eqn.

EXERCISE do this (might be on exam).

Theorem

Assume t_n is fixed ($n, \Delta t$ vary). ie $n\to\infty$ $\Delta t\to 0$.
Consider (1) and (2) and assume (i) stability $|A|\le 1$
(ii) consistency $|\tau(t)| \le c\Delta t^{k+1}$ $k>0$
then $e(t_n) = d_n - d(t_n) \to 0$ as $\Delta t\to 0$.

Subtracting (2) from (1) then
$e(t_{n+1}) = A\,e(t_n) - \tau(t_n)$ this is the error eqn.
now $e(t_n) = A\,e(t_{n-1}) - \tau(t_{n-1})$
$\therefore \quad e(t_{n+1}) = A^2 e(t_{n-1}) - A\,\tau(t_{n-1}) - \tau(t_n)$
$e(t_{n+1}) = A^3 e(t_{n-2}) - A^2\tau(t_{n-2}) - A\,\tau(t_{n-1}) - \tau(t_n)$



for accuracy here use $\alpha = \frac{1}{2}$ but use $\alpha = 1$ for stability of higher modes

Fig. 2 Integration Formula Characteristics

Figure 1 Amplification factor for typical one-step method.

etc

$$e(t_{n+1}) = A^{n+1} e(0) - \sum_{i=0}^{n} A^i \tau(t_{n-i})$$

but $e(0) = d_0 - d(0) = 0$

$$\therefore \quad e(t_n) = -\sum_{i=0}^{n-1} A^i \tau(t_{n-1-i}) \quad \Rightarrow \quad |e(t_n)| = \left| \sum \right| \leq \sum_{i=0}^{n-1} |A|^i \max_{t \in [0,t]} |\tau(t)|$$

using stability $|e(t_n)| \leq \sum_{i=0}^{n-1} \max_{t \in [0,T]} |\tau(t)| \leq n c \Delta t^{k+1} \leq t_n \cdot c \, \Delta t^k = O(\Delta t^k)$ $\Rightarrow |A| < 1$

use consistency; t_n fixed

as $\Delta t \to 0 \quad |e(t_n)| \to 0$

Since we've shown this for a single mode & we said that the discrete system is a superpos. of different modes, then $\Rightarrow$ the discrete system converges to the actual problem if $\Delta t \to 0$.

Let us look at the hypothetical case in elastodynamics. This is an IBVP.

$f_i : \Omega \times]0,t[\to \mathbb{R}$ forcing term $\quad \rho : \Omega \to \mathbb{R}$ (mass density)

(S) $g_i : \Gamma_{g_i} \times]0,T[\to \mathbb{R}$
$h_i : \Gamma_{h_i} \times (0,T) \to \mathbb{R}$ } BC's

Given IC: $u_{0_i} : \Omega \to \mathbb{R}$ (displ.)
$\dot{u}_{0_i} : \Omega \to \mathbb{R}$ (velocity)

we want to find $\Rightarrow$ $u_i : \overline{\Omega} \times [0,T] \to \mathbb{R}$ s.

$$\rho u_{i,tt} = \sigma_{ij,j} + f_i \quad \text{on } \Omega \times (0,T)$$

and $u_i = g_i$ on $\Gamma_{g_i} \times (0,T)$
$\sigma_{ij} n_j = h_i$ on $\Gamma_{h_i} \times (0,T)$

domain is fixed w/ time but data may vary

$u_i(\underset{\sim}{x},0) = u_{0_i}(\underset{\sim}{x})$
$u_{i,t}(\underset{\sim}{x},0) = \dot{u}_{0_i}(\underset{\sim}{x})$ } $\underset{\sim}{x} \in \Omega$

$\sigma_{ij} = C_{ijkl} \, u_{(k,l)}$ where $C_{ijkl} \neq C_{ijkl}(t)$

(W) Given $u_i : [0,T] \to \mathcal{S}_i$ s. $\forall w_i \in V_i$ (no time dependence here)

Then
$$(\underset{\sim}{w}, \rho \ddot{\underset{\sim}{u}}) + a(\underset{\sim}{w}, \underset{\sim}{u}) = (\underset{\sim}{w}, \underset{\sim}{f}) + (\underset{\sim}{w}, \underset{\sim}{h})_{\Gamma_1} \quad \Big] \text{ gives PDE + BC}$$

$$\to \int_\Omega w_i \rho \ddot{u}_i \, d\Omega$$

also
$$\left.\begin{aligned} (\underset{\sim}{w}, \underset{\sim}{u}(0) - \underset{\sim}{u}_0) &= 0 \\ (\underset{\sim}{w}, \dot{\underset{\sim}{u}}(0) - \dot{\underset{\sim}{u}}_0) &= 0 \end{aligned}\right\} \text{ IC in weak form}$$

We can establish under approp hypothesis (S) $\Leftrightarrow$ (W).

5/12/81

Review of last week

We were looking at the elastodynamics problem in its (S) $\Leftrightarrow$ (W) forms. we will write the (G1) form next

Given: f_i, g_i, h_i, $(u_{0i}, \dot{u}_{0i} = \text{IC})$.

Find:
$$u_i^h : [0,T] \to \mathcal{S}_i^h \text{ s. } \forall w_i^h \in V_i^h$$
($\mathcal{S}_i^h$ with g type bc included; w_i^h, V_i^h same as static prob.)

$$(w^h, \rho \ddot{\underset{\sim}{u}}^h) + a(\underset{\sim}{w}^h, \underset{\sim}{u}^h) = (\underset{\sim}{w}^h, \underset{\sim}{f}) + (\underset{\sim}{w}^h, \underset{\sim}{h})_{\Gamma}$$

with IC
$$\begin{aligned} (\underset{\sim}{w}^h, \underset{\sim}{u}^h(0) - \underset{\sim}{u}_0^h) &= 0 \\ (\underset{\sim}{w}^h, \dot{\underset{\sim}{u}}^h(0) - \dot{\underset{\sim}{u}}_0^h) &= 0 \end{aligned}$$

Now we separate into unknown + known form

let
$$v_i^h(\underset{\sim}{x}, t) = \sum_{\eta - \eta_{g_i}} N_a(\underset{\sim}{x}) \, d_{iA}(t) \quad \in V_i^h$$

(G2)
$$g_i^h(\underset{\sim}{x}, t) = \sum_{\eta_{g_i}} N_a(\underset{\sim}{x}) \, g_{iA}(t) \quad \in \mathcal{S}_i^h$$

Then we get if $\underset{\sim}{u}^h = \underset{\sim}{v}^h + \underset{\sim}{g}^h$
$$(\underset{\sim}{w}^h, \rho \ddot{\underset{\sim}{v}}^h) + a(\underset{\sim}{w}^h, \underset{\sim}{v}^h) = (\underset{\sim}{w}^h, \underset{\sim}{f}) + (\underset{\sim}{w}^h, \underset{\sim}{h})_{\Gamma} - (\underset{\sim}{w}^h, \rho \ddot{\underset{\sim}{g}}^h) - a(\underset{\sim}{w}^h, \underset{\sim}{g}^h)$$

and
$$\begin{aligned} (\underset{\sim}{w}^h, \underset{\sim}{v}^h(0)) &= (\underset{\sim}{w}^h, \underset{\sim}{u}_0^h - \underset{\sim}{g}^h(0)) \\ (\underset{\sim}{w}^h, \dot{\underset{\sim}{v}}^h(0)) &= (\underset{\sim}{w}^h, \dot{\underset{\sim}{u}}_0^h - \dot{\underset{\sim}{g}}^h(0)) \end{aligned}$$

The matrix formulation then becomes:

Find: $\underset{\sim}{d} : [0,T] \to \mathbb{R}^{n_{eq}}$ s.t.

$$\underset{\sim}{M}\ddot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}(t)$$

where $\underset{\sim}{K}$ is same as before for elastostatics

where $\underset{\sim}{M} = \overset{n_{el}}{\underset{e=1}{A}} (\underset{\sim}{m}^e)$ $\quad \underset{\sim}{m}^e = [m^e_{pq}]$ $\quad p = n_{ed}(a-1)+i$ $\quad q = n_{ed}(b-1)+j$ $\quad n_{ed} = n_{sd}$

$$[m^e_{pq}] = \int_{\Omega^e} \rho N_a N_b \delta_{ij}\, d\Omega$$

now $\underset{\sim}{f}^e = \{f^e_p\}$; $\quad f^e_p = \int_{\Omega^e} N_a f_i(t)\, d\Omega + \int_{\Gamma^e_h} N_a h_i\, d\Gamma$

$$- \sum_{q=1}^{n_{ee}} \left(k^e_{pq} g^e_q + m^e_{pq} \ddot{g}^e_q \right)$$

Now the IC's are that

$$\underset{\sim}{d}(0) = \underset{\sim}{d}_0 \; ; \quad \dot{\underset{\sim}{d}}(0) = \dot{\underset{\sim}{d}}_0$$

the discretized is for the IC's

$$\underset{\sim}{d}_0 = \underset{\sim}{\bar{M}}^{-1} \left\{ \overset{n_{el}}{\underset{e=1}{A}} (\bar{\underset{\sim}{d}}^e) \right\} \qquad \underset{\sim}{\bar{M}} = A(\underset{\sim}{\bar{m}}^e) \qquad \bar{m}^e_{pq} = \int_{\Omega^e} N_a N_b \delta_{ij}\, d\Omega$$

$$\bar{\underset{\sim}{d}}^e = \{\bar{d}^e_p\} \qquad \bar{d}^e_p = \int_{\Omega^e} N_a u_{0i}\, d\Omega - \sum_q \bar{m}^e_{pq} g^e_q(0)$$

$$\dot{\underset{\sim}{d}}_0 = \underset{\sim}{\bar{M}}^{-1} \left\{ \overset{n_{el}}{\underset{e=1}{A}} (\dot{\bar{\underset{\sim}{d}}}^e) \right\}$$

where $\dot{\bar{d}}^e_p = \int_{\Omega^e} N_a \dot{u}_{0i}\, d\Omega - \sum \bar{m}^e_{pq} \dot{g}^e_q(0)$

before we had

$$\underset{\sim}{M}\ddot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}(t)$$

and IC

now let's add a viscous damping matrix

$$\textcircled{*} \qquad \underset{\sim}{M}\ddot{\underset{\sim}{d}} + \underset{\sim}{C}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}(t) \qquad \text{where } \underset{\sim}{C} \text{ is a viscous damping matrix}$$

if $\underset{\sim}{K} = 0$ then we really have a 1st order system on $\dot{\underset{\sim}{d}}$ but this also leads to nonlinear analysis also. We will now look at integrators of the "NEWMARK Family" ie "one-step" algorithms.

We can write $\textcircled{*}$ as 3 eqns

$$\underset{\sim}{M}\underset{\sim}{a}_{n+1} + \underset{\sim}{C}\underset{\sim}{v}_{n+1} + \underset{\sim}{K}\underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1} = \underset{\sim}{F}(t_{n+1})$$

where $\underset{\sim}{a}_n \approx \underset{\sim}{a}(t_n) \quad \underset{\sim}{v}_n \approx \underset{\sim}{v}(t_n) \quad \underset{\sim}{d}_n \approx \underset{\sim}{d}(t_n)$

$\approx \ddot{\underset{\sim}{d}}(t_n) \qquad \approx \dot{\underset{\sim}{d}}(t_n)$

These are Taylor series approx.

$$\underset{\sim}{d}_{n+1} = \underset{\sim}{d}_n + \Delta t\, \underset{\sim}{v}_n + \frac{\Delta t^2}{2}\left\{(1-2\beta)\,\underset{\sim}{a}_n + 2\beta\,\underset{\sim}{a}_{n+1}\right\}$$

$$\underset{\sim}{v}_{n+1} = \underset{\sim}{v}_n + \Delta t\left\{(1-\gamma)\,\underset{\sim}{a}_n + \gamma\,\underset{\sim}{a}_{n+1}\right\}$$

where β is a parameter, γ " a " } they control accuracy & stability of algorithm

For the algorithmic eqns suppose we are given everything at $\underset{\sim}{d}_n, \underset{\sim}{v}_n, \underset{\sim}{a}_n$ and we want to find $\underset{\sim}{a}_{n+1}, \underset{\sim}{d}_{n+1}, \underset{\sim}{v}_{n+1}$

for the initial step.

$$\underset{\sim}{M}\,\underset{\sim}{a}_0 = \underset{\sim}{F}_0 - \underset{\sim}{C}\,\underset{\sim}{v}_0 - \underset{\sim}{K}\,\underset{\sim}{d}_0 \quad \text{now this is known.}$$

Now look at members of newmark family

Ex. 1. suppose we look at the (constant) average accel. (implicit) method $\beta = 1/4$ $\gamma = 1/2$

this is equiv to applying trapez rule to first order form of $\underset{\sim}{M}\ddot{\underset{\sim}{d}} + \underset{\sim}{C}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}$

what is first order form.

let $\underset{\sim}{y} = (\underset{\sim}{d}, \dot{\underset{\sim}{d}})^T$ let $\underset{\sim}{f} = \underset{\sim}{f}(\underset{\sim}{y}, t) = \begin{Bmatrix} \dot{\underset{\sim}{d}} \\ \underset{\sim}{M}^{-1}\left(\underset{\sim}{F}(t) - \underset{\sim}{C}\dot{\underset{\sim}{d}} - \underset{\sim}{K}\underset{\sim}{d}\right) \end{Bmatrix}$

Then $\dot{\underset{\sim}{y}} = \underset{\sim}{f}(\underset{\sim}{y}, t)$ the first order form. Now apply trapez form.

now $\underset{\sim}{z}_n \approx \dot{\underset{\sim}{y}}(t_n) \Rightarrow \underset{\sim}{y}_{n+1} = \underset{\sim}{y}_n + \frac{\Delta t}{2}\left(\underset{\sim}{z}_n + \underset{\sim}{z}_{n+1}\right)$ trapez. rule $\quad \underset{\sim}{y}_n \approx \underset{\sim}{y}(t_n)$

$\underset{\sim}{z}_{n+1} = \underset{\sim}{f}\left(\underset{\sim}{y}_{n+1}, t_{n+1}\right)$ trap is uncond. stable & 2nd order accurate.

exercise show that the above integrator is the newmark member w/ $\beta = 1/4$ $\gamma = 1/2$.

Ex. 2. if we use linear accel method $\beta = 1/6$, $\gamma = 1/2$ conditionally stable, as 2nd order.

this is an implicit method.

What is implicit or explicit

Implicit is a matrix problem needs to be solved to advance sol from step to step.

explicit — no matrix problem needs to be solved.

the conditionally stable syste is dependent on the highest e.value.

Ex. 3. Central difference method $\beta = 0$ $\gamma = 1/2$

ie $\underset{\sim}{v}_n = \left(\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_{n-1}\right)/2\Delta t$

$\underset{\sim}{a}_n = \left(\underset{\sim}{d}_{n+1} - 2\,\underset{\sim}{d}_n + \underset{\sim}{d}_{n-1}\right)/\Delta t^2$

2nd order accurate

explicit; conditionally stable

attractive due to 2nd order & explicit

if $\underline{C}=0$ and $\underline{M}$ is diag : $\underline{M}\,\underline{a}_{n+1} + \underline{K}\,d_{n+1} = \underline{F}_{n+1}$
by some method,

if $\beta=0$ $\quad \underline{d}_{n+1} = \underline{d}_n + \Delta t\,\underline{v}_n + \frac{\Delta t^2}{2}\underline{a}_n$; if $\underline{d}_n, \underline{v}_n, \underline{a}_n$ are known; $\underline{d}_{n+1}$ is found

$\left(F_{n+1} - \underline{K} d_{n+1}\right) / [M_{ii}]$ explicit in nature
can get $\underline{v}_{n+1}$

$_{n+1}$ } Newmark formulas i.e. $\tilde{\underline{d}}_{n+1} = \underline{d}_n + \Delta t\,\underline{v}_n + (\frac{1-2\beta}{2})\Delta t^2 \underline{a}_n$

$_{n+1}$ } $\tilde{\underline{v}}_{n+1} = \underline{v}_n + (1-\gamma)\Delta t\,\underline{a}_n$

are predictors
eliminate we can get

$\tilde{d}_{n+1}$ = known.
diag $\underline{K}$ hence we have a
$(\underline{M} + \beta\Delta t^2 \underline{K})$ is implicit

relationship for $\underline{v}_{n+1}$

$\underline{C}\,\tilde{\underline{v}}_{n+1} = \underline{K}\,\tilde{\underline{d}}_{n+1}$

if $\underline{M}$ is diag
we an explicit syste.

5/20/81

$\underline{d}_{n+1} = \underline{F}_{n+1}$
$\underline{v}_n + \frac{\Delta t}{2}\{(1-2\beta)\underline{a}_n + 2\beta\underline{a}_{n+1}$
$\gamma)\,\underline{a}_n + \gamma\underline{a}_{n+1}$
$\underline{d}_0$ given

ster flow incomp

We want to reduce this to an SDOF system. Recall EV problem.

Undamped EV problem $(\underset{\sim}{K} - \lambda \underset{\sim}{M})\underset{\sim}{\psi} = 0 \Rightarrow \{\lambda_l, \underset{\sim}{\psi}_l\}$ $1 \le l \le neq$

also $\underset{\sim}{\psi}_l^T \underset{\sim}{M} \underset{\sim}{\psi}_m = \delta_{lm}$ and $\underset{\sim}{\psi}_l^T \underset{\sim}{K} \underset{\sim}{\psi}_m = \lambda_l \delta_{lm}$ (no sum) where $\underset{\sim}{d} = \sum d_i \underset{\sim}{\psi}_i$

Look at the Rayleigh matrix damping ie $\boxed{\underset{\sim}{C} = a\underset{\sim}{M} + b\underset{\sim}{K}}$

then $\underset{\sim}{\psi}_l^T \underset{\sim}{C} \underset{\sim}{\psi}_m = a\delta_{lm} + b\lambda_l \delta_{lm}$ λ is an EV if $(\underset{\sim}{K} - \lambda \underset{\sim}{M})\underset{\sim}{\psi} = 0$

we will define $\boxed{\omega_l = (\lambda_l)^{1/2}}$ if we remember $\underset{\sim}{M}$ & $\underset{\sim}{K}$ are posit. semidef & def respect.
$= l$th undamped frequency ≥ 0

define $\boxed{\xi_l = \left(\frac{a}{\omega_l} + b\omega_l\right)\frac{1}{2}}$ lth modal damping factor.

now $\underset{\sim}{C}$ is diag as well as $\underset{\sim}{M}$ & $\underset{\sim}{K}$
If we do modal decomp (as in heat eqn.) on the original problem

(3)
$$\begin{cases} \ddot{d} + 2\xi\omega\dot{d} + \omega^2 d = F \\ d(0) = d_0 \\ \dot{d}(0) = \dot{d}_0 \end{cases}$$

these are the typical Fourier coeff

If we apply modal decomp on the temporally discretized version, we get

(4) $\underset{\sim}{a}_{n+1} + 2\xi\omega \underset{\sim}{v}_{n+1} + \omega^2 d_{n+1} = F_{n+1}$
same eqn for d_{n+1}, v_{n+1} w/ d_0 given & $v_0 = \dot{d}_0$ given

By taking (3) and temporally discretize we will get (4)

if we look at d_n and $d_n \to d(t_n)$ (t_n fixed) $\Rightarrow$ $\underset{\sim}{d}_n \to \underset{\sim}{d}(t_n)$
$v_n \to \dot{d}(t_n)$ $\Rightarrow$ $\underset{\sim}{v}_n \to \dot{\underset{\sim}{d}}(t_n)$

Now let's generalize the arguments from heat conduction by analyzing the algorithm. Now look at (4) in first order form.

$$\underset{\sim}{Y}_{n+1} = \underset{\sim}{A}\,\underset{\sim}{Y}_n + \underset{\sim}{L}_n$$

$\underset{\sim}{Y}_n = \begin{Bmatrix} d_n \\ v_n \end{Bmatrix}$ $\underset{\sim}{A}$ amplif matrix $\underset{\sim}{L}_n$ load vector

eqns for d_{n+1} & v_{n+1} and subst for a_{n+1} from $\underset{\sim}{a}_{n+1} + \cdots$ & for $\underset{\sim}{a}_n$ write eqn for previous time step. This leaves eqns in d_n, d_{n+1}, v_n, v_{n+1}. This

leads to $A = \underset{\sim}{A}_1^{-1} \underset{\sim}{A}_2$

$$\underset{\sim}{A}_1 = \begin{bmatrix} 1+\Delta t^2 \beta \omega^2 & 2\Delta t^2 \beta \xi \omega \\ \Delta t\, \gamma \omega^2 & 1 + 2\Delta t \gamma \xi \omega \end{bmatrix}$$

$$\underset{\sim}{A}_2 = \begin{bmatrix} 1 - \frac{\Delta t^2}{2}(1-2\beta)\omega^2 & \Delta t\left(1-\Delta t(1-2\beta)\xi\omega\right) \\ -\Delta t(1-\gamma)\omega^2 & 1 - 2\Delta t(1-\gamma)\xi\omega \end{bmatrix}$$

$$\underset{\sim}{L}_n = \underset{\sim}{A}_1^{-1} \left\{ \begin{matrix} \frac{\Delta t^2}{2}\left[(1-2\beta)F_n + 2\beta F_{n+1}\right] \\ \Delta t\left[(1-\gamma)F_n + \gamma F_{n+1}\right] \end{matrix} \right\}$$

We now do as before: define $\underset{\sim}{\tau}$ [truncation error vector - local] by replacing $\underset{\sim}{y}_{n+1}$ by $\underset{\sim}{y}(t_{n+1})$

$\therefore \quad \underset{\sim}{y}(t_{n+1}) = A\,\underset{\sim}{y}(t_n) + \underset{\sim}{L}_{n+1} + \underset{\sim}{\tau}(t_n)$ where $\underset{\sim}{y}(t_n) = \left\{ \begin{matrix} d(t_n) \\ \dot{d}(t_n) \end{matrix} \right\}$ etc

for newmark we can show that

$$\|\underset{\sim}{\tau}\| = O\left(\Delta t^{k+1}\right) \qquad \begin{matrix} k=2 & \text{when } \gamma = \frac{1}{2} \\ k=1 & \text{otherwise} \end{matrix}$$

ie doesn't depend on β through 2nd order.

aside $\gamma = \frac{1}{2}$ $\beta = \frac{1}{12}$ $\underset{\sim}{C} = 0 \Rightarrow k = 3$ the fox-goodwin method
Implicit and conditionally stable

We pick β to get unconditional stability or to get explicit type algorithm
($\beta = \frac{1}{4}$ implicit) ($\beta = 0$ cndit stable)

Now subtract $d(t_n)$ from d_n then

$$\underset{\sim}{y}_n - \underset{\sim}{y}(t_n) = \underset{\sim}{e}(t_n) = \underset{\sim}{A}^n \underset{\sim}{e}(0) - \sum_{i=0}^{n-1} \underset{\sim}{A}^i \underset{\sim}{\tau}(t_{n-1-i})$$

we want the matrix norm ≤ 1 for convergence.

Must now define consistent matrix norm.

Consider $\underset{\sim}{y} = \left\{ \begin{matrix} y_1 \\ \vdots \\ y_m \end{matrix} \right\} \in \mathbb{R}^m$ examples of vector norm (Euclidean) $(\underset{\sim}{y}^T \underset{\sim}{y})^{1/2} = \|\underset{\sim}{y}\|_2 \in \mathbb{R}$

matrix norm for $\underset{\sim}{A} \in \mathbb{R}^{m \times m}$

$$\|\underset{\sim}{A}\| \in \mathbb{R}^{m \times m} \to \mathbb{R} \quad \text{s.t.} \quad \text{(i)} \ \|\underset{\sim}{A}\| \geq 0 \quad \|\underset{\sim}{A}\| = 0 \text{ iff } \underset{\sim}{A} = \underset{\sim}{0}$$

(ii) $\|\underset{\sim}{A}\alpha\| = |\alpha| \, \|\underset{\sim}{A}\|$

(iii) $\|\underset{\sim}{A} + \underset{\sim}{B}\| \leq \|\underset{\sim}{A}\| + \|\underset{\sim}{B}\|$

(iv) $\|\underset{\sim}{A}\,\underset{\sim}{B}\| \leq \|\underset{\sim}{A}\| \, \|\underset{\sim}{B}\|$

So far matrix norm & vector norm are defined seperately. Want now to define vector & matrix norms that are compatible, ie $\|\underset{\sim}{A}\,\underset{\sim}{y}\| \leq \|\underset{\sim}{A}\| \, \|\underset{\sim}{y}\|$ $\Leftarrow$ compatibility

We now ask if $\exists$ compatible matrix & vector norm.
yes $\exists$ matrix that can be naturally defined.

example $\|\underset{\sim}{y}\|_2 = (\underset{\sim}{y}^T\underset{\sim}{y})^{1/2}$ $\quad$ $\|\underset{\sim}{A}\|_2 = (\text{max eigenvalue of } \underset{\sim}{A}^T\underset{\sim}{A})^{\frac{1}{2}}$

Convergence in one norm $\Rightarrow$ convergence in any other norm. We will henceforth use the $\|\ \|_2$ norms.

Now look at $\underset{\sim}{A}$ & look at its EV $\{\lambda^i\}_{i=1}^m$ $\quad$ $\underset{\sim}{A}$ need not be sym. in general.

$\therefore$ its EV are complex thus $\bar{\lambda}^i = Re(\lambda^i) - i\,Im(\lambda^i)$
now $|\lambda^i| = (\lambda^i\bar{\lambda}^i)^{1/2}$ & pick $\max_{i=1,\dots m} |\lambda^i| = \rho(\underset{\sim}{A})$ this is the spectral rad. of $\underset{\sim}{A}$

stability is defined by what no. $\rho(\underset{\sim}{A})$

Consider any natural matrix norm : $\rho(\underset{\sim}{A}) \le \|\underset{\sim}{A}\|$
In linear systems $\rho(\underset{\sim}{A})$ is enough to define stability but for non lin systems need also norm.

example Ex: $\begin{bmatrix} 0 & k \\ 0 & 0 \end{bmatrix}$ $\quad k >> 0$ $\quad \rho(\underset{\sim}{A}) = 0$ $\quad \|\underset{\sim}{A}\| = k$
$A^TA = \begin{bmatrix} 0 & k \\ k & 0 \end{bmatrix}$ ev solve $\lambda^2 - k^2 = 0$ $\therefore \|A\|_2 = k$
suppose $\underset{\sim}{A}$ has linearly indep eigenvectors.

$$\underset{\sim}{A} = \underset{\sim}{P}\,\underset{\sim}{\Lambda}\,\underset{\sim}{P}^{-1}$$
$\underset{\sim}{P}$ matrix of ev $\quad \underset{\sim}{\Lambda} = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_m \end{bmatrix}$ matrix of ev. $\quad \lambda_m = \max_i \lambda_i$

$$\underset{\sim}{A}^n = \underset{\sim}{P}\,\underset{\sim}{\Lambda}^n\,\underset{\sim}{P}^{-1} \quad \text{if } \lambda_m < 1 \Rightarrow \lambda_m^m \to 0 \text{ & } \underset{\sim}{A}^n \to \underset{\sim}{0}$$

5/21/81

from last time
$\|\underset{\sim}{A}\|_{(2)} = (\text{max ev of } \underset{\sim}{A}^T\underset{\sim}{A})^{1/2}$
$\|\underset{\sim}{y}\|_{(2)} = (\underset{\sim}{y}\cdot\underset{\sim}{y})^{1/2}$ corresponding vector no.
$\{\lambda^i\}_{i=1}^{neq}$ = EV of $\underset{\sim}{A}$; $\rho(\underset{\sim}{A}) = \max_i |\lambda_i| \le \|\underset{\sim}{A}\|$

Define: the Jordan Block $\underset{\sim}{J}$ is a square matrix

$$\underset{\sim}{J} = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & \ddots \\ & & 1 \\ 0 & & \lambda \end{bmatrix} \quad \lambda \text{ complex}$$

Th. $\underset{\sim}{A}$ is given. $\exists$ a matrix $\underset{\sim}{Q}$ s.t. $\underset{\sim}{Q}^{-1}\underset{\sim}{A}\underset{\sim}{Q} = \begin{bmatrix} \underset{\sim}{J}_1 & & \\ & \ddots & \\ & & \underset{\sim}{J}_k \end{bmatrix}$

where $\underset{\sim}{J}_i$ are jordan blocks

$$= \begin{bmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda_i \end{bmatrix}$$

This is a jordan form.

Thm: If $\underset{\sim}{A}$ has L.I. $\vec{ev}$ then for $\underset{\sim}{P} = [\vec{ev}_1, \vec{ev}_2 \dots] \Rightarrow \underset{\sim}{P}^{-1}\underset{\sim}{A}\underset{\sim}{P} = \Lambda$

where $\Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_m \end{bmatrix}$

We're doing this work to provide the groundwork to show that $\underset{\sim}{M}\ddot{d} + \underset{\sim}{K}d = F$ is convergent. We do this to find out how $\underset{\sim}{A}^n$ grows.

We will now look at the (spectral) stability —
we want $\|\underset{\sim}{A}^n\| \leq$ const $\forall n$ since error contains powers of A

To achieve this we will require spectral stability of $\underset{\sim}{A}$
ie (i) $\rho(\underset{\sim}{A}) \leq 1$
(ii) λ_i of multiplicity > 1 satisfy that $|\lambda_i| < 1$

look at $\underset{\sim}{A}$ for 2x2 pertaining to the Newmark Case.
For newmark case $\underset{\sim}{A}^n = (\underset{\sim}{P}\underset{\sim}{\Lambda}\underset{\sim}{P}^{-1})^n = \underset{\sim}{P}\underset{\sim}{\Lambda}^n\underset{\sim}{P}^{-1}$

then $\|\underset{\sim}{A}^n\| \leq \|\Lambda^n\|\,\|\underset{\sim}{P}\|\,\|\underset{\sim}{P}^{-1}\| =$ const $\|\Lambda^n\|$

for 2x2 $\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}^n = \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix}$; if $|\lambda_i| \leq 1 \Rightarrow |\lambda_1^n|, |\lambda_2^n| \leq 1$

now $\rho(\underset{\sim}{A}^n) = |\lambda_2^n| \leq 1$ but $\lambda_1 \neq \lambda_2$ ∴ 2nd condition is inoperative

for the most general canonical form.

$$\underset{\sim}{A}^n = (\underset{\sim}{Q}\underset{\sim}{J}\underset{\sim}{Q}^{-1})^n = (\underset{\sim}{Q}\underset{\sim}{J}^n\underset{\sim}{Q}^{-1})$$

$$\|\underset{\sim}{A}^n\| \leq \|\underset{\sim}{Q}\|\,\|\underset{\sim}{J}^n\|\,\|\underset{\sim}{Q}^{-1}\| = \text{const} \cdot \|\underset{\sim}{J}^n\|$$

suppose $\underset{\sim}{J} = \begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix}$ $\underset{\sim}{J}^n = \begin{bmatrix} \lambda^n & (n-1)\lambda^{n-1} \\ 0 & \lambda^n \end{bmatrix}$ grows w/ n ∴ if $|\lambda| = 1$ then

$\rho(\underset{\sim}{J}^n) = \lambda^n$ and $\|A\| \Rightarrow |(n-1)\lambda^{n-1}| < 1$

then if $|\lambda| = 1$ $\|A\|$ blows up

∴ need (ii) condition

the 2 conditions $\rho(A) \le 1$ & multiplicity are sufficient but not necessary conditions for stability. ex $[I] = A$ $I^n = I$ & is still stable $|\lambda| = 1$

Th Now suppose we fix t_n; assume $\underset{\sim}{A}$ is spectrally stable.

assume $\|\underset{\sim}{T}(t)\|_{(2)} \le c\,\Delta t^{k+1} \quad \forall t \in [0, T]$

then $\underset{\sim}{e}(t_n) = \underset{\sim}{d}_n - \underset{\sim}{d}(t_n) \to 0$ as $\Delta t \to 0$

Pf: we found $\underset{\sim}{e}(t_n) = \underset{\sim}{A}^n \underset{\sim}{e}(0) - \sum_{i=0}^{n-1} \underset{\sim}{A}^i \underset{\sim}{T}(t_{n-1-i})$

if $\underset{\sim}{e}(0)$ is given then

$$\|\underset{\sim}{e}(t_n)\| \le \sum_{i=0}^{n-1} \|\underset{\sim}{A}^i\| \, \|\underset{\sim}{T}(t_{n-1-i})\| \le n \cdot \rho(\underset{\sim}{A}) \, \|\underset{\sim}{T}\| \le n \, \overbrace{c\,\rho(A)}^{\text{const.}} \, \Delta t^{k+1}$$

$$\le \text{const}\; n\,\Delta t^{k+1} = \text{const}\; t_n \Delta t^k \quad \therefore \; \|\underset{\sim}{e}(t_n)\| \text{ is } O(k)$$

Spectral stability of newmark family $\underset{\sim}{A}$ being 2×2

look at EV. $0 = \det(\underset{\sim}{A} - \lambda \underset{\sim}{I}) = \lambda^2 - 2A_1\lambda + A_2$

$A_1 = \frac{1}{2}\,\text{tr}\,\underset{\sim}{A} = \frac{1}{2}(a_{ii})$

$A_2 = \det \underset{\sim}{A}$

$\therefore \quad \lambda_\pm = A_1 \pm \sqrt{A_1^2 - A_2}$

if $A_1^2 < A_2$ λ is complex

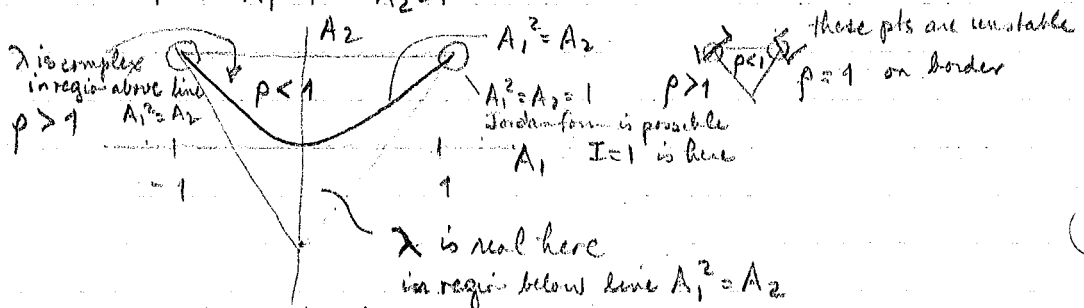
if $A_1^2 = A_2$ multiple root

if $A_1^2 > A_2$ λ is distinct real roots

for spectral stability are equivalent to $-(A_2+1)/2 \le A_1 \le (A_2+1)/2 \quad -1 \le A_2 \le 1$

& $-1 < A_1 < 1 \quad A_2 = 1$

exam question



if we define $\Omega = \omega \Delta t$ = "sampling frequency"

$$A_1 = 1 - [\xi\Omega + \Omega^2(\gamma + \tfrac{1}{2})/2]/D$$

$$A_2 = 1 - [2\xi\Omega + \Omega^2(\gamma - \tfrac{1}{2})/2]/D$$

$$D = 1 + 2\gamma\xi\Omega + \beta\Omega^2$$

Summary of Spectral stability for newmark

unconditionally stable $\frac{1}{2} \le \gamma \le 2\beta$ w/ $\underline{K}$ has positive ev

$\underline{\underline{A}} = a\underline{\underline{M}} + b\underline{\underline{K}}$ assume $a, b \ge 0$

$\xi \hat{A} \ge 0$ syste is physically stable

conditional stability $\gamma \ge \frac{1}{2}$ & $(2\beta < \gamma)$ $\gamma < \frac{1}{2}$ numerical amplifier

$$\Rightarrow \Omega = \omega \Delta t \le \Omega_{crit} = \frac{\xi(\gamma - \frac{1}{2}) + \left[\frac{\gamma}{2} - \beta + \xi^2(\gamma - \frac{1}{2})^2\right]^{\frac{1}{2}}}{\frac{\gamma}{2} - \beta}$$

we assume that $\beta \ge 0$

your max frequency ω determines Δt. If we define in terms of period of vibration ie $\omega T = 2\pi$ T if fund time then

$$\Delta t \cdot \frac{\omega}{2\pi} = \frac{1}{T} \cdot \Delta t \Longrightarrow \frac{\Delta t}{T} \le \frac{\Omega_{crit}}{2\pi}$$

Consider $\xi = 0$ physically undamped case. then $\Omega_{crit} = (\frac{\gamma}{2} - \beta)^{-\frac{1}{2}}$

for $\xi > 0$ $\gamma \ge \frac{1}{2}$ we increase stable time step since $\xi(\gamma - \frac{1}{2}) > 0$

Look at some known methods $\xi = 0$

	Method	β	γ	stabil. cond.	
accuracy determines Δt	(1) trap rule (average accel)	$\frac{1}{4}$	$\frac{1}{2}$	unconditionally	$2\beta \ge \gamma \ge \frac{1}{2}$ $\frac{1}{2} \ge \frac{1}{2} \ge \frac{1}{2}$ yes	
Stability determines Δt	(2) lin accel	$\frac{1}{6}$	$\frac{1}{2}$	$\Omega_{crit} = 2\sqrt{3} \cong 3.464$		IMPLICIT
	(3) Fox Goodwin	$\frac{1}{12}$	$\frac{1}{2}$	$\Omega_{crit} = \sqrt{6} \cong 2.449$		
	(4) Central Diff	0	$\frac{1}{2}$	$\Omega_{crit} = 2$		EXPLICIT

between implicit methods use (1). Between condit stable syste use (4) since explicit & you don't get much better Δt by using (2), (3).

for linear syste use (1). For nonlinear & wave prop type problems use (4).

5/26/81

Office Hr. Today only 3-4

Stability conditions: Newmark

requires implicit — Unconditional: $2\beta \geq \gamma \geq 1/2$

Conditional: $\gamma \geq 1/2$, $0 \leq \rho < \gamma/2$; $\omega\Delta t \leq \frac{2\pi}{T}\Delta t$

$$\Omega \leq \Omega_{crit} = \frac{\xi(\gamma-1/2) + \sqrt{[\gamma/2-\beta] + \xi^2(\gamma-1/2)^2}}{[\gamma/2-\beta]}$$

can be related to dissipative

conservative bound $\Omega \leq (\gamma/2-\beta)^{-1/2}$ undamped; $\xi > 0$ damped case

($\xi = 0$)

"Oscillatory Response" : $A_1^2 < A_2$ involves complex conjugate.

the above is true $\forall \omega$: we are governed by highest λ $\therefore (\Delta t)_{\lambda_{neq}} \leq (\Delta t)_{\lambda_{neq-1}} \ldots$

$\therefore (\Delta t)_{\lambda_{neq}}$ is conservative wrt whole syste.

Stronger (Slightly) notion of stability $\gamma \geq 1/2$ $\xi < 1$ underdamped.

unconditional requires $\beta \geq (\gamma+1/2)^2/4$

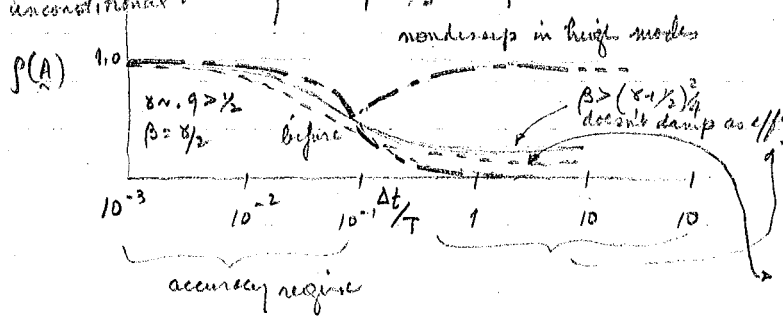
conditional $\beta < (\gamma+1/2)^2/4$ $\Omega < \Omega_{bifurcation} = \frac{\xi(\gamma-1/2)/2 + \sqrt{(\gamma+1/2)^2/4 - \beta + \xi^2(\beta-\gamma/2)}}{[(\gamma+1/2)^2 - \beta]}$

bifurcation ie pt at which complex conj roots become real

for $\xi = 0$ $\Omega_{bif} = [(\gamma+1/2)^2 - \beta]^{-1/2}$

$[\rho(\underset{\sim}{A})]^n$ ~ amplif "factor" of the mode in question. If $\rho(\underset{\sim}{A}) < 1$ then we get dissipation. For example

this is for unconditional stability with $\beta = \gamma/2$ being the lower limit



in accuracy we will take 10 is more steps/per

high modes - artifacts of spatial discretz

we're interested in damping syste.

so...

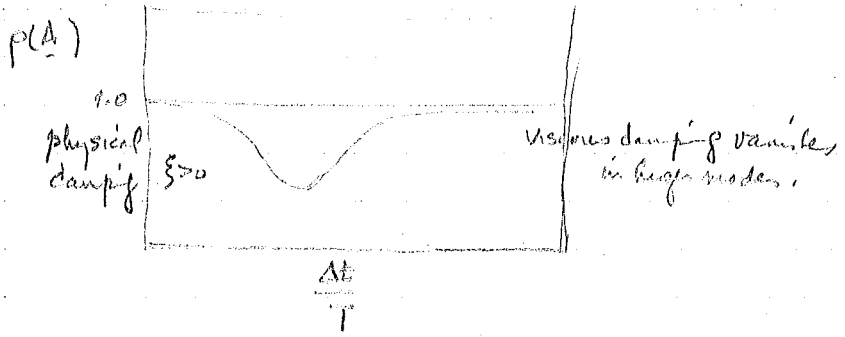
if we take $\beta = (\gamma+1/2)^2/4$ no bifurc and get dissip

if $\beta > \gamma/2$ & $(\gamma+1/2)^2 > \beta$ then we get bifurc but don't return back to 1.

Asymptote is ≥ 1. We will get damped oscillatory response.

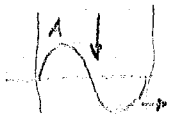
Hughes has a $\rho(\underline{A}) = 1 \quad \forall \omega, \Delta t$ $\quad \gamma = \frac{1}{2} \quad \beta = \frac{1}{4}$
Newmark tends to damp lower modes (which isn't good). Other methods tend to do better in lower modes.

Suppose we fix $\xi \neq 0$ (VISCOUS DAMPING)



What we want is accuracy in low modes, dissip in high modes

Accuracy Measures
2 types of phenomena for sinusoidal response $T \uparrow$ = amplitude $\downarrow$ dispersive (dissipative)



Look at $\ddot{d} + 2\xi\omega\dot{d} + \omega^2 d = 0$ + IC S.D.O.F. model prob (*)

Suppose $0 \leq \xi < 1$ underdamped oscillatory $\quad d(t) = e^{-\xi\omega t}\left(d_0 \cos\omega_d t + \left(\frac{\dot{d}_0 + \xi\omega d_0}{\omega_d}\right)\sin\omega_d t\right)$

$\omega_d = (1-\xi^2)^{1/2}\omega$

case (ii) Critically damped $\xi = 1$
(iii) overdamped $\xi > 1$
Look at

Continuous System	discrete analog
undamped	Complex conj λ's
Critic	Real & equal
Overdamped	Real & distinct

discrete algorithm (ie newmark) mimics (*)

$$d_{n+1} = A_{11} d_n + A_{12} V_n \quad \text{for no external forces}$$
$$V_{n+1} = A_{21} d_n + A_{22} V_n$$

at step n
$$d_n = A_{11} d_{n-1} + A_{12} V_{n-1}$$
$$V_n = A_{21} d_{n-1} + A_{22} V_{n-1}$$

want to find displ difference equation form

$$d_{n+1} - 2A_1 d_n + A_2 d_{n-1} = 0$$

A_1 = ½ trace $\hat{A}$; A_2 = det $\hat{A}$

$\lambda_{1,2}$ eigenvalues of $\hat{A}$ $\quad \lambda_1 \neq \lambda_2$

soln $d_n = C_1 \lambda_1^n + C_2 \lambda_2^n$ (const)

want to write this in terms of (#) assuming complex conjugate λ

Besides write exact soln. $\underset{\sim}{y}(t_{n+1}) = \underset{\sim}{E}\, \underset{\sim}{y}(t_n)$ $\quad \underset{\sim}{E}$ is the evolution operator & is the exact amplif matrix

looking at EV of $\underset{\sim}{E}$ then $\underset{\sim}{\lambda}(\underset{\sim}{E}) = e^{-\xi\Omega \pm i\Omega_d}$ $\quad \Omega, \Omega_d = \omega\Delta t, \omega_d \Delta t$

thus $d(t_n) = C_1 \lambda_1(\underset{\sim}{E})^n + C_2 \lambda_2(\underset{\sim}{E})^n$ complete parallel of discrete s.c.

since we have $\underset{\sim}{A}$ then we can now assume $\lambda = e^{-\bar{\xi}\bar{\Omega} \pm i\bar{\Omega}_d}$ & solve for $\bar{\xi}, \bar{\omega}$

using the following

for $\xi < 1$ $\quad \bar{\Omega}_d = \arctan\left(\frac{A_2}{A_1^2} - 1\right)^{1/2}$ $\quad \bar{\xi} = -\frac{1}{2\bar{\Omega}} \ln A_2$ $\quad \bar{\Omega}_d = (1-\xi^2)^{1/2}\bar{\Omega}$ (ξ = actual value of ξ)

Thus $d_n = e^{-\bar{\xi}\bar{\omega} t_n}\left(d_0 \cos \bar{\omega}_d t_n + \bar{C} \sin \bar{\omega}_d t_n\right)$ (diss, disp) from which a plot can get a picture of what mode is actually doing at each time step.

if set $d_1, d_0 \Rightarrow \bar{C} = \frac{d_1 - A_1 d_0}{(A_2 - A_1^2)^{1/2}} = \frac{1}{2}\frac{(A_{11} - A_{22})d_0 + A_{12} v_0}{(A_2 - A_1^2)^{1/2}}$ $\quad$ if $\xi = 0$ $\bar{\omega} = \omega_d$

Thus $\bar{\xi}, \bar{\omega}$ are the parameters to look at

people have looked at the following to characterize an algorithm 5/28/81

ie $\bar{\xi}$ algorithmic damping ratio

$\bar{\omega}$ " frequency, however people use $\frac{\bar{T} - T}{T}$ the relative period error

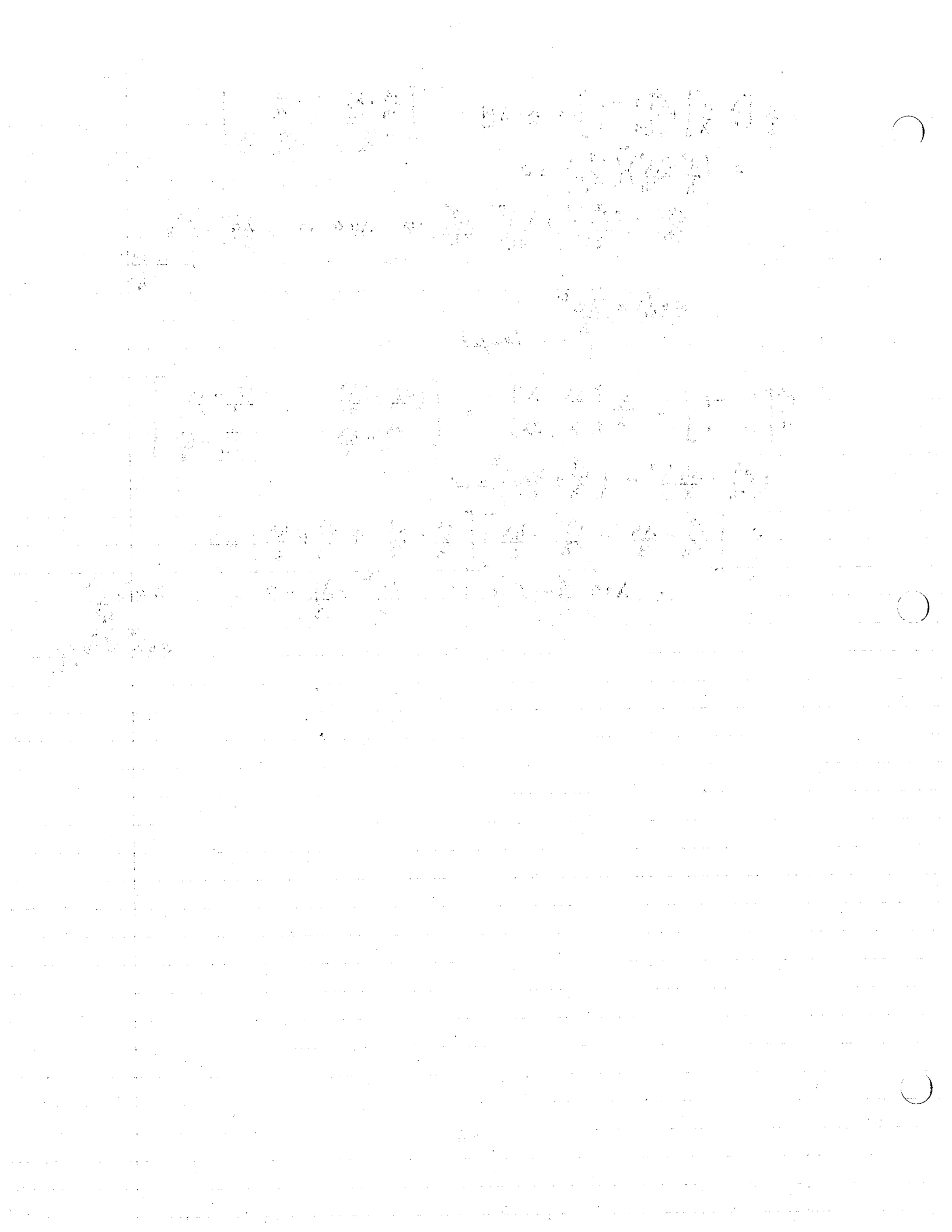
where $T\omega = 2\pi$ $\quad \bar{T}\bar{\omega} = 2\pi$

we mostly look at the cases where $\xi = 0$ even if algorithm introduces damping

for newmark damping is weakest feature of algorithm.

for newmark $\bar{\xi} = \xi + (\gamma - \tfrac{1}{2})\Omega_{/2} + O(\Omega^2)$ $\quad = \omega\Delta t = (\Delta t/T)2\pi$

thus we get a 1st order error in damping.



$$-\frac{h}{2}\begin{bmatrix}\lambda & 0\\ 0 & \lambda\end{bmatrix}+\frac{c^2}{h}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix} = \underline{K}-\lambda\underline{M} = \begin{bmatrix}c^2/h-\frac{\lambda h}{2} & -c^2/h\\ -c^2/h & c^2/h-\frac{\lambda h}{2}\end{bmatrix}$$

$$\therefore \left(\frac{c^2}{h}-\frac{\lambda h}{2h}\right)^2-\left(c^4/h^2\right)=0$$

$$c^4/h^2 - \frac{2c^2\lambda h^2}{2h^2}+\frac{\lambda^2h^4}{4h^2}-c^4/h^2 \Rightarrow \lambda=0 \text{ or } \frac{\lambda h^2}{4}-c^2=0$$

$$\lambda=\frac{4c^2}{h^2}$$

$$\omega=\sqrt{\lambda}=\frac{2c}{h}$$

Lumped

$$\frac{c^2}{h}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}-\frac{h}{6}\begin{bmatrix}2\lambda & \lambda\\ \lambda & 2\lambda\end{bmatrix} = \begin{bmatrix}(c^2/h-\frac{\lambda h}{3}) & -c^2/h-\frac{h\lambda}{6}\\ -c^2/h-\frac{h\lambda}{6} & c^2/h-\frac{\lambda h}{3}\end{bmatrix}$$

$$\left(c^2/h-\frac{\lambda h}{3}\right)^2-\left(c^2/h+\frac{h\lambda}{6}\right)^2=0$$

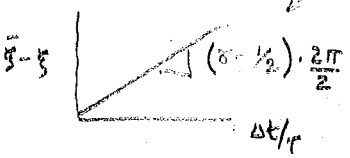
$$\therefore \left[c^2/h-\frac{\lambda h}{3}-c^2/h-\frac{h\lambda}{6}\right]\left[c^2/h-\frac{\lambda h}{3}+c^2/h+\frac{h\lambda}{6}\right]=0$$

$\therefore \lambda=0$ from here or $\frac{2c^2}{h}-\frac{\lambda h}{6}=0$ or $\lambda=12\frac{c^2}{h^2}$

$\omega=\lambda^{1/2}=2\sqrt{3}\,c/h$

(highest order)

If we plotted the graph we want to match modal truncation. However we don't get it.



but plotting $\frac{\bar{T}-T}{T}$ turns out the be $O(\Omega^2)$

Other things people have looked at has been amplitude decay. This is essentially equivalent to $\bar{\xi}$

Recall for conditionally stable algorithms.

* ie for heat conduction for forward difference
  $\alpha=0$ $\Delta t \le 2/\lambda_{max}$ in eqn on pg 16
* ie for elastodynamics for central difference
  $\Delta t \le 2/\omega_{max}$ $\xi=0$ $\beta=0$ $\gamma=1/2$ in eqn on pg 22

$$\lambda_{max} \le \max_e \lambda^e_{max} \text{(elemental } \lambda_{max}) \qquad \omega_{max} \le \max_e (\omega^e_{max})$$

We want to look at this in the lumped mass context; explicit schemes cond. stable.

Example (1.) LIN elem o——o elastic extensional for rod $u_{tt} = c^2 u_{xx}$ $c=\sqrt{E/\rho}$ bar wave velocity

Then $\underset{\sim}{k}^e = \frac{c^2}{h}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$ from 1st quarter

$\underset{\sim}{m}^e = \int_{\Omega^e} N_a N_b \, d\Omega = \frac{h}{6}\begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix}$ exact consistent

see Wright handout pg 8

sum of $\bar{a}_{ij}$ = mass $\frac{2+1+1+2}{6} = 1$

the Lumped mass matrix $\underset{\sim}{m}^e = \frac{h}{2}\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ Δ sum $\bar{a}_{ij} = 1/2 + 1/2 = 1$

$N_1 = \frac{1-\xi}{2}$ $N_2 = \frac{\xi+1}{2}$ then we get $\left(N_1^2\big|_{\xi=-1} N_1 + N_2 N_1^2\big|_{\xi=1}\right)\frac{h}{2} = \frac{h}{2}$ also same for N_2^2 ; $N_1 N_2\big|_{\xi=\pm 1} = 0$

Lumped $\underset{\sim}{m}^e$ can also be obtained using trapez rule ie $\xi_1=1$, $\xi_2=-1$ $N_1 = N_2 = 1$ on $\int N_a N_b \, d\Omega$

leads to $\underset{\sim}{m}^e = \delta_{ab} h/2$ $\frac{1}{2}\left[N_A N_B\big|_{\xi=-1} N_1 + N_A N_B\big|_{\xi=1} N_2\right] \cdot h$

we will get $\underset{\sim}{\omega}=0$ (freq of rigid mode) & $\omega_1 \ne 0 = \omega_{max}$ obtained from $\det(\underset{\sim}{K} - \lambda \underset{\sim}{M}) = 0$ see Wright notes pg

LUMPED $\lambda^{\text{LUMPED}} = \omega^e_{max} = 2c/h^e$ ✓ now $\Delta t \le \frac{2}{\omega^e_{max}} = \frac{h^e}{c}$ and the smallest value governs

Δt is the transit time for a bar wave to travel 1 elem

for the consistent case $\omega^e_{max} = 2\sqrt{3}\, c/h^e$ ✓ thus $\Delta t = \frac{h^e}{\sqrt{3}\, c} < \frac{h^e}{c}$

General rule Consistent case tend to produced highest frequencies ie shorter critical time steps than lumped ~~const~~

thus may require 2x no. of time steps

example 2  quadrature $\xi_1 = -1, \ \xi_2 = 0 \quad \xi_3 = 1$

1 4 1

ratio of masses produced by simpson's rule if using lumped masses $\tilde{m} = \frac{h}{6} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Exercise do the consistent case. $\int N_a N_b \, d\Omega$

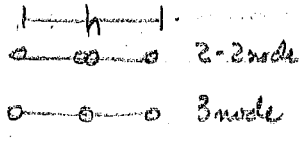
$N_1 = \frac{\xi(\xi-1)}{2} \quad N_2 = -(\xi^2-1) \quad N_3 = \frac{(\xi+1)\xi}{2}$

$d\Omega = j \, d\xi \quad j = x_{,\xi} = \partial/\partial\xi \left[\sum N_i x_i \right] = \left[\frac{2\xi-1}{2} x_1 - 2\xi x_2 + \frac{2\xi+1}{2} x_3 \right]$

if $x_1 = -h/2 \quad x_2 = 0 \quad x_3 = h/2 \quad j = h/2$

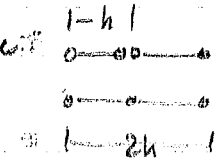
now $\omega_{max}^{lumped} = 2\sqrt{6}\, c/h \implies \Delta t \le \frac{2}{\omega} = \frac{h}{\sqrt{6}\, c} = .4082 \times \Delta t$ (of 2 node lumped)

this result is compared to 2-2node elements :



$\Delta t_{3node} = .4082 \, \Delta t_{2\text{-}2node}$

we can compare it also as follows



this leads to $\Delta t = .8164$ of Δt for 2-node

in both, cases Δt is more stringent for 3 node elements.

for the heat cond. solutions $\lambda_{max} = \omega_{max}^2$ replace c^2 in elasticity by k

thus $\Delta t \le 2/\lambda_{max} = \frac{h^2}{2k}$ for the 2 node lumped.

$= 2/\omega^2 = 2/(c^2/h^2)$

note that $\Delta t \sim O(h^2)$ this is a more stringent condition than the elasticity problem.

For 4 node quadrilateral 1 pt quadrature on stiffness (ie accepts linear fns exactly) hence $\Rightarrow$ trapezoidal rule on $\underset{\sim}{m}$ OK thus lumped mass)).

$\underset{\sim}{m}^{lumped} \Rightarrow$ 1/4 at each node.

$\omega_{max} \le c_d \, g^{1/2} \qquad c_d = (\lambda + 2\mu)^{1/2}$

g is a geometric parameter $= \frac{4}{A^2} \sum_{i=1}^{2} \sum_{a=1}^{4} b_{ia} \, b_{ia}$

area of elmt

$[b_{ia}] = \frac{1}{2} \begin{bmatrix} y_1 - y_4 & y_3 - y_1 & y_4 - y_2 & y_1 - y_3 \\ x_4 - x_2 & x_1 - x_3 & x_2 - x_4 & x_3 - x_1 \end{bmatrix}$

Now $\Delta t \le 2/(c_D \, g^{1/2})$ if we assume  h_2 then

h_1

$\le 1/\left[c_D \left(1/h_1^2 + 1/h_2^2 \right)^{1/2} \right]$

if $h_1 = h_2 = h$ then $\Delta t \leq \sqrt{2}\, h/c$ — however this is worse; yet this result is a sufficient condition

Linear multistep methods (LMS) methods

we will look at first order systems.

$$\underset{\sim}{\dot{y}} = \underset{\sim}{f}(\underset{\sim}{y}, t) = \underset{\sim}{\underset{\sim}{G}}\,\underset{\sim}{y} + \underset{\sim}{H}(t)$$

A k-step LMS method can be written as.

$$\sum_{i=0}^{k} \left\{ \alpha_i \underset{\sim}{y}_{n+1-i} + \Delta t\, \beta_i \underset{\sim}{f}\left(\underset{\sim}{y}_{n+1-i}, t_{n+1-i}\right) \right\} = \underset{\sim}{0}$$

α & β are param. that define algorithm.

LMS terminology

LMS method explicit case: $\beta_0 = 0$

implicit: if $\beta_0 \neq 0$.

backward diff: $\beta_i = 0 \quad i \geq 1$

[example generalized trapez family can be expressed in this notation

$k = 1 \quad \alpha_0 = -\alpha_1 = 1 \quad \beta_0 = -\alpha \quad \beta_1 = \alpha - 1$ where α is param. of gen. trapezoidal family.

for the heat eqn. if $\underset{\sim}{G} = -\underset{\sim}{M}^{-1}\underset{\sim}{K}$ & $\underset{\sim}{H}(t) = \underset{\sim}{M}^{-1}\underset{\sim}{F}(t)$

and $\underset{\sim}{y} = \underset{\sim}{d}$ we get the generalized trap. family.

6/2/81

Linear Multistep Methods for 1st Order systems. Continued

$$\underset{\sim}{\dot{y}} = \underset{\sim}{f}(\underset{\sim}{y}, t) = \underset{\sim}{\underset{\sim}{G}}\,\underset{\sim}{y} + \underset{\sim}{H}(t) \qquad (1)$$

for the semi discrete heat eqn.

$$\underset{\sim}{\underset{\sim}{M}}\,\underset{\sim}{\dot{d}} + \underset{\sim}{\underset{\sim}{K}}\,\underset{\sim}{d} = \underset{\sim}{F}$$

If we let $\underset{\sim}{\underset{\sim}{G}} = -\underset{\sim}{\underset{\sim}{M}}^{-1}\underset{\sim}{\underset{\sim}{K}}$ (4) and $\underset{\sim}{H} = \underset{\sim}{\underset{\sim}{M}}^{-1}\underset{\sim}{F}$ (5) then we get the above

$\underset{\sim}{y} = \underset{\sim}{d}$ (3)

$\underset{\sim}{\underset{\sim}{M}}$ is posetive def

$\underset{\sim}{\underset{\sim}{K}}$ " " semi def.

SEE ABOVE FOR REVIEW:

- if $\alpha = 0$ we get explicit method.
- $\alpha = 1$ we get a backward difference
- if $k \to \infty$ we get computational difficulties since we must keep more vectors (which costs money).

let us look at the spectrum of $\underset{\sim}{G}$

ie $(\underset{\sim}{G} - \lambda_g \underset{\sim}{I})\,\underset{\sim}{\psi}_g = 0$

subst. for G to get

$$(K + \lambda_g M)\,\underset{\sim}{\psi}_g = 0$$

Thus the $\lambda_g = -\lambda \leq 0$ $\quad \lambda$ } was soln for $(K - \lambda M)\,\underset{\sim}{\psi} = 0$

and $\underset{\sim}{\psi}_g = \underset{\sim}{\psi}$ $\quad \underset{\sim}{\psi}$ }

the λ_g fall on negative real axis.

Now we looke at the semi discrete eqns of motion.

General form was

$$\underset{\sim}{M}\,\ddot{\underset{\sim}{d}} + \underset{\sim}{C}\,\dot{\underset{\sim}{d}} + \underset{\sim}{K}\,\underset{\sim}{d} = \underset{\sim}{F} \qquad \underset{\sim}{M} > 0, \quad \underset{\sim}{C}, \underset{\sim}{K} \geq 0$$

Now let $\underset{\sim}{y} = \begin{pmatrix} \underset{\sim}{d} \\ \dot{\underset{\sim}{d}} \end{pmatrix}$

where $\underset{\sim}{G} = \begin{pmatrix} \underset{\sim}{0} & \underset{\sim}{I} \\ -\underset{\sim}{M}^{-1}\underset{\sim}{K} & -\underset{\sim}{M}^{-1}\underset{\sim}{C} \end{pmatrix} \qquad \underset{\sim}{H} = \begin{pmatrix} \underset{\sim}{0} \\ \underset{\sim}{M}^{-1}\underset{\sim}{F} \end{pmatrix}$

Then $\dot{\underset{\sim}{y}} = \underset{\sim}{G}\,\underset{\sim}{y} + \underset{\sim}{H}$

Now let $\underset{\sim}{\Psi} = \begin{pmatrix} \underset{\sim}{\psi}_1 \\ \underset{\sim}{\psi}_2 \end{pmatrix}$

the generalized EV

$$(\underset{\sim}{G} - \lambda \underset{\sim}{I})\,\underset{\sim}{\Psi} = \left(\begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}C \end{bmatrix} - \lambda \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \right) \begin{pmatrix} \underset{\sim}{\psi}_1 \\ \underset{\sim}{\psi}_2 \end{pmatrix} = \begin{pmatrix} \underset{\sim}{0} \\ \underset{\sim}{0} \end{pmatrix}$$

using: $\underset{\sim}{\psi}_2 - \lambda \underset{\sim}{\psi}_1 = 0 \qquad (1)$

$-\underset{\sim}{M}^{-1}\underset{\sim}{K}\,\underset{\sim}{\psi}_1 - (\underset{\sim}{M}^{-1}\underset{\sim}{C} + \lambda \underset{\sim}{I})\,\underset{\sim}{\psi}_2 = 0$ $\quad$ sub ibs for $\underset{\sim}{\psi}_2$ for (1)

$$\therefore \; -\underset{\sim}{K}\,\underset{\sim}{\psi}_1 - (\underset{\sim}{C} + \lambda \underset{\sim}{M})\,\lambda \underset{\sim}{\psi}_1 = 0$$

or $$(\lambda^2 \underset{\sim}{M} + \lambda \underset{\sim}{C} + \underset{\sim}{K})\,\underset{\sim}{\psi}_1 = 0$$

define now $\hat{\underset{\sim}{\psi}}_\ell^T \underset{\sim}{M}\, \hat{\underset{\sim}{\psi}}_m = \delta_{\ell m}$

and $\hat{\underset{\sim}{\psi}}_\ell^T \underset{\sim}{K}\, \hat{\underset{\sim}{\psi}}_m = \omega_\ell^2 \delta_{\ell m}$

if $\underset{\sim}{\Psi}$ diagonalized $\underset{\sim}{C}$ then we can find a semiple soln to the above

ie use rayleigh damping $\underset{\sim}{C} = a\underset{\sim}{M} + b\underset{\sim}{K}$

then $\underset{\sim}{\psi}_l^T \underset{\sim}{C} \underset{\sim}{\psi}_m = a \delta_{lm} + b \omega_l^2 \delta_{lm} = 2\xi_l \omega_l \delta_{lm}$

thus $\lambda_{gl}^2 + 2\xi_l \omega_l \lambda_{gl} + \omega_l^2 = 0 \qquad \lambda = \lambda_g$ ##

for a SDOF $\lambda_g^2 + 2\xi\omega\lambda_g + \omega^2 = 0$

system is

oscillatory if $0 \le \xi < 1$ underdamped then $\lambda_{1,2} = -\xi\omega \pm i\omega\sqrt{1-\xi^2}$ conjugate

$\xi = 1$ critically damp then $\lambda_{1,2} = -\xi\omega = -\omega$ (double root)

$\xi > 1$ overdamped $\lambda_{1,2} = -\xi\omega \pm \omega\sqrt{\xi^2 - 1}$ real

1. Remarks if $\xi = 0$ (no damping) $\lambda = \pm i\omega$
2. Since ξ & $\omega \ge 0$ then $\lambda_{1,2}$ are elements of the negative half plane of the complex plane.
3. Considering LMS methods stability region contains spectrum of $\underset{\sim}{G}$

Stability of LMS methods

- Assume $\underset{\sim}{S}$ possesses LI $\overrightarrow{ev}$ (assumes diagonalizability)

a) we will reduce to SDOF

let $\lambda = \lambda_g$ $\quad \underset{\sim}{P} = \underset{\sim}{P}_g = (\underset{\sim}{\psi}_1, \underset{\sim}{\psi}_2, \ldots \underset{\sim}{\psi}_n)$

$\underset{\sim}{\psi} = \underset{\sim}{\psi}_g$ & $\underset{\sim}{\Lambda} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{bmatrix}$

because of assumption $\underset{\sim}{P}^{-1} \underset{\sim}{G} \underset{\sim}{P} = \underset{\sim}{\Lambda}$

let $\underset{\sim}{z} = \underset{\sim}{P}^{-1} \underset{\sim}{y}$ we will consider $\underset{\sim}{H}(t) = 0$

then $\dot{\underset{\sim}{z}} = \underset{\sim}{\Lambda}\, \underset{\sim}{z} \Rightarrow \dot{z} = \lambda z$ because of diag $\Leftarrow \begin{cases} \underset{\sim}{P}^{-1}\dot{\underset{\sim}{y}} = \underset{\sim}{P}^{-1}\underset{\sim}{G}\underset{\sim}{P}\underset{\sim}{P}^{-1}\underset{\sim}{y} \\ \dot{\underset{\sim}{z}} = \underset{\sim}{\Lambda}\,\underset{\sim}{z} \end{cases}$

Put $\dot{y}$ into LMS k-step

$$\sum_{i=0}^{k} \left(\alpha_i \underset{\sim}{z}_{n+1-i} + \delta t\, \beta_i \underset{\sim}{\Lambda}\, \underset{\sim}{z}_{n+1-i} \right) = 0$$

for SDOF since diag & $\underset{\sim}{\psi}$ are LI

$$\sum_{i=0}^{k} \left(\alpha_i z_{n+1-i} + \Delta t\, \beta_i \lambda\, z_{n+1-i} \right) = 0$$

these are linear diff eqn.

the solution: $z_{n+1-i} = C\zeta^{n+1-i}$. Put into k-step to get

$$\sum_{i=0}^{k} (\alpha_i + \Delta t\, \beta_i \lambda) C\zeta^{n+1-i} = 0$$ now look at roots of this polynomial

for stability analysis

Definition 1: An LMS method of the form (2) (ie $\sum_{i=0}^{k}(\quad)=0$)

is said to be absolutely stable for a fixed $\Delta t\lambda$

if $\forall$ ξ satisfying eqn them $|\xi| \leq 1$. ie lie of unit circle.

(similar to spectral stability). There is an equiv between this & spectral stability of A.

Definition 2: Region of absolute stability

ie the set of all $\lambda\Delta t \in \mathbb{C}$ at which the LMS is absolutely stable.

Now we can take the k-step method & rewrite it as

$$\begin{Bmatrix} z_{n+1} \\ z_n \\ \vdots \\ z_{n+2-k} \end{Bmatrix} = \underset{k\times k}{\underset{\sim}{A}} \begin{Bmatrix} z_n \\ \vdots \\ z_{n+1-k} \end{Bmatrix}$$ where

$$\underset{\sim}{A} = \begin{bmatrix} -\frac{(\alpha_1+\Delta t\lambda\beta_1)}{(\alpha_0+\Delta t\lambda\beta_0)} & -\frac{(\alpha_2+\Delta t\lambda\beta_2)}{(\alpha_0+\Delta t\lambda\beta_0)} & \cdots & -\frac{(\alpha_k+\Delta t\lambda\beta_k)}{(\alpha_0+\Delta t\lambda\beta_0)} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & \vdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad k\times k$$

looking at $\rho(\underset{\sim}{A})$ its polynomial will be the same as the ξ polynomial we derived before.

Definition 3: A - Stable Algorithm

An LMS method is A-Stable if the solutions of (*) $\to 0$ as $n\to\infty$ when $\mathrm{Re}\,\lambda < 0$

(*) $\sum_{i=0}^{k}(\alpha_i+\Delta t\lambda\beta_i) z_{n+1-i} = 0$

Appendix II - Proof of (7.25)

A preliminary lemma.

Let $\underset{\sim}{A}$ and $\underset{\sim}{B}$ be symmetric matrices of dimension n, let $\underset{\sim}{B}$ be positive definite and consider the eigenvalue problem

$$\underset{\sim}{A}\underset{\sim}{u} - \lambda\underset{\sim}{B}\underset{\sim}{u} = \underset{\sim}{0}. \tag{II1}$$

There are n real eigenvalues

$$\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n, \tag{II2}$$

and corresponding orthonormal (with respect to $\underset{\sim}{B}$) eigenvectors $\underset{\sim}{u}_1, \ldots, \underset{\sim}{u}_n$. The Rayleigh quotient is defined by

$$R(\underset{\sim}{u}) = \frac{\underset{\sim}{v}^T\underset{\sim}{A}\underset{\sim}{v}}{\underset{\sim}{v}^T\underset{\sim}{B}\underset{\sim}{v}}, \quad \underset{\sim}{v} \neq \underset{\sim}{0}. \tag{II3}$$

Lemma

$$\lambda_n = \max_{\substack{\underset{\sim}{v}\in \mathbb{R}^n \\ \underset{\sim}{v} \neq \underset{\sim}{0}}} R(\underset{\sim}{v})$$

and the maximum occurs for $\underset{\sim}{v} = c\underset{\sim}{u}_n$, where c is an arbitrary non-zero constant.

Proof. Since the eigenvectors $\underset{\sim}{u}_1, \ldots, \underset{\sim}{u}_n$ form a basis for $\mathbb{R}^n$, we can expand any vector $\underset{\sim}{v}$ as

$$\underset{\sim}{v} = \sum_{i=1}^{n} c_i \underset{\sim}{u}_i ,$$

and, therefore,

$$\underset{\sim}{v}^T \underset{\sim}{B} \underset{\sim}{v} = \sum_{i=1}^{n} |c_i|^2$$

$$\underset{\sim}{v}^T \underset{\sim}{A} \underset{\sim}{v} = \sum_{i=1}^{n} \lambda_i |c_i|^2 .$$

For $\underset{\sim}{v} \neq \underset{\sim}{0}$,

$$R(\underset{\sim}{v}) = \frac{\sum_{i=1}^{n} \lambda_i |c_i|^2}{\sum_{i=1}^{n} |c_i|^2} \leq \lambda_n ,$$

now since $\lambda_n \geq \lambda_i \quad \forall i = 1,2,\ldots$

$\Leftarrow \quad \sum_{i=1}^{n} \lambda_n |c_i|^2 \geq \sum_{i=1}^{n} \lambda_i |c_i|^2$

by (II2). Since $R(c\underset{\sim}{u}_n) = \lambda_n$, the proof of the lemma is complete. ∎

Let $\underset{\sim}{K}$ and $\underset{\sim}{M}$ be global matrices formed by assembling the collections of matrices $\{\underset{\sim}{k}^e\}_1^{n_{el}}$ and $\{\underset{\sim}{m}^e\}_1^{n_{el}}$ according to some connectivity array.

Consider the symmetric, block-diagonal matrices

$$\tilde{\underset{\sim}{K}} = \begin{bmatrix} \underset{\sim}{k}^1 & \underset{\sim}{0} & \cdots & \underset{\sim}{0} \\ \underset{\sim}{0} & \underset{\sim}{k}^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underset{\sim}{0} \\ \underset{\sim}{0} & \cdots & \underset{\sim}{0} & \underset{\sim}{k}^{n_{el}} \end{bmatrix} \tag{II4}$$

$$\tilde{\underset{\sim}{M}} = \begin{bmatrix} \underset{\sim}{m}^1 & \underset{\sim}{0} & \cdots & \underset{\sim}{0} \\ \underset{\sim}{0} & \underset{\sim}{m}^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \underset{\sim}{0} \\ \underset{\sim}{0} & \cdots & \underset{\sim}{0} & \underset{\sim}{m}^{n_{el}} \end{bmatrix} \tag{II5}$$

The dimension of $\tilde{\underset{\sim}{K}}$ and $\tilde{\underset{\sim}{M}}$ is $\tilde{n}_{eq} = \sum_{e=1}^{n_{el}} n^e_{eq}$, where n^e_{eq} denotes the number of equations for element e. If n_{eq} is the dimension of $\underset{\sim}{K}$ and $\underset{\sim}{M}$ then, clearly, $\tilde{n}_{eq} \geq n_{eq}$.

The eigenvalue problems for the $\underset{\sim}{K}$, $\underset{\sim}{M}$ and $\tilde{\underset{\sim}{K}}$, $\tilde{\underset{\sim}{M}}$ systems are (resp.)

$$(\underset{\sim}{K} - \lambda \underset{\sim}{M})\underset{\sim}{u} = \underset{\sim}{0} \tag{II6}$$

and

$$(\tilde{\underset{\sim}{K}} - \lambda \tilde{\underset{\sim}{M}})\tilde{\underset{\sim}{u}} = \underset{\sim}{0}\,. \tag{II7}$$

All the matrices involved are assumed symmetric and $\underset{\sim}{M}$ and $\underset{\sim}{\tilde{M}}$ are assumed positive-definite. Thus both (II6) and (II7) fit into the framework of (II1) and, therefore, the lemma holds in each case, viz.

$$\lambda_{n_{eq}} = \max_{\substack{\underset{\sim}{v} \in \mathbb{R}^{n_{eq}} \\ \underset{\sim}{v} \neq \underset{\sim}{0}}} \frac{\underset{\sim}{v}^T \underset{\sim}{K} \underset{\sim}{v}}{\underset{\sim}{v}^T \underset{\sim}{M} \underset{\sim}{v}} \tag{II8}$$

$$\tilde{\lambda}_{\tilde{n}_{eq}} = \max_{\substack{\underset{\sim}{v} \in \mathbb{R}^{\tilde{n}_{eq}} \\ \underset{\sim}{v} \neq \underset{\sim}{0}}} \frac{\underset{\sim}{v}^T \underset{\sim}{\tilde{K}} \underset{\sim}{v}}{\underset{\sim}{v}^T \underset{\sim}{\tilde{M}} \underset{\sim}{v}} \tag{II9}$$

Let $\mathbb{R}^{n_{eq}}_{\text{constrained}} = \{\underset{\sim}{v} \in \mathbb{R}^{n_{eq}} \mid$ components of $\underset{\sim}{v}$ which correspond to the same degree-of-freedom in $\mathbb{R}^{n_{eq}}$ under the identification of the connectivity array used in assembling $\underset{\sim}{K}$ and $\underset{\sim}{M}$ are to be equal$\}$. With this definition it is clear that

$$\lambda_{n_{eq}} = \max_{\substack{\underset{\sim}{v} \in \mathbb{R}^{n_{eq}}_{\text{const.}} \\ \underset{\sim}{v} \neq \underset{\sim}{0}}} \frac{\underset{\sim}{v}^T \underset{\sim}{\tilde{K}} \underset{\sim}{v}}{\underset{\sim}{v}^T \underset{\sim}{\tilde{M}} \underset{\sim}{v}} \leq \tilde{\lambda}_{\tilde{n}_{eq}} . \tag{II10}$$

Remarks

1. The equality in (II10) may be intuitively obvious to some readers. To make it convincing to those readers who are dubious, consider the following simple example:

Let

$$\underset{\sim}{k}^e = \begin{bmatrix} k^e_{11} & k^e_{12} \\ k^e_{21} & k^e_{22} \end{bmatrix}$$

$$\underset{\sim}{m}^e = \begin{bmatrix} m^e_{11} & m^e_{12} \\ m^e_{21} & m^e_{22} \end{bmatrix}$$

$$e = 1, 2$$

$$\underset{\sim}{K} = \begin{bmatrix} k^1_{11} & k^1_{12} & 0 \\ k^1_{21} & (k^1_{22}+k^2_{11}) & k^2_{12} \\ 0 & k^2_{21} & k^2_{22} \end{bmatrix}$$

$$\underset{\sim}{M} = \begin{bmatrix} m^1_{11} & m^1_{12} & 0 \\ m^1_{21} & (m^1_{22}+m^2_{12}) & m^2_{12} \\ 0 & m^2_{21} & m^2_{22} \end{bmatrix}$$

$$\underbrace{\tilde{\tilde{K}}}_{4x4} = \begin{bmatrix} \tilde{k}^1 & \tilde{0} \\ \tilde{0} & \tilde{k}^2 \end{bmatrix}$$

$$\underbrace{\tilde{\tilde{M}}}_{4x4} = \begin{bmatrix} \tilde{m}^1 & \tilde{0} \\ \tilde{0} & \tilde{m}^2 \end{bmatrix}$$

$$\tilde{\varphi} = \begin{Bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{Bmatrix}$$

$$\tilde{\tilde{\varphi}} = \begin{Bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_2 \\ \varphi_3 \end{Bmatrix} \in \mathbb{R}^4_{\text{const.}}$$

The equality in (II10) is established for the example under consideration by the following calculations:

$$\tilde{\varphi}^T \tilde{\tilde{K}} \tilde{\varphi} = \begin{Bmatrix} \varphi_1 \\ \varphi_2 \end{Bmatrix}^T \tilde{k}^1 \begin{Bmatrix} \varphi_1 \\ \varphi_2 \end{Bmatrix} + \begin{Bmatrix} \varphi_2 \\ \varphi_3 \end{Bmatrix}^T \tilde{k}^2 \begin{Bmatrix} \varphi_2 \\ \varphi_3 \end{Bmatrix}$$

$$= \tilde{\varphi}^T K \tilde{\varphi}$$

$$\underset{\sim}{\varphi}^T \underset{\sim}{\tilde{M}} \underset{\sim}{\varphi} \quad \begin{Bmatrix} \varphi_1 \\ \varphi_2 \end{Bmatrix}^T \underset{\sim}{m}^1 \begin{Bmatrix} \varphi_1 \\ \varphi_2 \end{Bmatrix} + \begin{Bmatrix} \varphi_2 \\ \varphi_3 \end{Bmatrix}^T \underset{\sim}{m}^2 \begin{Bmatrix} \varphi_2 \\ \varphi_3 \end{Bmatrix}$$

$$= \underset{\sim}{\varphi}^T \underset{\sim}{M} \underset{\sim}{\varphi}$$

The essence of a formal proof may be deduced from the above calculations. The interested reader may wish to provide the details.

2. The inequality in (II10) follows directly from (II9).

In terms of the element matrices, the eigenvalue problem (II7) can be written as

$$\begin{bmatrix} \underset{\sim}{k}^1 - \tilde{\lambda}\underset{\sim}{m}^1 & \underset{\sim}{0} & \cdots & \underset{\sim}{0} \\ \underset{\sim}{0} & \underset{\sim}{k}^2 - \tilde{\lambda}\underset{\sim}{m}^2 & & \vdots \\ \vdots & & \ddots & \underset{\sim}{0} \\ \underset{\sim}{0} & \cdots & \underset{\sim}{0} & \underset{\sim}{k}^{n_{el}} - \tilde{\lambda}\underset{\sim}{m}^{n_{el}} \end{bmatrix} \begin{Bmatrix} \underset{\sim}{\tilde{u}}^1 \\ \underset{\sim}{\tilde{u}}^2 \\ \vdots \\ \underset{\sim}{\tilde{u}}^{n_{el}} \end{Bmatrix} = \begin{Bmatrix} \underset{\sim}{0} \\ \underset{\sim}{0} \\ \vdots \\ \underset{\sim}{0} \end{Bmatrix} \tag{II11}$$

From this expression it can be seen that the eigenvalues of (II7) are the same as the totality of eigenvalues of the uncoupled element problems

$$(\underset{\sim}{k}^e - \lambda^e \underset{\sim}{m}^e)\underset{\sim}{u}^e = \underset{\sim}{0}, \qquad e = 1, \ldots, n_{el} \tag{II12}$$

Therefore

$$\underset{\sim}{\lambda}_{n_{eq}} = \max_e (\lambda^e_{max}), \tag{II13}$$

where

$$\lambda^e_{max} = \max_{\substack{\underset{\sim}{v} \in \mathbb{R}^{n_{eq}} \\ \underset{\sim}{v} \neq 0}} \frac{\underset{\sim}{v}^T \underset{\sim}{k}^e \underset{\sim}{v}}{\underset{\sim}{v}^T \underset{\sim}{m}^e \underset{\sim}{v}} .$$

Combining (II10) and (II13), we have that

$$\lambda_{n_{eq}} \leq \max_e (\lambda^e_{max}),$$

which completes the proof of the theorem. ∎

FINITE ELEMENT NOTES

1) LINEAR MULTISTEP METHODS FOR FIRST-ORDER EQUATION SYSTEMS

CONSIDER A SYSTEM OF LINEAR, ORDINARY DIFFERENTIAL EQUATIONS

$$\dot{\underset{\sim}{y}} = \underset{\sim}{f}(\underset{\sim}{y}, t) = \underset{\sim}{G}\,\underset{\sim}{y} + \underset{\sim}{H}(t) \qquad (1)$$

WHERE

$\underset{\sim}{G}$ ... IS A GIVEN CONSTANT MATRIX

$\underset{\sim}{H}$ ... IS A GIVEN VECTOR-VALUED FUNCTION OF t

DEFINITION: A k-STEP LINEAR MULTISTEP METHOD FOR (1) IS DEFINED BY

$$\sum_{i=0}^{k} \{\alpha_i \underset{\sim}{y}_{n+1-i} + \Delta t\, \beta_i \underset{\sim}{f}(\underset{\sim}{y}_{n+1-i}, t_{n+1-i})\} = \underset{\sim}{0} \qquad (2)$$

WHERE THE α_i's AND β_i's ARE GIVEN PARAMETERS WHICH DEFINE THE METHOD

NOTE: THE TERM "LINEAR" IN LINEAR MULTISTEP METHOD REFERS TO THE FORM OF (2) NOT TO THE LINEARITY OF $\underset{\sim}{f}$ AS GIVEN ABOVE. IN FACT (2) HOLDS FOR $\underset{\sim}{f}$ NONLINEAR.

REMARKS:

i) AN LMS METHOD IS CALLED EXPLICIT IF $\beta_0 = 0$ OTHERWISE IT IS CALLED IMPLICIT.

ii) AN LMS METHOD IS CALLED A BACKWARD-DIFFERENCE METHOD IF $\beta_i = 0$ FOR ALL $i \geq 1$

$\alpha_0 \underset{\sim}{y}_{n+1} + \Delta t\, \beta_0 \underset{\sim}{f}(\underset{\sim}{y}_{n+1}, t_{n+1}) + \alpha_1 \underset{\sim}{y}_n + \alpha_2 \underset{\sim}{y}_{n-1} + \cdots + \alpha_k \underset{\sim}{y}_{n+1-k} = 0$ then $\underset{\sim}{y}_{n+1}$ is dependent only on all $\underset{\sim}{y}_i$ $i \leq n+1$

iii) AN EXAMPLE OF A 1-STEP LMS METHOD IS THE GENERALIZED TRAPEZOIDAL FAMILY, FOR WHICH

$k=1 \quad \alpha_0 = 1 \quad \beta_0 = -\alpha$

$\alpha_1 = -1 \quad \beta_1 = \alpha - 1$

$\Rightarrow y_{n+1} - \Delta t\, \alpha f(y_{n+1}, t_{n+1}) - y_n - \Delta t\,(1-\alpha) f(y_n, t_n) = 0$

RECALL $\alpha = 0 \Rightarrow$ EXPLICIT $\Rightarrow y_{n+1} = y_n + \Delta t\, f(y_n, t_n)$

$\alpha = 1 \Rightarrow$ BACKWARD DIFFERENCE $y_{n+1} - \Delta t\, f(y_{n+1}, t_{n+1}) = y_n$

$\alpha=0$: $\alpha_0=1$ $\beta_0=0$, $\alpha_1=-1$ $\beta_1=-1$

$\alpha=1$: $\alpha_0=1$ $\beta_0=-1$, $\alpha_1=-1$ $\beta_1=0$

iv) THE GREATER THE NUMBER OF STEPS INVOLVED ("k") THE GREATER THE "HISTORICAL DATA POOL" WHICH MUST BE STORED $\rightarrow$ A PRACTICAL DISADVANTAGE!

2) SEMI-DISCRETE HEAT EQUATION AND EQUATION OF MOTION IN FIRST-ORDER FORM

IN ORDER TO STUDY LMS METHODS AS APPLIED TO THE HEAT EQUATION AND THE EQUATION OF MOTION, WE NEED TO FIRST PUT THESE EQUATIONS IN THE FORM OF (1), AND THEN STUDY THE SPECTRAL PROPERTIES OF THE RESULTING G MATRICES.

a) SEMI-DISCRETE HEAT EQUATION

GIVEN:

$$M \dot{d} + K d = F$$
$$d(0) = D_0$$

$\rightarrow \quad \dot{y} = G y$

let $y = \psi e^{\lambda t}$ for homog case

$\dot{y} = \lambda \psi e^{\lambda t}$

$\Rightarrow \lambda \psi e^{\lambda t} = G \psi e^{\lambda t}$

$\Rightarrow (\lambda I - G)\psi e^{\lambda t} = 0$

$\Rightarrow$ ev problem to find λ s.t. $[\lambda I - G]$ [illegible]

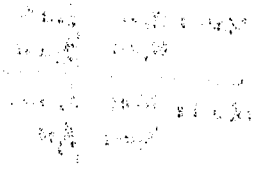
WHERE:

M ... CAPACITY MATRIX, SYMM. POS. DEF.

K ... CONDUCTIVITY MATRIX, SYMM. POS. SEMI-DEF.

$F = F(t)$, $t \in [0, T]$ ... HEAT SUPPLY VECTOR (PRESC.)

d ... TEMPERATURE VECTOR



$\dot{\underset{\sim}{d}}$ ... TIME (t) DERIVATIVE OF $\underset{\sim}{d}$

$\underset{\sim}{D_0}$ ... GIVEN INITIAL TEMPERATURE.

LETTING $\underset{\sim}{y} = \underset{\sim}{d}$, WE FIND (3)

$$\underset{\sim}{G} = -\underset{\sim}{M}^{-1}\underset{\sim}{K} \tag{4}$$

$$\underset{\sim}{H}(t) = \underset{\sim}{M}^{-1}\underset{\sim}{F}(t) \tag{5}$$

THE SPECTRUM OF $\underset{\sim}{G}$ IS DETERMINED FROM THE FOLLOWING EIGENPROBLEM

$$(\underset{\sim}{G} - \lambda(\underset{\sim}{G})\underset{\sim}{I})\,\underset{\sim}{\Psi}(\underset{\sim}{G}) = \underset{\sim}{0} \tag{6}$$

SUBSTITUTING (4) AND MULTIPLYING BY $\underset{\sim}{M}$ YIELDS

$$(\underset{\sim}{K} + \lambda(\underset{\sim}{G})\underset{\sim}{M})\,\underset{\sim}{\Psi}(\underset{\sim}{G}) = \underset{\sim}{0} \tag{7}$$

COMPARISON WITH PREVIOUS RESULTS (I6) YIELDS

$$\lambda(\underset{\sim}{G}) = -\lambda \leq 0 \tag{8}$$

$$\underset{\sim}{\Psi}(\underset{\sim}{G}) = \underset{\sim}{\Psi} \tag{9}$$

THUS IN THIS CASE <u>THE SPECTRUM OF $\underset{\sim}{G}$ ($\lambda(\underset{\sim}{G})$) RESIDES UPON THE NEGATIVE REAL AXIS IN THE COMPLEX PLANE, $\mathbb{C}$</u>.

b) <u>SEMI-DISCRETE EQUATIONS OF MOTION</u>

GIVEN:

$$\underset{\sim}{M}\ddot{\underset{\sim}{d}} + \underset{\sim}{C}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}$$

let $\dot{d} = v$ $d = u$

$$\underset{\sim}{M}\dot{\underset{\sim}{v}} + \underset{\sim}{C}\underset{\sim}{v} + \underset{\sim}{K}\underset{\sim}{u} = \underset{\sim}{F}$$

$$\dot{\underset{\sim}{v}} = \underset{\sim}{M}^{-1}\underset{\sim}{F} - \underset{\sim}{M}^{-1}\underset{\sim}{C}\underset{\sim}{v} - \underset{\sim}{M}^{-1}\underset{\sim}{K}\underset{\sim}{u}$$

$$\begin{pmatrix}\dot{\underset{\sim}{u}}\\ \dot{\underset{\sim}{v}}\end{pmatrix} = \begin{bmatrix}0 & I\\ -\underset{\sim}{M}^{-1}\underset{\sim}{K} & -\underset{\sim}{M}^{-1}\underset{\sim}{C}\end{bmatrix}\begin{pmatrix}\underset{\sim}{u}\\ \underset{\sim}{v}\end{pmatrix} + \begin{pmatrix}0\\ \underset{\sim}{M}^{-1}\underset{\sim}{F}\end{pmatrix}$$

WHERE:

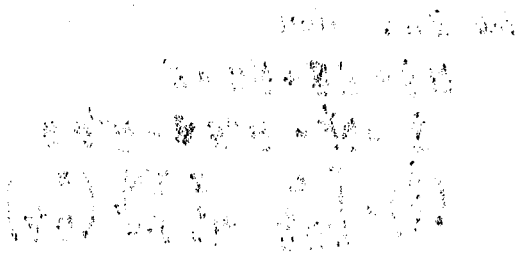
$\underset{\sim}{M}$ ... MASS MATRIX, SYMM. POS. DEF.

$\underset{\sim}{C}$ ... DAMPING MATRIX, SYMM, POS. SEMI-DEF.

$\underset{\sim}{K}$ ... STIFFNESS MATRIX, SYMM. POS. SEMI-DEF.

$\underset{\sim}{F} = \underset{\sim}{F}(t)$ ... EXTERNAL FORCE VECTOR (PRESC.)

$\underset{\sim}{d}$ ... DISPLACEMENT VECTOR.



$\dot{\underset{\sim}{q}}$ ... VELOCITY VECTOR (DERIV. OF $\underset{\sim}{q}$ WRT t)

$\ddot{\underset{\sim}{q}}$ ... ACCELERATION VECTOR (2nd DERIV. OF $\underset{\sim}{q}$ WRT t)

IN THIS CASE WE LET

$$\underset{\sim}{y} = \begin{Bmatrix} \underset{\sim}{q} \\ \dot{\underset{\sim}{q}} \end{Bmatrix} \tag{10}$$

THUS

$$\underset{\sim}{G} = \begin{bmatrix} \underset{\sim}{0} & \underset{\sim}{I} \\ -\underset{\sim}{M}^{-1}\underset{\sim}{K} & -\underset{\sim}{M}^{-1}\underset{\sim}{C} \end{bmatrix}, \qquad \underset{\sim}{H}(t) = \begin{Bmatrix} \underset{\sim}{0} \\ \underset{\sim}{M}^{-1}\underset{\sim}{F}(t) \end{Bmatrix} \tag{11),(12}$$

IF WE LET

$$\underset{\sim}{\Psi}(\underset{\sim}{G}) = \begin{Bmatrix} \underset{\sim}{\Psi}_1(\underset{\sim}{G}) \\ \underset{\sim}{\Psi}_2(\underset{\sim}{G}) \end{Bmatrix}$$ THEN THE EIGENPROBLEM CAN BE EXPRESSED AS (13)

$$\left(\begin{bmatrix} \underset{\sim}{0} & \underset{\sim}{I} \\ -\underset{\sim}{M}^{-1}\underset{\sim}{K} & -\underset{\sim}{M}^{-1}\underset{\sim}{C} \end{bmatrix} - \begin{bmatrix} \lambda(\underset{\sim}{G})\underset{\sim}{I} & \underset{\sim}{0} \\ \underset{\sim}{0} & \lambda(\underset{\sim}{G})\underset{\sim}{I} \end{bmatrix} \right) \begin{Bmatrix} \underset{\sim}{\Psi}_1(\underset{\sim}{G}) \\ \underset{\sim}{\Psi}_2(\underset{\sim}{G}) \end{Bmatrix} = \begin{Bmatrix} \underset{\sim}{0} \\ \underset{\sim}{0} \end{Bmatrix} \tag{14}$$

OR EQUIVALENTLY,

$$\underset{\sim}{\Psi}_2(\underset{\sim}{G}) - \lambda(\underset{\sim}{G})\,\underset{\sim}{\Psi}_1(\underset{\sim}{G}) = \underset{\sim}{0} \tag{15}$$

$$-\underset{\sim}{M}^{-1}\underset{\sim}{K}\,\underset{\sim}{\Psi}_1(\underset{\sim}{G}) - (\underset{\sim}{M}^{-1}\underset{\sim}{C} + \lambda(\underset{\sim}{G})\underset{\sim}{I})\,\underset{\sim}{\Psi}_2(\underset{\sim}{G}) = \underset{\sim}{0} \tag{16}$$

COMBINING (15) AND (16) AND MULTIPLYING BY $\underset{\sim}{M}$ YIELDS A QUADRATIC EIGENPROBLEM:

$$(\underset{\sim}{K} + \lambda(\underset{\sim}{G})\,\underset{\sim}{C} + \lambda(\underset{\sim}{G})^2\underset{\sim}{M})\,\underset{\sim}{\Psi}_1(\underset{\sim}{G}) = 0 \tag{17}$$

FROM PREVIOUS WORK WE KNOW THAT THERE EXIST A SET OF VECTORS $\hat{\underset{\sim}{\Psi}}_\ell$ SUCH THAT

$$\hat{\underset{\sim}{\Psi}}_\ell^T\,\underset{\sim}{M}\,\hat{\underset{\sim}{\Psi}}_m = \delta_{\ell m}$$

$$\hat{\underset{\sim}{\Psi}}_\ell^T\,\underset{\sim}{K}\,\hat{\underset{\sim}{\Psi}}_m = \delta_{\ell m}\,\omega_\ell^2 \quad \text{(NO SUM)}$$

NOW IF WE ASSUME THAT $\underset{\sim}{C}$ IS DIAGONAL IN THE $\hat{\underset{\sim}{\Psi}}$ BASIS (WHICH IS THE CASE FOR RAYLEIGH DAMPING WHERE $\underset{\sim}{C} = a\underset{\sim}{M} + b\underset{\sim}{K}$ a, b CONSTANTS) THEN

$$\hat{\underset{\sim}{\Psi}}_\ell^T \underset{\sim}{C} \hat{\underset{\sim}{\Psi}}_m = 2\xi_\ell \omega_\ell \delta_{\ell m} \quad \text{(NO SUM)}$$

THUS PREMULTIPLYING (17) BY $\hat{\underset{\sim}{\Psi}}_\ell^T$ AND EXPANDING $\underset{\sim}{\Psi}_1(G)$ IN TERMS OF THE $\hat{\underset{\sim}{\Psi}}_\lambda$ WE FIND

$$\lambda_\ell(\underset{\sim}{G})^2 + 2\xi_\ell \omega_\ell \lambda_\ell(\underset{\sim}{G}) + \omega_\ell^2 = 0 \quad \text{(NO SUM)}$$

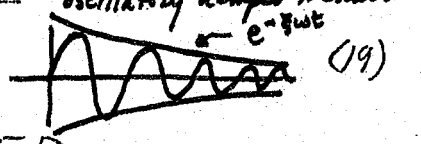
DROPPING THE ℓ SUBSCRIPT WE FIND THE UNDERLYING FORM OF THE $\lambda(\underset{\sim}{G})$ EIGENPROBLEM TO BE

$$\lambda(\underset{\sim}{G})^2 + 2\xi\omega\lambda(\underset{\sim}{G}) + \omega^2 = 0 \qquad (18)$$

THE SOLUTION OF (18) TAKES ON THE FOLLOWING CHARACTERISTIC FORMS:

(i) $0 \leq \xi < 1 \Rightarrow$ <u>UNDERDAMPED</u>

$$\lambda_{1,2}(\underset{\sim}{G}) = -\xi\omega \pm i\omega\sqrt{1-\xi^2} \qquad (19)$$

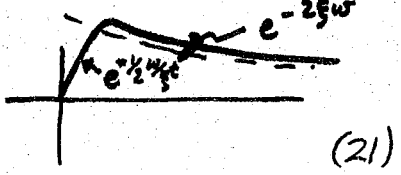


(ii) $\xi = 1 \Rightarrow$ <u>CRITICALLY DAMPED</u>

$$\lambda_{1,2}(\underset{\sim}{G}) = -\omega \quad \text{(DOUBLE ROOT)} \qquad (20)$$

(iii) $\xi > 1 \Rightarrow$ <u>OVERDAMPED</u>

$$\lambda_{1,2}(\underset{\sim}{G}) = -\xi\omega \pm \omega\sqrt{\xi^2-1} \qquad (21)$$



REMARKS:

i) NO DAMPING $\Rightarrow \xi = 0 \Rightarrow \lambda_{1,2}(\underset{\sim}{G}) = \pm i\omega$ (CONJUGATE IMAGINARY ROOTS).

ii) IN ALL CASES, SINCE ξ AND $\omega \geq 0$ THE EIGENVALUES OF $\underset{\sim}{G}$ ARE CONFINED TO THE NEGATIVE HALF-PLANE OF $\mathbb{C}$ INCLUDING THE IMAGINARY AXIS.

iii) <u>IN CONSIDERING LMS METHODS IT IS IMPORTANT TO INSURE THAT THE STABILITY REGION IN THE COMPLEX PLANE ENCLOSES THE SPECTRUM OF THE MATRIX $\underset{\sim}{G}$ FOR THE CASE IN QUESTION.</u>

3) STABILITY OF LMS METHODS

a) REDUCTION TO AN SDOF PROBLEM

AS WITH PREVIOUS ANALYSES THE REDUCTION TO A MODEL PROBLEM PLAYS AN ESSENTIAL ROLE. WE WILL ASSUME THAT $\underset{\sim}{G}$ POSSESES LINEARLY INDEPENDENT EIGENVECTORS.

NOTATION:

$\lambda = \lambda(\underset{\sim}{G})$ ... EIGENVALUE

$\underset{\sim}{\Psi} = \underset{\sim}{\Psi}(\underset{\sim}{G})$ ... EIGENVECTOR

$\underset{\sim}{P} = \underset{\sim}{P}(\underset{\sim}{G})$ ... MATRIX OF EIGENVECTORS

$\underset{\sim}{\Lambda} = \underset{\sim}{\Lambda}(\underset{\sim}{G})$ ... DIAGONAL MATRIX OF EIGENVALUES

THUS

$$\underset{\sim}{P} = [\Psi_1, \Psi_2, \cdots, \Psi_N]$$

$$\underset{\sim}{\Lambda} = diag[\lambda_1, \lambda_2, \cdots, \lambda_N] = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_N \end{bmatrix}$$

WHERE N IS THE DIMENSION OF $\underset{\sim}{G}$

THUS $\underset{\sim}{P}$ AND $\underset{\sim}{\Lambda}$ SATISFY

$$\underset{\sim}{P}^{-1} \underset{\sim}{G} \underset{\sim}{P} = \underset{\sim}{\Lambda} \tag{22}$$

DEFINING

$$\underset{\sim}{z} = \underset{\sim}{P}^{-1} \underset{\sim}{y} \tag{23}$$

ENABLES US TO OBTAIN FROM THE HOMOGENOUS FORM OF (1), THE FOLLOWING UNCOUPLED SYSTEM

$$\dot{\underset{\sim}{z}} = \underset{\sim}{\Lambda} \underset{\sim}{z} \tag{24}$$

WHICH IS CHARACTERIZED BY THE GENERIC SCALAR EQUATION

$P^{-1}\cdot$ eqn(2) $= \sum(\alpha_i \underbrace{P^{-1}y_{n+1-i}}_{z_{n+1-i}} + \Delta t\beta_i \underbrace{P^{-1}G}_{P^{-1}\lambda} y_{n+1-i})$ 7

$$\dot{z} = \lambda z \qquad \underbrace{\Lambda z_{n+1-i}}_{} \qquad (25)$$

LIKEWISE THE HOMOGENEOUS FORM OF THE LMS ALGORITHM (2), CAN BE UNCOUPLED

$$\sum_{i=0}^{k} (\alpha_i z_{n+1-i} + \Delta t \beta_i \Lambda z_{n+1-i}) = 0 \qquad (26)$$

WHICH IS CHARACTERIZED BY

$$\sum_{i=0}^{k} (\alpha_i + \Delta t \lambda \beta_i) z_{n+1-i} = 0 \qquad (27)$$

TO DETERMINE STABILITY ONE ASSUMES THE SOLUTION OF THE LINEAR DIFFERENCE EQUATION (27) TO BE OF THE FORM

$$z_{n+1-i} = c\,\zeta^{n+1-i}, \quad c \ldots \text{A CONSTANT}$$

SUBSTITUTION INTO (27) YIELDS A k^{TH} ORDER POLYNOMIAL IN ζ, THE ROOTS OF WHICH DETERMINE THE STABILITY OF THE LMS METHOD.

$$\sum_{i=0}^{k} (\alpha_i + \Delta t \lambda \beta_i) \zeta^{n+1-i} = 0 \qquad (28)$$

if taping $k=1$

$(\alpha_0 + \Delta t\lambda\beta_0)\zeta^{n+1} + (\alpha_1 + \Delta t\lambda\beta_1)\zeta^n = 0 \Rightarrow \zeta^n \left[(\alpha_1 + \Delta t\lambda\beta_1) + \zeta(\alpha_0 + \Delta t\lambda\beta_0)\right] = 0$

or $\zeta = 0$ (n roots) or $\zeta = -\dfrac{\alpha_1 + \Delta t\lambda\beta_1}{\alpha_0 + \lambda\Delta t\beta_0}$

explicit $\zeta = \dfrac{1+\Delta t\lambda}{1-\Delta t\lambda}$; implicit B.D. $\zeta = \dfrac{1}{1-\lambda\Delta t}$

b) CONCEPTS OF STABILITY

DEFINITION 1: A LMS METHOD OF THE FORM (2) IS SAID TO BE ABSOLUTELY STABLE, FOR A FIXED $\Delta t\lambda$ IF ALL ζ SATISFYING (28) ARE SUCH THAT $|\zeta| \leq 1$

for explicit $\Rightarrow$ for $\lambda < 0$ any Δt will do; for BD $+\lambda\Delta t \leq 0$

DEFINITION 2: THE REGION OF ABSOLUTE STABILITY OF A LMS METHOD IS THE SET OF $\Delta t\lambda \in \mathbb{C}$ AT WHICH IT IS ABSOLUTELY STABLE.

REMARK: THE DEFINITION OF ABSOLUTE STABILITY IS VERY CLOSE TO THAT OF SPECTRAL STABILITY GIVEN PREVIOUSLY. WITH THE EXCEPTION THAT NO SPECIAL ACCOUNT IS GIVEN TO MULTIPLE ROOTS OF UNIT MODULUS, WHICH POTENTIALLY GIVE RISE TO WEAK INSTABILITIES. ONE CAN SHOW THAT THE ROOTS OF (28) AND THE EIGENVALUES OF AN ASSOCIATED AMPLIFICATION MATRIX ARE THE SAME BY CONSTRUCTING A ONE-STEP MULTIVALUE METHOD EQUIVALENT TO THE GIVEN LMS METHOD, VIZ.

$$\underbrace{\begin{Bmatrix} z_{n+1} \\ z_n \\ \vdots \\ z_{n+2-k} \end{Bmatrix}}_{k\times 1} = \underset{k\times k}{\underset{\sim}{A}} \underbrace{\begin{Bmatrix} z_n \\ z_{n-1} \\ \vdots \\ z_{n+1-k} \end{Bmatrix}}_{k\times 1} \tag{29}$$

WHERE

$$\underset{\sim}{A} = \begin{bmatrix} \dfrac{-(\alpha_1+\Delta t\lambda\beta_1)}{(\alpha_0+\Delta t\lambda\beta_0)} & \dfrac{-(\alpha_2+\Delta t\lambda\beta_2)}{(\alpha_0+\Delta t\lambda\beta_0)} & \cdots & \cdots & \dfrac{-(\alpha_k+\Delta t\lambda\beta_k)}{(\alpha_0+\Delta t\lambda\beta_0)} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & & 0 & \vdots \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{30}$$

DEFINITION 3: A LMS METHOD IS A-STABLE IF SOLUTIONS OF (27) $\to 0$ AS $n\to\infty$ WHEN $\mathrm{Re}\,\lambda < 0$.

REMARKS:

i) PHYSICALLY SPEAKING, AN A-STABLE ALGORITHM IS ONE WHICH PRODUCES SOLUTIONS WHICH DECAY TO ZERO WHENEVER THE CORRESPONDING EXACT SOLUTIONS OF (25) DECAY TO ZERO. CLEARLY IF AN ALGORITHM IS A-STABLE THEN THE REGION OF ABSOLUTE STABILITY CONTAINS THE LEFT HALF-PLANE OF $\mathbb{C}$.

ii) THE CONDITION OF A-STABILITY PLACES NO LIMITATION ON THE SIZE OF Δt, CONSEQUENTLY IT IS CLOSELY

ie look at trapezoidal algorithm no condition on Δt exists except >0

RELATED TO WHAT WE PREVIOUSLY TERMED UNCONDITIONAL SPECTRAL STABILITY. THE ONLY DIFFERENCE BEING THAT SPECTRAL STABILITY ALLOWS EIGENVALUES OF UNIT MODULUS WHICH POTENTIALLY COULD CONSERVE A SOLUTION RATHER THAN CONTRACT IT TO ZERO.

IT SHOULD BE APPARENT THAT A-STABLE ALGORITHMS ARE IMPORTANT FOR MANY PHYSICAL PROBLEM CLASSES. HOWEVER, WITHIN THE CLASS OF LMS METHODS, THE SUBCLASS OF A-STABLE METHODS IS SEVERLY DELIMITED. THIS IS THE CONTENT OF A CELEBRATED THEOREM DUE TO DAHLQUIST.

DAHLQUIST'S THEOREM ON A-STABLE LMS METHODS

1) AN EXPLICIT A-STABLE LMS METHOD DOES NOT EXIST.
2) A THIRD-ORDER ACCURATE A-STABLE LMS METHOD DOES NOT EXIST.
3) THE SECOND-ORDER ACCURATE A-STABLE LMS METHOD WITH THE SMALLEST ERROR CONSTANT IS THE TRAPEZOIDAL RULE. (RECALL THAT THE ERROR CONSTANT IS THE COEFFICIENT OF Δt^{k+1} IN THE LOCAL TRUNCATION ERROR.)

REMARKS:

i) THE UPSHOT OF THIS THEOREM IS THAT IF WE SEEK AN A-STABLE LMS METHOD WHICH POSSESSES SOME SPECIAL FEATURE (SUCH AS HIGH-FREQUENCY NUMERICAL DISSIPATION) WE NECESSARILY ENTAIL SOME LOSS OF ACCURACY WITH RESPECT TO THE TRAPEZOIDAL RULE.

ii) SINCE THE SPECTRUM OF THE G-MATRIX FOR THE EQUATION OF MOTION FALLS WITHIN THE LEFT-HALF OF THE COMPLEX PLANE (AND FOR THE UNDAMPED CASE FALLS ON THE IMAGINARY AXIS) IT WOULD APPEAR THAT THE TRAPEZOIDAL RULE IS THE CANONICAL A-STABLE LMS METHOD FOR STRUCTURAL DYNAMICS. THE WIDESPREAD USE OF THE TRAPEZOIDAL RULE IN THIS CONTEXT APPEARS TO CONFIRM THIS OBSERVATION.

iii) THERE ARE A-STABLE LMS METHODS OTHER THAN TRAPEZOIDAL RULE WHICH ARE USEFUL FOR STRUCTURAL DYNAMICS (FOR EXAMPLE PARKS METHOD)

(v) MANY STRUCTURAL DYNAMICS ALGORITHMS DO NOT FALL WITHIN THE CLASS OF FIRST-ORDER LMS METHODS CONSIDERED THUS FAR. (FOR EXAMPLE THE NEWMARK FAMILY OF ALGORITHMS)

REFERENCES:

(D1) G. DAHLQUIST, "A SPECIAL STABILITY PROBLEM FOR LINEAR MULTISTEP METHODS", BIT 3, 27-43 [1963].

(G1) C. W. GEAR, NUMERICAL INITIAL VALUE PROBLEMS IN ORDINARY DIFFERENTIAL EQUATIONS, PRENTICE-HALL, ENGLEWOOD CLIFFS, N. J., [1971]

Remarks

- If an algorithm is A-stable region of absolute stability contains left half of complex $\bar{z}$ plane

- Condition of A-stability places no restrictions on Δt (similar to unconditional spectral stability - however it allows $|\xi| = 1$).

A-stable places severe restrictions on physical systems.

Dahlquist Theorem

i) An explicit A-stable LMS method doesn't exist,

ii) That a 3rd order accurate LMS method does not exist (A-stable)

iii) The 2nd order accurate LMS (A-stable) method w/ best error constant trapezoidal rule

remember $\|e\| \leq \bar{\bar{C}} \Delta t^{k+1}$

Remarks

1. A Special LMS method that has A-stability & desirable effect (ie high frequency numerical dissipation) will be less accurate than trapezoidal

2. Since spectr of G is in left hand plane, trapezy rule is canonical rule.
   ie For eqn of motion trapezoidal rule is canonical LMS A-stable method.

3. Other LMS methods that are used extensively for eqns of motion (Park's method - keeps more historical data - more computational).

→ # Newmarks cannot be recast into 1st order system

4. There are other integ. algorithms for structural dynamics (ie Newmarks)

ME 235B - SPRING 80-81
Finite Element Methods

Stanford University
Instructor: T. Hughes

HOMEWORK

CESAR LEVY

Consider the equations of two-dimensional, classical, linear isotropic, elasticity theory on the domain illustrated in Figure 1

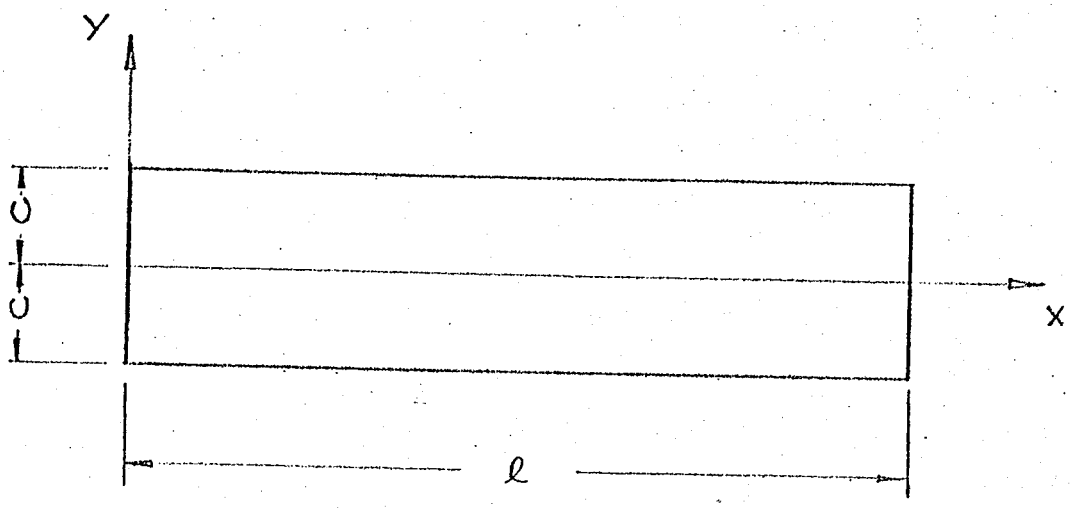


Figure 1

Assume there is no body force, and the boundary conditions are given as follows:

(displacement)

$$u_1(0,0) = u_2(0,0) = 0$$

$$u_1(0,\pm c) = 0$$

(traction)

$$t_1(x,\pm c) = t_2(x,\pm c) = 0 \quad , \quad x \in]0,\ell[$$

The cross diag and bisect patterns were found not to give good results for $\nu = .3$ 1×1 λ,μ w/ hourglass / 2×2 w/ compatible modes give same results

$u_2(l,0)$: actual / num is -244.14 / -242.98

$u_1(0,y)$	actual / num
	0 / 0
$y=0$	$-.1295$ / $-.1372$
.5	$-.2072$ / $-.2207$
1.0	$-.1813$ / $-.1953$
1.5	0 / 0
2.0	

$\sigma_{11}(7,y)$	actual / num
y	
0.25	.4299 / .4218
.75	1.266 / 1.266
1.25	2.109 / 2.110
1.75	2.953 / 2.953

$\sigma_{12}(7,y)$	actual / num
y	
0.25	$-.369$ / $-.367$
.75	$-.3223$ / $-.3204$
1.25	$-.2285$ / $-.2266$
1.75	$-.0879$ / $-.0877$

2×2 quad gave large errors in $u_1(0,y)$ 8-10% in σ_{11} 3~14% in τ_{12}

1×1 λ 2×2 μ quad " " in $u_1(0,y)$ 8-10% in σ_{11} 3~14% : τ_{12}

$\nu = .499$ 1×1 λ,μ w/ hourglass & 2×2 w/ comp

$u_2(l,0)$: actual / num is -205.47 / -204.11

$u_1(0,y)$	act / num
$y=0$	0 / 0
.5	$-.1317$ / $-.1466$
1.0	$-.2807$ / $-.2369$
1.5	$-.7843$ / $-.2117$
2.0	0 / 0

$\sigma_{11}(7,y)$	actual / num
0.25	.4219 / .4227
0.75	1.266 / 1.264
1.25	2.109 / 2.112
1.75	2.953 / 2.952

$\tau_{12}(7,y)$	
	$-.369$ / $-.367$
	$-.3223$ / $-.3205$
	$-.2285$ / $-.2268$
	$-.0879$ / $-.0859$

1×1 λ 2×2 μ gave large errors in $u_1(0,y)$ 5% in σ_{11} 1-6% in σ_{12}

2×2 quad 67% in $u_2(1,0)$ large errors in $u_2(0,y)$ large errors in σ_{11}

would recommend quad overall w/ 2×2 w/incomp
or 1×1 w/hourglass

	QUAD	BISECT	CROSS DIAG	
even worse 1 λ 2 μ WORSE THAN (++)	$\nu = .3$ 2001 · $\nu = .499$ 2003 ·	not good very huge cut $\nu = .499$ 2012 · very bad $\nu = .3$ 2011 · σ_{12} has wrong sign	$\nu = .499$ 2017 · $\nu = .3$ 2016 ·	(like 1 λ μ + hour) not good 2 × 2
Same as hourglass V.G Same as *	$\nu = .3$ 2005 · $\nu = .499$ 2008 ·	like the above $\nu = .499$ 2020 ·	$\nu = .499$ 2019 ·	(like 1 λ μ + hour) 2 × 2 w/ uncomp
not good comp disp ~~NOT GOOD~~ ~~Same as *~~	$\nu = .3$ 2004 · $\nu = .499$ 2007 · (++)	$\nu = .499$ 2014 · ~~like the above~~ like the above	$\nu = .499$ 20?? not good errors in $u_1 \approx 150\%$ in σ_{11} & τ_{12} high (++)	1 × 1 λ 2 × 2 μ
very good on σ_{11} V.G.	$\nu = .3$ 2006 · * $\nu = .499$ 2009 ·	$\nu = .499$ 2018 · like the above not good	$\nu = .499$ 2018 · ~~just as bad~~ same as λ 1 & μ 2 w/ τ_{12} changing sign also	1 × 1 w/ hourglass
	2000 data check 2007 2018 2010 purged.	2010 data check	2015 data check	

DATE: May 6, 1981

TO : Students in Applied Mechanics

FROM : Tom Hughes and Charles Steele

SUBJECT: Applied Mechanics Industrial Affiliates Dinner

You are invited for cocktails and dinner on Wednesday evening at 6:00 p.m. at the Stanford Faculty Club you are strongly urged also to attend the presentations on Wednesday and Thursday, May 13, 14, 1981. Please sign up with Theo, Room 252, Durand Bldg. by Friday MAY 8, 1981

$$\left.\begin{aligned} t_1(\ell, y) &= 0 \\ t_2(\ell, y) &= \frac{P}{2I}(c^2 - y^2) \end{aligned}\right\} \quad y \in]-c, c[$$

$$\left.\begin{aligned} t_1(0, y) &= \int^{+} \frac{P\ell y}{I} \\ t_2(0, y) &= -\frac{P}{2I}(c^2 - y^2) \end{aligned}\right\} \quad y \in]-c, 0[\cup]0, c[$$

where P is a given constant and $I = 2c^3/3$.

The traction boundary conditions are those encountered in simple bending theory for a cantilever beam with root section at $x = 0$, that is, parabolically varying end shear and linearly varying bending stress at the root. The displacement boundary conditions allow the root section to warp.

The exact solution of this problem is easily derived and is given as follows:

$$t_{11} = -\frac{P\tilde{x}y}{I}$$

$$t_{22} = 0$$

$$t_{12} = \frac{P}{2I}(c^2 - y^2)$$

$$\frac{6E^1I}{P} u_1(x, y) = -y\{3(\ell^2 - \tilde{x}^2) + (2 + \nu^1)(y^2 - c^2)\}$$

$$\frac{6E^1I}{P} u_2(x, y) = (\tilde{x}^3 - \ell^3) - \{(4 + 5\nu^1)c^2 + 3\ell^2\}(\tilde{x} - \ell) + 3\nu^1\tilde{x}y^2$$

where $\tilde{x} = \ell - x$ and

	E^1	ν^1
plane stress	E	ν
plane strain	$E/(1-\nu^2)$	$\nu/(1-\nu)$

Obtain approximate solutions of the above elasticity problem by using the linear static analysis program "LEARN."

- Employ the following data in your calculations:

  $P = -1$, $\ell = 16$, $c = 2$, $E = 1$, $\nu = .3$ and $.499$

  (The latter value of Poissons' ratio corresponds to the nearly-incompressible case.)

- The mesh to be used is depicted in Figure 2. (Only half the domain need be modelled since the x-axis is a line of anti-symmetry.) You will need to compute nodal forces corresponding to the traction boundary conditions along $x = 0$ and $x = \ell$.

- Assume plane strain conditions.

- Use 4-node quadrilaterals and employ each of the following quadrature rules:

  (i) 2×2

  (ii) 2×2 μ-term, 1×1 λ-term

  (iii) 2×2 with incompatible modes

  Also employ (iv) the hourglass stiffness option.

- Use 3-node triangles in the

  (v) bisection pattern

  (vi) cross-diagonal pattern

  (There are twelve cases in all.)

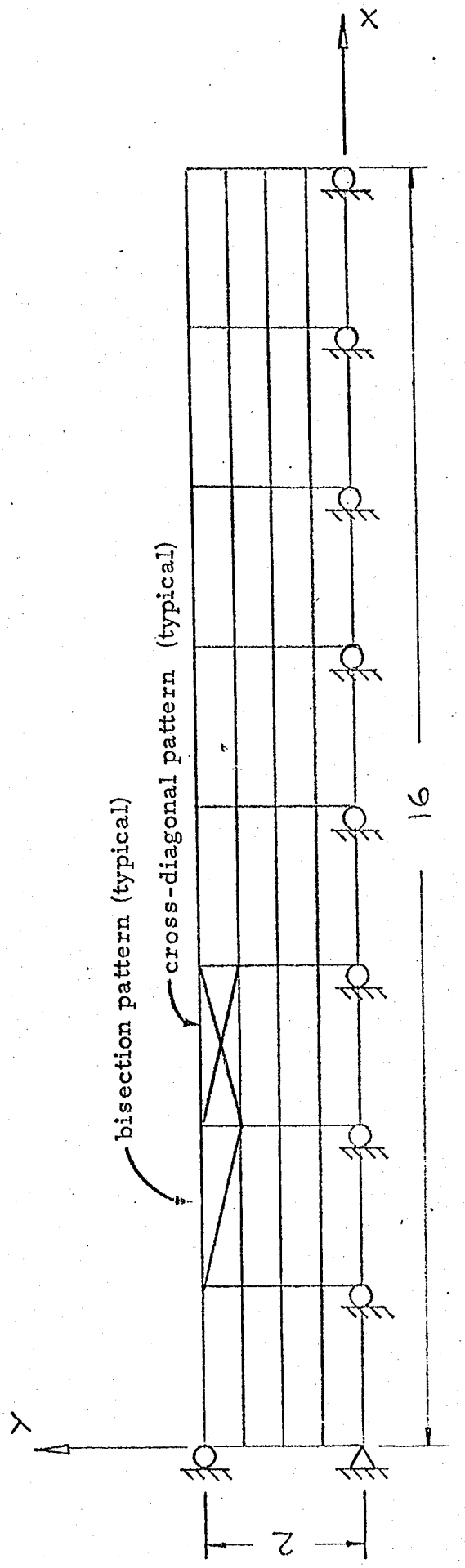

X
Y
16
2
bisection pattern (typical)
cross-diagonal pattern (typical)

Figure 2

- In each case, compare the computed vertical tip displacement [i.e. $u_2(16,0)$], the horizontal displacement of the root section (i.e. $u_1(0,y)$, $y \in [-c,c]$) and the bending and shear stress (i.e. σ_x and τ_{xy}, respectively) for $x = 7$, $y \in [-c,c]$, with the exact solution.

- Write a brief report summarizing your findings and include an evaluation of the performance of each element.

Instructions for Using "LEARN"

Consult the LEARN User's Manual regarding deck set up.

To save keypunch labor, use the generation options on coordinate and element data.

Your first run should be a data check run (i.e., MODE.EQ.0 on the master control card). Run the 2×2 quadrature case for $\nu = .3$. Carefully check the output to insure that you have keypunched your data correctly.

Keep using the data check mode until you are satisfied that your input data is correct. Then set MODE.EQ.1 which will cause the program to execute. The vertical displacement $u_2(16,0) = -220.4$. If you obtain this value, begin running the other cases. If you don't obtain the correct value, or if you are having any trouble whatsoever, see me or the T.A. immediately and bring your output and deck.

4 node quad : based on the results for all the compared items (ie $u_2(16,0)$, $u_1(0,y)$, $\sigma_x(7,y)$ $\tau_{xy}(7,y)$) it was found that the 2x2 w/incompressible and the 1x1 w/ hourglass elements were the best and provided fairly accurate results for the displacements and stresses. The results degraded somewhat for the incompressible case. The next best case was the reduced selective integration case ($1\times1\ \lambda$ $2\times2\ \mu$) which was fairly accurate in the stresses but not in the displacements. The results were somewhat upgraded for the incompressible case. The 2x2 case shows degradation of results as we approach the incompressible case. Severe lock-up of the elements occurs; the extensional stresses degrade but the shear stresses remain fairly accurate. This was considered to be the worst case. In terms of overall time, for each value of ν, they are comparable.

3 node bisection : The first thing we note is that the results are the same for all the cases run (ie $\nu = .499$: 2x2, 2x2 w/incomp, 1x1 w/hourglass & reduced selective integration since $\nabla \cdot u \approx$ constant). As the value of $\nu \to .5$ the elements suffer severe lock-up, but overall the displacements agree better; σ_x becomes worse and τ_{xy} becomes better. The time comparisons are somewhat worse than the 4-node quad. With respect to the 4 node quad as $\nu \to .5$ the displacement $u_1(0,y)$ are better and the stresses σ_x, τ_{xy} degrade.

3 node cross-diag : as in the bisection case we note that the results are the same for all cases for the same value of ν (since $\nabla \cdot u \approx$ constant). We note that the errors in $u_1(0,y)$ decrease, that the errors in σ_x decrease, but that the errors in τ_{xy} increase as $\nu \to .5$. We also note a

reversals in sign in τ_{xy}. By looking at a quadrilateral, we find that τ_{xy} exhibits a checkerboard pattern which should not happen. For the 2×2 case, with respect to the 4 node quad as $\nu \to .5$, $u_1(0,y)$ is better; $\sigma_x(7,y)$ is better but τ_{xy} is worse. With respect the 3 node bisection, as $\nu \to .5$, $u_1(0,y)$ is worse, σ_x is better and τ_{xy} becomes worse.

Overall comparisons:

The 4-node quad (2×2 w/incomp and 1×1 λ&μ w/ hourglass) by far provides the most superior results. In general for $\nu = .3$ the 4-node quad does much better than the other two elements. In general for $\nu = .499$ the 4 node quad 2×2 case gives better results in τ_{xy}; the 2×2 cross diag gives better results in σ_x & $u_2(l)$, and the 2×2 bisection gives better results in $u_1(0,y)$.

4 node quad $u_2(16,0)$

ν	EXACT	2 x 2	2x2 μ 1x1 λ	2x2 W/ INCOMP	1x1 λ & μ W/ HOURGLASS
.3	-244.14	-220.78	-222.54	-242.98	-242.98
ERROR		(10)	(9)	(0.5)	(0.5)
.499	-205.47	-68.68	-192.65	-204.12	-204.12
ERROR		(66)	(6)	(0.7)	(0.7)

$u_1(0,y)$ $\nu = .3$ (ERROR IN PARENTH)

Y	EXACT							
0	0	0.0		0.0		0.0	(0)	0.0	(0)
.5	-.1295	-.3455	(167)	-.3375	(161)	-.1372	(6)	-.1372	(6)
1.0	-.2072	-.5809	(180)	-.5677	(174)	-.2207	(7)	-.2207	(7)
1.5	-.1813	-.5606	(209)	-.5490	(203)	-.1953	(8)	-.1953	(8)
2.0	0	0.0		0.0		0.0	(0)	0.0	(0)
	STIFF & LOAD FORM	.06		.06		.06		.06	
$\nu = .499$	STIFF FACTORIZ	.00		.00		.00		.00	

Y	EXACT							
0	0	0.		0		0	(0)	0	(0)
.5	-.1317	-.5089	(286)	-.2727	(107)	-.1466	(11)	-.1466	(11)
1.0	-.2107	-.8248	(291)	-.4595	(118)	-.2369	(12)	-.2369	(12)
1.5	-.1843	-.7391	(301)	-.4591	(148)	-.2117	(10)	-.2117	(10)
2.0	0	0		0		0	(0)	0	(0)
	STIFF & LOAD FORM	.04		.06		.04		.04	
	STIFF FACTORIZ	.04		.00		.04		.04	

4 node quad $\sigma_x(7,y)$ $\nu = .3$ (error in parent.)

y	EXACT	2×2		2×2 μ 1×1 λ		2×2 w/incomp		1×1 λ&μ w/HOUR	
.25	.4219	.3871	(8)	.3903	(7)	.4218	(~0)	.4218	(~0)
.75	1.2656	1.161	(8)	1.171		1.266	(~0)	1.266	(~0)
1.25	2.1094	1.934	(8)	1.949		2.110	(~0)	2.110	(~0)
1.75	2.9531	2.706	(8)	2.726	(8)	2.953	(~0)	2.955	(~0)

$\nu = .499$ (error in parenth)

y	EXACT								
.25	.4219	.1252	(70)	.4005	(5)	.4227	(~0)	.4227	(~0)
.75	1.2656	.3934	(69)	1.206	(5)	1.264	(~0)	1.264	(~0)
1.25	2.1094	.7029	(66)	1.996	(5)	2.112	(~0)	2.112	(~0)
1.75	2.9531	1.081	(63)	2.800	(5)	2.952	(~0)	2.952	(~0)

$\tau_{xy}(7,y)$ $\nu = .3$ (error in parent

y	EXACT								
.25	-.3691	-.3580	(3)	-.3596	(3)	-.3673	(~.5)	-.3673	(~.5)
.75	-.3223	-.3150	(2)	-.3160	(2)	-.3204	(~.6)	-.3204	(~.6)
1.25	-.2285	-.2280	(~0)	-.2285	(~0)	-.2266	(~1)	-.2266	(~1)
1.75	-.0879	-.0991	(13)	-.09695	(10)	-.0858	(~2)	-.0858	(~2)

$\nu = .499$ (error in parent)

.25	-.3691	-.3523	(4)	-.3629	(2)	-.3671	(~.5)	-.3671	(~.5)
.75	-.3223	-.3103	(4)	-.3184	(1)	-.3205	(~.6)	-.3205	(~.6)
1.25	-.2285	-.2284	(~0)	-.2266	(~1)	-.2268	(~1)	-.2268	(~1)
1.75	-.0879	-.1089	(24)	-.0921	(5)	-.0856	(~3)	-.0856	(~3)

3-node triangle $u_2(16,0)$ BISECTION

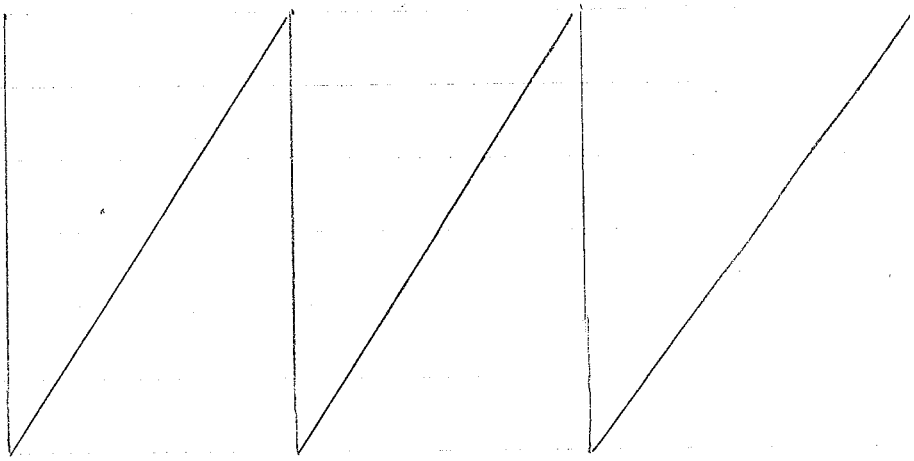
ν	EXACT	2x2	2x2 w/INCOMP	1x1 λθμ w/hour	1x1 λ 2x2μ
.3	-244.14	-183.15 (25)	———	———	———
.499	-205.47	-39.03 (81)	-39.03	-39.03	-39.03

$u_1(0,y)$ $\nu = .499$ (error in parenthesis)

y	EXACT				
0	0	0	0	0	0
.5	-.1317	-.1630 (23)	-.1630	-.1630	-.1630
1.0	-.2107	-.4392 (108)	-.4392	-.4392	-.4392
1.5	-.1843	-.5128 (178)	-.5128	-.5128	-.5128
2.0	0	0	0	0	0
	STIFF & LOAD FORM	.04	.04	.04	.04
	STIFF FACTOR/S	.04	.04	.04	.04

$\nu = .3$ (error in parenthesis)

Y	EXACT		
0	0	0	
.5	-.1295	-.5485 (324)	
1.0	-.2072	-.9056 (337)	
1.5	-.1813	-.8333 (360)	
2.0	0	0.0	
	STIFF & LOAD FORM	.04	
	STIFF FACTOR/S	.04	

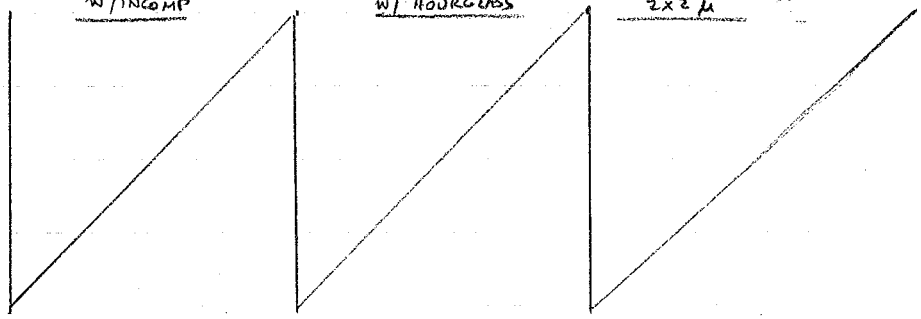


3-node triangle

BISECTION

$\sigma_x(7,y)$ $\nu = .3$ (error in parenth)

y	EXACT	2×2		2×2 W/INCOMP	1×1 2d/μ W/ HOURGLASS	1×1 2×2 μ
.375	.6328	.7142	(13)			
.875	1.4765	1.378	(7)			
1.375	2.3203	2.041	(12)			
1.875	3.164	2.703	(15)			

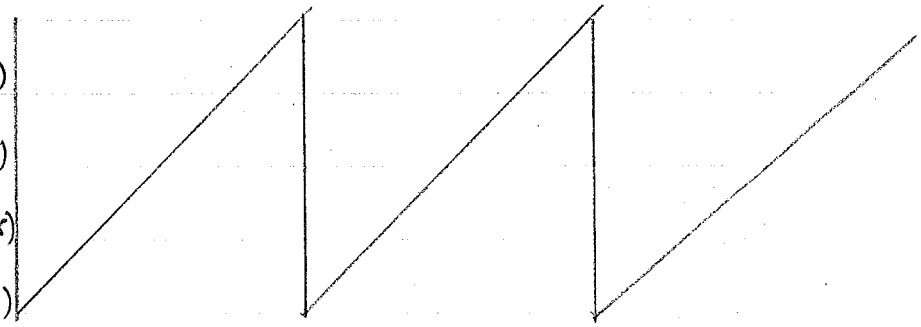


$\nu = .499$

y	EXACT					
.375	.6328	11.99	(1795)	11.99	11.99	11.99
.875	1.4765	9.966	(575)	9.966	9.966	9.966
1.375	2.3203	8.191	(253)	8.191	8.191	8.191
1.875	3.164	5.411	(71)	5.411	5.411	5.411

$\tau_{xy}(7,y)$ $\nu = .3$ (error in parent)

y	EXACT		
.375	-.3618	.1143	(132)
.875	-.3032	.1529	(150)
1.375	-.1978	.2275	(215)
1.875	-.0458	.3381	(838)



$\nu = .499$ (error in parenthesis)

y	EXACT					
.375	-.3618	-.2741	(24)	-.2741	-.2741	-.2741
.875	-.3032	-.2195	(28)	-.2195	-.2195	-.2195
1.375	-.1978	-.1693	(14)	-.1693	-.1693	-.1693
1.875	-.0458	-.1140	(149)	-.1140	-.1140	-.1140

3-node triangle $u_2(16,0)$ CROSS-DIAG.

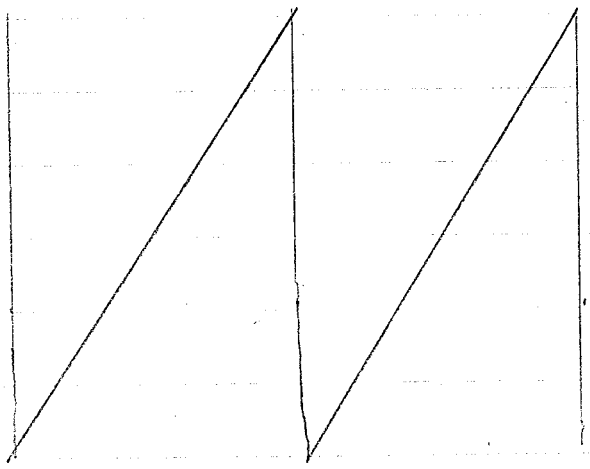
y	EXACT	2×2		2×2 w/ INCOMP	1×1 λ 2×2 μ	1×1 λ & μ w/ HOUR
.3	−244.14	−212.93	(13)			−212.93
.499	−205.47	−183.17	(11)	−183.17	−183.17	−183.17

$u_1(0,y)$ $\nu = .499$ (error in parenth)

y	EXACT					
0	0	0		0	0	0
.5	−.1317	−.3112	(136)	−.3112	−.3112	−.3112
1.0	−.2107	−.5195	(147)	−.5195	−.5195	−.5195
1.5	−.1843	−.5139	(178)	−.5139	−.5139	−.5139
2.0	0	0.		0	0	0
	STIFF & LOAD FORM.	.04		.04	.04	.04
	STIFF FACTOR/S	.04		.04	.04	.04

$\nu = .3$ (error in parenth)

y	EXACT			
0	0	0.		0.
0.5	−.1295	−.3960	(206)	−.3960
1.0	−.2072	−.6625	(220)	−.6625
1.5	−.1813	−.6304	(248)	−.6304
2.0	0	0.		0
	STIFF & LOAD FORM	0.0		0.06
	STIFF FACTOR/S	.06		.06



3-node triangle

CROSS-DIAG

$\sigma_x(7,y)$ $\nu = .3$ (error in parenth.)

y	EXACT	2x2	2x2 w/ INCOMP	1x1 λ 2x2 μ	1x1 λ&μ w/ HOUR
.125	.2109	.0537 (75)			.0537
.625	1.0547	.8006 (24)			.8006
1.125	1.8984	1.547 (19)			1.549
1.625	2.7422	2.294 (16)			2.294

$\nu = .499$ (error in parenthesis)

y	EXACT				
.125	.2109	.0798 (62)	.0798	.0798	.0798
.625	1.0547	.8737 (17)	.8737	.8737	.8737
1.125	1.8984	1.630 (14)	1.630	1.630	1.630
1.625	2.7422	2.415 (12)	2.415	2.415	2.415

$\tau_{xy}(7,y)$ $\nu = .3$

y	EXACT				
.125	-.3735	-.3521 (6)			-.3521
.625	-.3384	-.3077 (9)			-.3077
1.125	-.2564	-.2211 (14)			-.2211
1.625	-.1274	-.0931 (27)			-.0931

$\nu = .499$ (error in parenthesis)

y	EXACT				
.125	-.3735	-.3162 (15)	-.3162	-.3162	-.3162
.625	-.3384	-.2021 (40)	-.2021	-.2021	-.2021
1.125	-.2564	-.0354 (86)	-.0354	-.0354	-.0354
1.625	-.1274	.1722 (235)	.1722	.1722	.1722

CESAR LEVY ✓

ME 235B
Finite Element Methods
Spring Quarter 1981

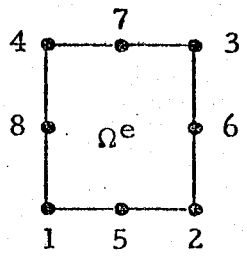
Stanford University

Instructor: T. Hughes

Homework Assignment
(due 1 week after handout)

Instructions: To cut down on computer costs, attempt to run all the data sets in problems 2 and 3 in one run.

1. Consider the 8-node isoparametric element shown below

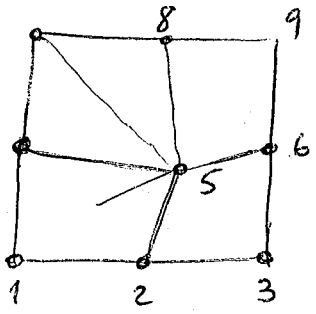


Assume the nodal coordinates are as follows:

a	x_a	y_a	
1	$-\frac{h}{2}$	$-\frac{h}{2}$	$\frac{h}{2}(-1,-1)$
2	$\frac{h}{2}$	$-\frac{h}{2}$	$\frac{h}{2}(1,-1)$
3	$\frac{h}{2}$	$\frac{h}{2}$	$\frac{h}{2}(1,1)$
4	$-\frac{h}{2}$	$\frac{h}{2}$	$\frac{h}{2}(-1,1)$
5	0	$-\frac{h}{2}$	$\frac{h}{2}(0,-1)$
6	$\frac{h}{2}$	0	$\frac{h}{2}(1,0)$
7	0	$\frac{h}{2}$	$\frac{h}{2}(0,1)$
8	$-\frac{h}{2}$	0	$\frac{h}{2}(-1,0)$

Consider the following body force

$$\underset{\sim}{f} = \begin{Bmatrix} 0 \\ -g \end{Bmatrix} \quad ; \qquad g \text{ const.}$$



1. PATCH TEST #1 D1A=1 D2A=0

COL	5	10	15	20	25	30	35	40	45
2	9	2	2	1	0	1	← execution; 0 for data check.		
3	1	1		0.0		0.0			
4	3	0		2.0		0.0			
5	4	0		0.0		1.0			
6	5	8		0.1		0.8			
7	6.	0		2.0		1.0			
	7	1		0.0		2.0			
	9	0		2.0		2.0			
0	blank								
1	1	1	1	1				~~Change~~ these	
2	4	0	1	1					
3	6	1	1	1					
4	9	0	1	1					
5	blank								
6	1	1		1.0		0.0		↑ Change these	
7	4	0		1.0		0.0			
8	6	1		1.0		0.0			
9	9	0		1.0		0.0		↓	
20	blank								
21	3	4	1	0	1	0	0	0	0
22	1			1.0		0.3		0.0	1.
23		0.0		0.0					
24	1	1	1	2	5	4	0		
25	2	1	2	3	6	5	0		
26	3	1	4	5	8	7	0		
27	4	1	5	6	9	8	0		
28	blank								

29	PATCH TEST #2	D1A=0	D2A=1			28		598
						28		626
30-43	Copy ~~of~~ Cards 2-15 here				86	30		656
44	1	1	0.0		116	30	1.0	686
45	4	0	0.0		146	30	1.0	716
46	6	1	0.0		176	30	1.0	746
47	9	0	0.0		190	14	1.0	760
						14		774
48-56	copy cards 20-28					14		788

or copy cards 2-28 FROM FILE1 to end & change in active cards 44-47

0.6709941E+01	0.9813007E+00	0.3446501E-03	0.1157379E-01	0.2901493E+04	1258	1
0.6714940E+01	0.9813573E+00	0.3430704E-03	0.1152086E-01	0.2914853E+04	1259	1
0.6719939E+01	0.9814137E+00	0.3414967E-03	0.1146936E-01	0.2928285E+04	1260	1
0.6724938E+01	0.9814693E+00	0.3399483E-03	0.1141739E-01	0.2941622E+04	1261	1
0.6729938E+01	0.9815241E+00	0.3384261E-03	0.1136684E-01	0.2954854E+04	1262	1
0.6734937E+01	0.9815784E+00	0.3369192E-03	0.1131630E-01	0.2968070E+04	1263	1
0.6739936E+01	0.9816331E+00	0.3354033E-03	0.1126623E-01	0.2981485E+04	1264	1
0.6744935E+01	0.9816880E+00	0.3338847E-03	0.1121569E-01	0.2995045E+04	1265	1
0.6749934E+01	0.9817426E+00	0.3323767E-03	0.1116514E-01	0.3008634E+04	1266	1
0.6754933E+01	0.9817963E+00	0.3308975E-03	0.1111555E-01	0.3022083E+04	1267	1
0.6759933E+01	0.9818500E+00	0.3294174E-03	0.1106644E-01	0.3035661E+04	1268	1
0.6764932E+01	0.9819034E+00	0.3279494E-03	0.1101732E-01	0.3049249E+04	1269	1
0.6769931E+01	0.9819567E+00	0.3264886E-03	0.1096869E-01	0.3062893E+04	1270	1
0.6774930E+01	0.9820101E+00	0.3250251E-03	0.1092005E-01	0.3076685E+04	1271	1
0.6779929E+01	0.9820637E+00	0.3235594E-03	0.1087141E-01	0.3090622E+04	1272	1
0.6784928E+01	0.9821170E+00	0.3221023E-03	0.1082277E-01	0.3104603E+04	1273	1
0.6789927E+01	0.9821693E+00	0.3206749E-03	0.1077509E-01	0.3118422E+04	1274	1
0.6794927E+01	0.9822209E+00	0.3192709E-03	0.1072836E-01	0.3132135E+04	1275	1
0.6799926E+01	0.9822721E+00	0.3178790E-03	0.1068163E-01	0.3145850E+04	1276	1
0.6804925E+01	0.9823236E+00	0.3164795E-03	0.1063538E-01	0.3159760E+04	1277	1
0.6809924E+01	0.9823753E+00	0.3150804E-03	0.1058817E-01	0.3173791E+04	1278	1
0.6814923E+01	0.9824267E+00	0.3136885E-03	0.1054192E-01	0.3187874E+04	1279	1
0.6819922E+01	0.9824782E+00	0.3122971E-03	0.1049519E-01	0.3202078E+04	1280	1
0.6824922E+01	0.9825287E+00	0.3109335E-03	0.1044989E-01	0.3216120E+04	1281	1
0.6829921E+01	0.9825793E+00	0.3095719E-03	0.1040411E-01	0.3230266E+04	1282	1
0.6834920E+01	0.9826295E+00	0.3082224E-03	0.1035929E-01	0.3244410E+04	1283	1
0.6839919E+01	0.9826795E+00	0.3068780E-03	0.1031494E-01	0.3258623E+04	1284	1
0.6844918E+01	0.9827298E+00	0.3055308E-03	0.1027012E-01	0.3272991E+04	1285	1
0.6849917E+01	0.9827802E+00	0.3041825E-03	0.1022530E-01	0.3287499E+04	1286	1
0.6854917E+01	0.9828303E+00	0.3028410E-03	0.1018047E-01	0.3302062E+04	1287	1
0.6859916E+01	0.9828795E+00	0.3015287E-03	0.1013708E-01	0.3316432E+04	1288	1
0.6864915E+01	0.9829279E+00	0.3002407E-03	0.1009369E-01	0.3330660E+04	1289	1
0.6869914E+01	0.9829759E+00	0.2989622E-03	0.1005125E-01	0.3344902E+04	1290	1
0.6874913E+01	0.9830242E+00	0.2976782E-03	0.1000834E-01	0.3359332E+04	1291	1
0.6879912E+01	0.9830728E+00	0.2963908E-03	0.9965420E-02	0.3373922E+04	1292	1
0.6884912E+01	0.9831210E+00	0.2951121E-03	0.9922028E-02	0.3388542E+04	1293	1
0.6889911E+01	0.9831693E+00	0.2938369E-03	0.9879589E-02	0.3403248E+04	1294	1
0.6894910E+01	0.9832166E+00	0.2925855E-03	0.9837627E-02	0.3417805E+04	1295	1
0.6899909E+01	0.9832640E+00	0.2913359E-03	0.9796143E-02	0.3432463E+04	1296	1
0.6904908E+01	0.9833110E+00	0.2900993E-03	0.9754658E-02	0.3447094E+04	1297	1
0.6909907E+01	0.9833579E+00	0.2888662E-03	0.9713173E-02	0.3461809E+04	1298	1
0.6914907E+01	0.9834050E+00	0.2876299E-03	0.9672642E-02	0.3476688E+04	1299	1
0.6919906E+01	0.9834521E+00	0.2863940E-03	0.9631157E-02	0.3491691E+04	1300	1
0.6924905E+01	0.9834992E+00	0.2851633E-03	0.9589672E-02	0.3506762E+04	1301	1
0.6929904E+01	0.9835462E+00	0.2839344E-03	0.9548664E-02	0.3521937E+04	1302	1
0.6934903E+01	0.9835923E+00	0.2827321E-03	0.9508610E-02	0.3536916E+04	1303	1
0.6939902E+01	0.9836375E+00	0.2815549E-03	0.9469032E-02	0.3551703E+04	1304	1
0.6944901E+01	0.9836822E+00	0.2803903E-03	0.9430408E-02	0.3566457E+04	1305	1
0.6949901E+01	0.9837274E+00	0.2792166E-03	0.9390831E-02	0.3581448E+04	1306	1
0.6954900E+01	0.9837728E+00	0.2780380E-03	0.9351254E-02	0.3596628E+04	1307	1
0.6959899E+01	0.9838179E+00	0.2768710E-03	0.9312153E-02	0.3611788E+04	1308	1
0.6964898E+01	0.9838629E+00	0.2757069E-03	0.9273529E-02	0.3627038E+04	1309	1
0.6969897E+01	0.9839081E+00	0.2745402E-03	0.9234428E-02	0.3642454E+04	1310	1
0.6974896E+01	0.9839523E+00	0.2733995E-03	0.9196281E-02	0.3657651E+04	1311	1
0.6979896E+01	0.9839965E+00	0.2722619E-03	0.9158134E-02	0.3672932E+04	1312	1
0.6984895E+01	0.9840403E+00	0.2711355E-03	0.9120464E-02	0.3688192E+04	1313	1
0.6989894E+01	0.9840840E+00	0.2700135E-03	0.9083271E-02	0.3703516E+04	1314	1
0.6994893E+01	0.9841279E+00	0.2688873E-03	0.9045124E-02	0.3719030E+04	1315	1
0.6999892E+01	0.9841719E+00	0.2677594E-03	0.9007454E-02	0.3734694E+04	1316	1
0.7004891E+01	0.9842157E+00	0.2666409E-03	0.8970261E-02	0.3750359E+04	1317	1

5 10 15 20 25 30 35 40 45

57. Patch Test #3 D1A=XA D2A=0

58-71 Copy Cards 2-15

72	1	1	0.0	0.0
73	3	0	2.0	0.0
74	4	0	0.0	0.0
75	6	0	2.0	0.0
~~78-84~~		~~Copy Cards 16-28~~		
76	7	1	0.0	0.0
77	9	0	2.0	0.0

78-86 Copy cards 20-28

87 Patch Test #4 D1A=0 D2A=XA

88-101 Copy Cards 2-15

102	1	1	0.0	0.0
103	3	0	0.0	2.0
104	4	0	0.0	0.0
105	6	0	0.0	2.0
106	7	1	0.0	0.0
107	9	0	0.0	2.0

108-116 Copy cards 20-28

117 Patch Test #5 D1A=YA D2A=0

118-131 Copy Cards 2-15

132	1	1	0.0	0.0
133	3	0	0.0	0.0
134	4	0	1.0	0.0
135	6	0	1.0	0.0
136	7	1	2.0	0.0
137	9	0	2.0	0.0

138-146 copy cards 20-28

147 Patch Test #6 D1A=0 D2A=YA

148-161 Copy cards 2-15

162	1	1	0.0	0.0
163	3	0	0.0	0.0
164	4	0	0.0	1.0
165	6	0	0.0	1.0
166	7	1	0.0	2.0
167	9	0	0.0	2.0

168-176 Copy cards 20-28

177 Rank Check run 1x1 L 2x2 M

178	4	2	2	1	1	0 executio
179	1	0		0.0		0.0
180	2	0		1.0		0.0
181	3	0		1.0		1.0
182	4	0		0.0		1.0
183	blank					
184	blank	no boundary				
185	blank					

0.6409991E+01	0.9773899E+00	0.4599683E-03	0.1539373E-01	0.2174062E+04	1198	1
0.6414990E+01	0.9774655E+00	0.4576293E-03	0.1531601E-01	0.2185174E+04	1199	1
0.6419990E+01	0.9775403E+00	0.4553176E-03	0.1524019E-01	0.2196268E+04	1200	1
0.6424989E+01	0.9776141E+00	0.4530386E-03	0.1516438E-01	0.2207317E+04	1201	1
0.6429988E+01	0.9776884E+00	0.4507517E-03	0.1508999E-01	0.2218515E+04	1202	1
0.6434987E+01	0.9777621E+00	0.4484854E-03	0.1501465E-01	0.2229727E+04	1203	1
0.6439986E+01	0.9778346E+00	0.4462593E-03	0.1494122E-01	0.2240850E+04	1204	1
0.6444985E+01	0.9779065E+00	0.4440576E-03	0.1486874E-01	0.2251959E+04	1205	1
0.6449985E+01	0.9779782E+00	0.4418669E-03	0.1479626E-01	0.2263125E+04	1206	1
0.6454984E+01	0.9780501E+00	0.4396727E-03	0.1472378E-01	0.2274419E+04	1207	1
0.6459983E+01	0.9781220E+00	0.4374811E-03	0.1465082E-01	0.2285813E+04	1208	1
0.6464982E+01	0.9781929E+00	0.4353262E-03	0.1458025E-01	0.2297127E+04	1209	1
0.6469981E+01	0.9782637E+00	0.4331749E-03	0.1450968E-01	0.2308536E+04	1210	1
0.6474980E+01	0.9783342E+00	0.4310396E-03	0.1443863E-01	0.2319972E+04	1211	1
0.6479980E+01	0.9784046E+00	0.4289087E-03	0.1436853E-01	0.2331498E+04	1212	1
0.6484979E+01	0.9784744E+00	0.4268032E-03	0.1429987E-01	0.2342999E+04	1213	1
0.6489978E+01	0.9785440E+00	0.4247047E-03	0.1422977E-01	0.2354576E+04	1214	1
0.6494977E+01	0.9786132E+00	0.4226228E-03	0.1416111E-01	0.2366176E+04	1215	1
0.6499976E+01	0.9786824E+00	0.4205445E-03	0.1409292E-01	0.2377869E+04	1216	1
0.6504975E+01	0.9787517E+00	0.4184663E-03	0.1402378E-01	0.2389678E+04	1217	1
0.6509974E+01	0.9788201E+00	0.4164202E-03	0.1395607E-01	0.2401420E+04	1218	1
0.6514974E+01	0.9788876E+00	0.4144046E-03	0.1388931E-01	0.2413100E+04	1219	1
0.6519973E+01	0.9789554E+00	0.4123796E-03	0.1382208E-01	0.2424949E+04	1220	1
0.6524972E+01	0.9790229E+00	0.4103710E-03	0.1375532E-01	0.2436818E+04	1221	1
0.6529971E+01	0.9790900E+00	0.4083766E-03	0.1368952E-01	0.2448719E+04	1222	1
0.6534970E+01	0.9791565E+00	0.4064050E-03	0.1362419E-01	0.2460599E+04	1223	1
0.6539969E+01	0.9792231E+00	0.4044331E-03	0.1355886E-01	0.2472596E+04	1224	1
0.6544969E+01	0.9792891E+00	0.4024825E-03	0.1349449E-01	0.2484580E+04	1225	1
0.6549968E+01	0.9793550E+00	0.4005372E-03	0.1343012E-01	0.2496647E+04	1226	1
0.6554967E+01	0.9794213E+00	0.3985846E-03	0.1336527E-01	0.2508877E+04	1227	1
0.6559966E+01	0.9794865E+00	0.3966666E-03	0.1330185E-01	0.2521009E+04	1228	1
0.6564965E+01	0.9795516E+00	0.3947557E-03	0.1323843E-01	0.2533212E+04	1229	1
0.6569964E+01	0.9796163E+00	0.3928586E-03	0.1317549E-01	0.2545445E+04	1230	1
0.6574964E+01	0.9796811E+00	0.3909629E-03	0.1311207E-01	0.2557786E+04	1231	1
0.6579963E+01	0.9797460E+00	0.3890651E-03	0.1304913E-01	0.2570263E+04	1232	1
0.6584962E+01	0.9798099E+00	0.3872043E-03	0.1298761E-01	0.2582615E+04	1233	1
0.6589961E+01	0.9798728E+00	0.3853738E-03	0.1292658E-01	0.2594882E+04	1234	1
0.6594960E+01	0.9799354E+00	0.3835538E-03	0.1286650E-01	0.2607196E+04	1235	1
0.6599959E+01	0.9799985E+00	0.3817240E-03	0.1280594E-01	0.2619693E+04	1236	1
0.6604959E+01	0.9800616E+00	0.3798965E-03	0.1274538E-01	0.2632296E+04	1237	1
0.6609958E+01	0.9801235E+00	0.3781074E-03	0.1268578E-01	0.2644750E+04	1238	1
0.6614957E+01	0.9801844E+00	0.3763479E-03	0.1262712E-01	0.2657115E+04	1239	1
0.6619956E+01	0.9802451E+00	0.3745994E-03	0.1256943E-01	0.2669518E+04	1240	1
0.6624955E+01	0.9803063E+00	0.3728401E-03	0.1251030E-01	0.2682114E+04	1241	1
0.6629954E+01	0.9803676E+00	0.3710790E-03	0.1245213E-01	0.2694844E+04	1242	1
0.6634953E+01	0.9804275E+00	0.3693614E-03	0.1239538E-01	0.2707375E+04	1243	1
0.6639953E+01	0.9804875E+00	0.3676452E-03	0.1233864E-01	0.2720013E+04	1244	1
0.6644952E+01	0.9805473E+00	0.3659355E-03	0.1228189E-01	0.2732721E+04	1245	1
0.6649951E+01	0.9806069E+00	0.3642347E-03	0.1222563E-01	0.2745481E+04	1246	1
0.6654950E+01	0.9806667E+00	0.3625322E-03	0.1216888E-01	0.2758375E+04	1247	1
0.6659949E+01	0.9807265E+00	0.3608309E-03	0.1211214E-01	0.2771381E+04	1248	1
0.6664948E+01	0.9807853E+00	0.3591627E-03	0.1205683E-01	0.2784253E+04	1249	1
0.6669948E+01	0.9808431E+00	0.3575238E-03	0.1200247E-01	0.2797015E+04	1250	1
0.6674947E+01	0.9809006E+00	0.3558965E-03	0.1194859E-01	0.2809804E+04	1251	1
0.6679946E+01	0.9809586E+00	0.3542581E-03	0.1189375E-01	0.2822799E+04	1252	1
0.6684945E+01	0.9810167E+00	0.3526211E-03	0.1183939E-01	0.2835904E+04	1253	1
0.6689944E+01	0.9810744E+00	0.3509966E-03	0.1178503E-01	0.2849030E+04	1254	1
0.6694943E+01	0.9811312E+00	0.3493996E-03	0.1173162E-01	0.2862051E+04	1255	1
0.6699943E+01	0.9811873E+00	0.3478257E-03	0.1167917E-01	0.2875002E+04	1256	1
0.6704942E+01	0.9812438E+00	0.3462411E-03	0.1162720E-01	0.2888161E+04	1257	1

	5	10	15	20	25	30	55	40	45	
186	5	1	1	0	1	0	0	0	0	
187	1			1.0		0.3		0.0		
188		0.0		0.0						
189	1	1	1	2	3	4	0			
190	blank									
191	Rank Check Run for 1x1 K and 1x1 M ~~[illegible]~~									
192-199	copy 178 to 187 here									
200	3	1	1	0	1	1	0	0	0	X
201-204	copy 189-192 here									
205	Rank Check Run for 1x1 L and 1x1 M w/stiffness									
206-213	copy 178 to 187 here									
214	3	1	1	0	1	1	0	1	0	X
215-218	copy 189-192 here									
219	blank									

or copy 178-192 from file 1 & change 216 of active file

0.6110042E+01	0.9720769E+00	0.6352291E-03	0.2111387E-01	0.1574235E+04	1138	1
0.6115041E+01	0.9721808E+00	0.6315990E-03	0.2099705E-01	0.1583283E+04	1139	1
0.6120040E+01	0.9722841E+00	0.6279999E-03	0.2088070E-01	0.1592357E+04	1140	1
0.6125039E+01	0.9723867E+00	0.6244357E-03	0.2076530E-01	0.1601446E+04	1141	1
0.6130038E+01	0.9724877E+00	0.6209309E-03	0.2065182E-01	0.1610485E+04	1142	1
0.6135037E+01	0.9725881E+00	0.6174541E-03	0.2053928E-01	0.1619553E+04	1143	1
0.6140037E+01	0.9726884E+00	0.6139888E-03	0.2042723E-01	0.1628694E+04	1144	1
0.6145036E+01	0.9727880E+00	0.6105558E-03	0.2031612E-01	0.1637852E+04	1145	1
0.6150035E+01	0.9728876E+00	0.6071297E-03	0.2020502E-01	0.1647094E+04	1146	1
0.6155034E+01	0.9729866E+00	0.6037317E-03	0.2009487E-01	0.1656365E+04	1147	1
0.6160033E+01	0.9730846E+00	0.6003759E-03	0.1998615E-01	0.1665623E+04	1148	1
0.6165032E+01	0.9731821E+00	0.5970427E-03	0.1987791E-01	0.1674921E+04	1149	1
0.6170032E+01	0.9732801E+00	0.5937030E-03	0.1976967E-01	0.1684343E+04	1150	1
0.6175031E+01	0.9733773E+00	0.5903945E-03	0.1966238E-01	0.1693782E+04	1151	1
0.6180030E+01	0.9734742E+00	0.5871027E-03	0.1955509E-01	0.1703279E+04	1152	1
0.6185029E+01	0.9735706E+00	0.5838363E-03	0.1944923E-01	0.1712809E+04	1153	1
0.6190028E+01	0.9736662E+00	0.5806035E-03	0.1934385E-01	0.1722346E+04	1154	1
0.6195027E+01	0.9737611E+00	0.5773988E-03	0.1923990E-01	0.1731905E+04	1155	1
0.6200027E+01	0.9738554E+00	0.5742230E-03	0.1913643E-01	0.1741483E+04	1156	1
0.6205026E+01	0.9739492E+00	0.5710719E-03	0.1903391E-01	0.1751093E+04	1157	1
0.6210025E+01	0.9740423E+00	0.5679494E-03	0.1893234E-01	0.1760720E+04	1158	1
0.6215024E+01	0.9741356E+00	0.5648250E-03	0.1883030E-01	0.1770460E+04	1159	1
0.6220023E+01	0.9742275E+00	0.5617575E-03	0.1873064E-01	0.1780127E+04	1160	1
0.6225022E+01	0.9743191E+00	0.5587034E-03	0.1863098E-01	0.1789858E+04	1161	1
0.6230021E+01	0.9744106E+00	0.5556599E-03	0.1853180E-01	0.1799662E+04	1162	1
0.6235021E+01	0.9745012E+00	0.5526519E-03	0.1843357E-01	0.1809457E+04	1163	1
0.6240020E+01	0.9745909E+00	0.5496799E-03	0.1833677E-01	0.1819240E+04	1164	1
0.6245019E+01	0.9746802E+00	0.5467280E-03	0.1824045E-01	0.1829062E+04	1165	1
0.6250018E+01	0.9747699E+00	0.5437676E-03	0.1814365E-01	0.1839021E+04	1166	1
0.6255017E+01	0.9748588E+00	0.5408411E-03	0.1804829E-01	0.1848971E+04	1167	1
0.6260016E+01	0.9749474E+00	0.5379280E-03	0.1795292E-01	0.1858985E+04	1168	1
0.6265016E+01	0.9750355E+00	0.5350364E-03	0.1785851E-01	0.1869031E+04	1169	1
0.6270015E+01	0.9751229E+00	0.5321763E-03	0.1776505E-01	0.1879076E+04	1170	1
0.6275014E+01	0.9752095E+00	0.5293470E-03	0.1767254E-01	0.1889119E+04	1171	1
0.6280013E+01	0.9752956E+00	0.5265404E-03	0.1758051E-01	0.1899189E+04	1172	1
0.6285012E+01	0.9753821E+00	0.5237262E-03	0.1748848E-01	0.1909394E+04	1173	1
0.6290011E+01	0.9754679E+00	0.5209409E-03	0.1739740E-01	0.1919603E+04	1174	1
0.6295011E+01	0.9755533E+00	0.5181725E-03	0.1730680E-01	0.1929859E+04	1175	1
0.6300010E+01	0.9756383E+00	0.5154209E-03	0.1721668E-01	0.1940161E+04	1176	1
0.6305009E+01	0.9757224E+00	0.5127059E-03	0.1712751E-01	0.1950436E+04	1177	1
0.6310008E+01	0.9758059E+00	0.5100169E-03	0.1703930E-01	0.1960719E+04	1178	1
0.6315007E+01	0.9758888E+00	0.5073508E-03	0.1695204E-01	0.1971022E+04	1179	1
0.6320006E+01	0.9759721E+00	0.5046758E-03	0.1686430E-01	0.1981470E+04	1180	1
0.6325006E+01	0.9760547E+00	0.5020292E-03	0.1677752E-01	0.1991916E+04	1181	1
0.6330005E+01	0.9761370E+00	0.4993996E-03	0.1669121E-01	0.2002404E+04	1182	1
0.6335004E+01	0.9762187E+00	0.4967914E-03	0.1660538E-01	0.2012917E+04	1183	1
0.6340003E+01	0.9762997E+00	0.4942114E-03	0.1652098E-01	0.2023425E+04	1184	1
0.6345002E+01	0.9763801E+00	0.4916571E-03	0.1643705E-01	0.2033938E+04	1185	1
0.6350001E+01	0.9764607E+00	0.4890987E-03	0.1635313E-01	0.2044577E+04	1186	1
0.6355000E+01	0.9765408E+00	0.4865618E-03	0.1626968E-01	0.2055237E+04	1187	1
0.6360000E+01	0.9766197E+00	0.4840686E-03	0.1618767E-01	0.2065822E+04	1188	1
0.6364999E+01	0.9766979E+00	0.4816016E-03	0.1610661E-01	0.2076404E+04	1189	1
0.6369998E+01	0.9767759E+00	0.4791424E-03	0.1602554E-01	0.2087062E+04	1190	1
0.6374997E+01	0.9768541E+00	0.4766847E-03	0.1594496E-01	0.2097822E+04	1191	1
0.6379996E+01	0.9769315E+00	0.4742593E-03	0.1586485E-01	0.2108551E+04	1192	1
0.6384995E+01	0.9770087E+00	0.4718404E-03	0.1578522E-01	0.2119360E+04	1193	1
0.6389995E+01	0.9770855E+00	0.4694390E-03	0.1570654E-01	0.2130202E+04	1194	1
0.6394994E+01	0.9771623E+00	0.4670441E-03	0.1562738E-01	0.2141125E+04	1195	1
0.6399993E+01	0.9772384E+00	0.4646734E-03	0.1554871E-01	0.2152049E+04	1196	1
0.6404992E+01	0.9773144E+00	0.4623113E-03	0.1547050E-01	0.2163046E+04	1197	1

0.5810092E+01	0.9646033E+00	0.9170666E-03	0.3002739E-01	0.1090433E+04	1078	1
0.5815091E+01	0.9647506E+00	0.9111117E-03	0.2984333E-01	0.1097560E+04	1079	1
0.5820090E+01	0.9648976E+00	0.9051857E-03	0.2965975E-01	0.1104746E+04	1080	1
0.5825089E+01	0.9650435E+00	0.8993191E-03	0.2947807E-01	0.1111952E+04	1081	1
0.5830089E+01	0.9651893E+00	0.8934708E-03	0.2929687E-01	0.1119230E+04	1082	1
0.5835088E+01	0.9653339E+00	0.8876899E-03	0.2911711E-01	0.1126520E+04	1083	1
0.5840087E+01	0.9654775E+00	0.8819643E-03	0.2893925E-01	0.1133833E+04	1084	1
0.5845086E+01	0.9656199E+00	0.8762968E-03	0.2876282E-01	0.1141166E+04	1085	1
0.5850085E+01	0.9657609E+00	0.8707053E-03	0.2858877E-01	0.1148494E+04	1086	1
0.5855084E+01	0.9659015E+00	0.8651423E-03	0.2841520E-01	0.1155879E+04	1087	1
0.5860084E+01	0.9660410E+00	0.8596387E-03	0.2824354E-01	0.1163279E+04	1088	1
0.5865083E+01	0.9661805E+00	0.8541460E-03	0.2807188E-01	0.1170760E+04	1089	1
0.5870082E+01	0.9663188E+00	0.8487189E-03	0.2790260E-01	0.1178246E+04	1090	1
0.5875081E+01	0.9664569E+00	0.8433093E-03	0.2773333E-01	0.1185804E+04	1091	1
0.5880080E+01	0.9665939E+00	0.8379607E-03	0.2756596E-01	0.1193373E+04	1092	1
0.5885079E+01	0.9667299E+00	0.8326659E-03	0.2740002E-01	0.1200962E+04	1093	1
0.5890079E+01	0.9668646E+00	0.8274305E-03	0.2723551E-01	0.1208561E+04	1094	1
0.5895078E+01	0.9669989E+00	0.8222293E-03	0.2707243E-01	0.1216206E+04	1095	1
0.5900077E+01	0.9671320E+00	0.8170852E-03	0.2691078E-01	0.1223863E+04	1096	1
0.5905076E+01	0.9672652E+00	0.8119494E-03	0.2674913E-01	0.1231604E+04	1097	1
0.5910075E+01	0.9673971E+00	0.8068776E-03	0.2658939E-01	0.1239345E+04	1098	1
0.5915074E+01	0.9675288E+00	0.8018252E-03	0.2643061E-01	0.1247155E+04	1099	1
0.5920074E+01	0.9676595E+00	0.7968273E-03	0.2627277E-01	0.1254977E+04	1100	1
0.5925073E+01	0.9677891E+00	0.7918798E-03	0.2611685E-01	0.1262818E+04	1101	1
0.5930072E+01	0.9679175E+00	0.7869932E-03	0.2596235E-01	0.1270659E+04	1102	1
0.5935071E+01	0.9680454E+00	0.7821361E-03	0.2580881E-01	0.1278550E+04	1103	1
0.5940070E+01	0.9681721E+00	0.7773379E-03	0.2565718E-01	0.1286441E+04	1104	1
0.5945069E+01	0.9682989E+00	0.7725458E-03	0.2550554E-01	0.1294421E+04	1105	1
0.5950068E+01	0.9684246E+00	0.7678100E-03	0.2535582E-01	0.1302405E+04	1106	1
0.5955068E+01	0.9685499E+00	0.7630971E-03	0.2520609E-01	0.1310449E+04	1107	1
0.5960067E+01	0.9686741E+00	0.7584379E-03	0.2505875E-01	0.1318499E+04	1108	1
0.5965066E+01	0.9687974E+00	0.7538239E-03	0.2491188E-01	0.1326570E+04	1109	1
0.5970065E+01	0.9689195E+00	0.7492674E-03	0.2476740E-01	0.1334637E+04	1110	1
0.5975064E+01	0.9690410E+00	0.7447423E-03	0.2462339E-01	0.1342746E+04	1111	1
0.5980063E+01	0.9691613E+00	0.7402746E-03	0.2448130E-01	0.1350850E+04	1112	1
0.5985063E+01	0.9692808E+00	0.7358456E-03	0.2434063E-01	0.1358980E+04	1113	1
0.5990062E+01	0.9694002E+00	0.7314347E-03	0.2419996E-01	0.1367176E+04	1114	1
0.5995061E+01	0.9695188E+00	0.7270572E-03	0.2406073E-01	0.1375407E+04	1115	1
0.6000060E+01	0.9696375E+00	0.7226928E-03	0.2392149E-01	0.1383713E+04	1116	1
0.6005059E+01	0.9697547E+00	0.7183896E-03	0.2378416E-01	0.1392002E+04	1117	1
0.6010058E+01	0.9698718E+00	0.7141004E-03	0.2364731E-01	0.1400363E+04	1118	1
0.6015058E+01	0.9699879E+00	0.7098601E-03	0.2351189E-01	0.1408728E+04	1119	1
0.6020057E+01	0.9701038E+00	0.7056340E-03	0.2337646E-01	0.1417165E+04	1120	1
0.6025056E+01	0.9702194E+00	0.7014291E-03	0.2324200E-01	0.1425660E+04	1121	1
0.6030055E+01	0.9703341E+00	0.6972684E-03	0.2310896E-01	0.1434168E+04	1122	1
0.6035054E+01	0.9704480E+00	0.6931475E-03	0.2297688E-01	0.1442694E+04	1123	1
0.6040053E+01	0.9705605E+00	0.6890842E-03	0.2284670E-01	0.1451201E+04	1124	1
0.6045053E+01	0.9706724E+00	0.6850502E-03	0.2271748E-01	0.1459747E+04	1125	1
0.6050052E+01	0.9707841E+00	0.6810352E-03	0.2258873E-01	0.1468353E+04	1126	1
0.6055051E+01	0.9708951E+00	0.6770543E-03	0.2246094E-01	0.1476986E+04	1127	1
0.6060050E+01	0.9710053E+00	0.6731122E-03	0.2233410E-01	0.1485636E+04	1128	1
0.6065049E+01	0.9711148E+00	0.6692007E-03	0.2220821E-01	0.1494320E+04	1129	1
0.6070048E+01	0.9712239E+00	0.6653152E-03	0.2208376E-01	0.1503047E+04	1130	1
0.6075047E+01	0.9713331E+00	0.6614316E-03	0.2195883E-01	0.1511872E+04	1131	1
0.6080047E+01	0.9714408E+00	0.6576139E-03	0.2183580E-01	0.1520649E+04	1132	1
0.6085046E+01	0.9715483E+00	0.6538110E-03	0.2171326E-01	0.1529493E+04	1133	1
0.6090045E+01	0.9716549E+00	0.6500462E-03	0.2159214E-01	0.1538352E+04	1134	1
0.6095044E+01	0.9717613E+00	0.6462960E-03	0.2147102E-01	0.1547278E+04	1135	1
0.6100043E+01	0.9718674E+00	0.6425686E-03	0.2135086E-01	0.1556254E+04	1136	1
0.6105042E+01	0.9719725E+00	0.6388831E-03	0.2123213E-01	0.1565231E+04	1137	1

Compute (by hand) the nodal forces for nodes 1 and 5, i.e.

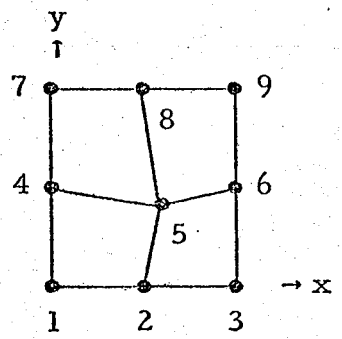
$$f^e_{21} = \int_{\Omega^e} f_2 N_1 \, d\Omega$$

$$f^e_{25} = \int_{\Omega^e} f_2 N_5 \, d\Omega$$

(Note that $f^e_{1a} = 0$, $1 \le a \le 8$, and, by symmetry, $f^e_{21} = f^e_{22} = f^e_{23} = f^e_{24}$ and $f^e_{25} = f^e_{26} = f^e_{27} = f^e_{28}$.)

The results of this simple computation should be somewhat surprising. Comment.

2. Consider the "patch" of elements shown below.



The nodal coordinates are:

a	x_a	y_a
1	0	0
2	1	0
3	2	0
4	0	1
5	1.1	.8
6	2	1
7	0	2
8	1	2
9	2	2

Use LEARN to execute the following displacement boundary value problems ($1 \le a \le 4$, $6 \le a \le 9$):

test #	d_{1a}	d_{2a}
1	1	0
2	0	1
3	x_a	0
4	0	x_a
5	y_a	0
6	0	y_a

Use the following data:

- plane strain option
- $E = 1.$, $\nu = .3$

If the first letter of your last name is

LK	A-H I-R S-Z	use the	2 × 2 quad. (1-pt. λ term 2 × 2 μ term) incomp. modes	element.

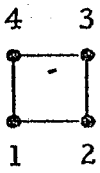
Based upon the results you obtain, determine whether or not your element passes the patch test.

(Consider results "exact", if they are correct to 8, or more, significant digits. Due to round-off errors, one cannot expect truly exact results on digital computers.)

3. The "rank check" option in the LEARN can be used to determine the number of zero eigenvalues of the global stiffness matrix $\underset{\sim}{K}$. That is, the diagonal elements of the diagonal matrix $\underset{\sim}{D}$ in the factorization

$\underset{\sim}{K} = \underset{\sim}{U}^T \underset{\sim}{D} \underset{\sim}{U}$, where $\underset{\sim}{U}$ is an upper triangular matrix, are printed. The number of zeroes in $\underset{\sim}{D}$ equals the number of zero eigenvalues of $\underset{\sim}{K}$, which corresponds to the number of zero-energy modes of the structure in question. Normally, there are, at most, 3 zero-energy modes -- the rigid body modes.

Consider the single element shown below



a	x_a	y_a
1	0	0
2	1	0
3	1	1
4	0	1

Assume the data is the same as in problem 2, and there are no boundary conditions.

Perform rank check analyses for

(a) the element you used in problem 2. $1\times1\ \lambda\ \ 2\times2\ \mu$

(b) 1-pt. on λ term and 1-pt. on μ term element. $1\times1\ \lambda+\mu$

Comment on your results.

1. $\underset{\sim}{f} = \begin{Bmatrix} 0 \\ -g \end{Bmatrix}$ $\quad f_p^e = \int N_a f_i \, d\Omega_e \quad p = ned(a-1)+i \quad ned=2 \quad i=1,2 \quad 1 \le a \le 8$

since $f_1 = 0 \Rightarrow \forall$ all odd $p \quad f_p^e = 0$

$$\therefore \quad f_{2a}^e = \int f_2 N_a \, d\Omega_e = \int_{-1}^{1}\int_{-1}^{1} f_2 N_a \cdot \frac{h^2}{4} \, d\xi d\eta$$

$$f_{2a}^e = -\frac{gh^2}{4} \int_{-1}^{1}\int_{-1}^{1} N_a \, d\xi d\eta$$

$N_1 = -\frac{1}{4}(1-\xi)(1-\eta)[1+\xi+\eta]$ $\qquad N_5 = \frac{1}{2}(1-\xi^2)(1-\eta)$ $\qquad$ where $N_i = N_{i\ \text{4 nodq quad}} - \frac{1}{2}(N_{i+1}+N_{i+2})$

$N_2 = -\frac{1}{4}(1+\xi)(1-\eta)[1-\xi+\eta]$ $\qquad N_6 = \frac{1}{2}(1+\xi)(1-\eta^2)$ $\qquad i=1,2,3,4$ —

$N_3 = -\frac{1}{4}(1+\xi)(1+\eta)[1-\xi-\eta]$ $\qquad N_7 = \frac{1}{2}(1-\xi^2)(1+\eta)$

$N_4 = -\frac{1}{4}(1-\xi)(1+\eta)[1+\xi-\eta]$ $\qquad N_8 = \frac{1}{2}(1-\eta^2)(1+\xi)$

$$\iint N_1 dA = -\frac{1}{4}\int_{-1}^{1}(1-\xi)d\xi \int_{-1}^{1}[(1-\eta)\xi + (1-\eta^2)]d\eta$$

$$\left(\xi \cdot 2\eta\Big|_{\eta=1} + 2\left(\eta - \eta^3/3\right)\Big|_{\eta=1}\right) = -\frac{1}{4}\int_{-1}^{1}(1-\xi)\cdot 2\left(\xi + \tfrac{2}{3}\right)d\xi = \frac{1}{2}\int_{-1}^{1}(\xi-1)(\xi+\tfrac{2}{3})d\xi$$

$$= \frac{1}{2}2\left(\xi^3/3 - \tfrac{2}{3}\xi\right)\Big|_{\xi=1} = -\tfrac{1}{3}$$

$$\therefore \quad f_{2a}^e = gh^2/12 \qquad \text{for} \quad a = 1,2,3,4$$

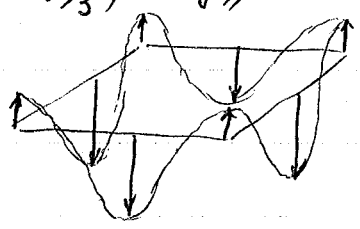
if for N_2 we let $-\xi' = +\xi$ then we get same integral ; for N_3 let $-\xi' = \xi$, $-\eta' = \eta$; for N_4 let $-\eta' = \eta$

$$\iint N_5 = \frac{1}{2}\int_{-1}^{1} d\xi (1-\xi^2)\int_{-1}^{1}(1-\eta)\,d\eta = \frac{1}{2}\cdot 2\left(\xi - \xi^3/3\right)_{\xi=1}\cdot 2\cdot\eta\Big|_{\eta=1} = 2\cdot\tfrac{2}{3} = 4/3$$

$$\therefore \quad f_{2a}^e = -gh^2/3 \qquad a = 5,6,7,8 \ .$$ Again same trick as before shows equality of integrals. Now $\sum_{a=1}^{8} N_a = 1$

$$\therefore \quad \Sigma f_{2a}^e = \Sigma \int_{\Omega_e} f_2 N_a \, d\Omega_e = \int f_2 \cdot 1 \, d\Omega_e = f_2 \Omega_e = -gh^2$$

Now $\quad 4 \cdot gh^2/12 + 4\left(-gh^2/3\right) = -gh^2$ // $= \Sigma f_{2a}^e$



expect a distribution all of which $= \left|gh^2/8\right|$ & are negative (true for linear case but not quadratic).

2. The patch test was run for the 1×1 λ 2×2 μ element using as data $E = 1.0$ $\nu = .3$ with the plane strain option. Below are the results of tests 1 thru 6

TEST 1 $D_{1a} = 1$ $D_{2a} = 0$

D I S P L A C E M E N T S

NODE NUMBER	DOF 1	DOF 2
1	1.000000000D+00	0.0
2	1.000000000D+00	0.0
3	1.000000000D+00	0.0
4	1.000000000D+00	0.0
5	1.000000000D+00	-4.01290290D-19
6	1.000000000D+00	0.0
7	1.000000000D+00	0.0
8	1.000000000D+00	0.0
9	1.000000000D+00	0.0

TEST 2 $D_{1a} = 0$ $D_{2a} = 1$

D I S P L A C E M E N T S

NODE NUMBER	DOF 1	DOF 2
1	0.0	1.000000000D+00
2	0.0	1.000000000D+00
3	0.0	1.000000000D+00
4	0.0	1.000000000D+00
5	1.90819582D-17	1.000000000D+00
6	0.0	1.000000000D+00
7	0.0	1.000000000D+00
8	0.0	1.000000000D+00
9	0.0	1.000000000D+00

TEST 3 $D_{1a} = X_a$ $D_{2a} = 0$

D I S P L A C E M E N T S

NODE NUMBER	DOF 1	DOF 2
1	0.0	0.0
2	1.000000000D+00	0.0
3	2.000000000D+00	0.0
4	0.0	0.0
5	1.100000000D+00	-3.77212872D-17
6	2.000000000D+00	0.0
7	0.0	0.0
8	1.000000000D+00	0.0
9	2.000000000D+00	0.0

TEST 4 $D_{1a} = 0$ $D_{2a} = X_a$

D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2
1	0.0	0.0
2	0.0	1.00000000D+00
3	0.0	2.00000000D+00
4	0.0	0.0
5	-6.93889390D-18	1.10000000D+00
6	0.0	2.00000000D+00
7	0.0	0.0
8	0.0	1.00000000D+00
9	0.0	2.00000000D+00

TEST 5 $D_{1a} = Y_a$ $D_{2a} = 0$

D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	1.00000000D+00	0.0
5	8.00000000D-01	2.40774174D-17
6	1.00000000D+00	0.0
7	2.00000000D+00	0.0
8	2.00000000D+00	0.0
9	2.00000000D+00	0.0

TEST 6 $D_{1a} = 0$ $D_{2a} = Y_a$

D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	1.00000000D+00
5	4.33680869D-17	8.00000000D-01
6	0.0	1.00000000D+00
7	0.0	2.00000000D+00
8	0.0	2.00000000D+00
9	0.0	2.00000000D+00

We note that the element passed the patch test with flying "colors".

3. Below are the elements of $\underset{\sim}{D}$ for the 1×1 λ, 2×2 μ case as well as those for the 1×1 λ & μ without and with the hourglass modes.

1×1 λ
2×2 μ

ARRAY D(NEQ) OF FACTORIZATI

1	5.28846154D-01
2	4.19580420D-01
3	2.88461538D-01
4	2.88461538D-01
5	2.84900285D-01
6	-2.70400564D-33
7	1.11289174D-03
8	-3.97598621D-14

Based on the results of this run we show that there are at most two energy modes (the 10^{-33} and 10^{-14} entries)

1×1 λ & μ
no hourglass

ARRAY D(NEQ) OF FACTORIZATI

1	4.32692308D-01
2	2.99145299D-01
3 *	2.74725275D-01
4 *	6.93889390D-17
5	-1.73472348D-17
6	1.94289029D-17
7	-2.04836768D-17
8	-2.89698820D-17

Based on the results of this run we show what appears to be 5 zero eigenvalues (all entries that are 10^{-17}). Because the hourglass options were not used we question the results. If we only use the negative entries then we show the three energy modes, but this method of picking the eigenvalues is dangero

1×1 λ & μ
w/ hourglass

ARRAY D(NEQ) OF FACTORIZATI

1	5.24267399D-01
2	4.14047379D-01
3	2.87885767D-01
4	2.77915633D-01
5 *	2.74725275D-01
6 *	6.93889390D-17
7	-1.06783072D-16
8	5.45391847D-17

Based on the results of this run we show at most three zero entries (10^{-16}, 10^{-17}). However note the starred entries. These entries lead me to believe that such numbers ~~coincidence~~ may not be accidental and that 6.94×10^{-17} is a non zero entry, thus reducing the no. of zero entries to two (same as the 1×1 λ 2×2 μ case). Also note entries here as very similar to the first run.

Hence we can say that the 1x1 λ & μ can require the use of the hourglass option and that the higher modes agree better with the 1x1 λ, 2x2 μ case. The lowest mode is somehow lost here and requires another case to be run in order to find it.

Tell me who you are → name _____

ME 235b
Finite Element Methods
Spring Quarter 1981

Stanford University

Instructor: T. J. R. Hughes

1 20
2 20
3 ~~40~~ 10
~~20~~
4
~~80~~ 70

Midterm Exam

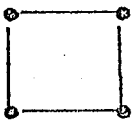
Time = 1 Hour
Total Points = 100

20

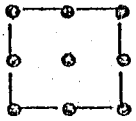
1. (20 points) A series of convergence rate studies on linear elastic boundary-value problems is performed. All results indicate that $\underset{\sim}{e} = \underset{\sim}{u}^h - \underset{\sim}{u}$ behaves as follows:

$$||e||_1 = O(h^2)$$

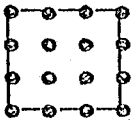
Which of the following elements was used in the studies?



4-node



9-node



16-node

Justify your answer.

✓ nine node element — since we had done problems in class with sobolev norms. If $\underset{\sim}{x}(\xi)$ is linear we showed that,

~~$\|e\|_{1,\Omega} \leq$~~ $\|\underset{\sim}{e}\|_1 \leq$ ~~const.~~ $c h^k \|u\|_{k+1}$ where k was the degree of completeness of polynomial

then for the nine pt node we can show that a quadratic polyn. is complete ∴ k = 2.

$$\text{approx.}_h - \|_m^2 \leq C_2^2 C_3^2 h^{2\alpha} \|u\|_s^2$$

$s \geq 1$

are connected

$k = 2 \qquad \|u\|_3$

the degree of complete polyn

S_{xy}

interp formula having 3 points

$$-1 \leq \frac{1}{1-\lambda\delta t}$$

$$-1 + \lambda\delta t \leq 1$$

$$\lambda\delta t \leq 2$$

$$\therefore \Delta t \geq \frac{2}{\lambda}$$

$\because \lambda < 0$

$$\lambda\delta t \leq 0 \leq 2$$

364

JOB 3525 HPFFVAR AWAITING
COMMAND? loc
JOB 3525 HPFFVAR AWAITING
COMMAND? fetch 3525 clr
COMMAND? list 1840/las
1840. *START

$u^h - u = e$.

$c_1 \|e\|_m^2 \leq a(e,e) \leq a(u^h - u, u^h - u)$. by be

we had shown that $\leq c_2 \|U$

$\therefore$

$\therefore \quad \|e\|_m \leq \sqrt{\frac{c_2 c_3}{c_1}}\, h^\alpha \|u\|_s$

$m = 1$

$\|e\|_1 \leq$ const $h^\alpha \|u\|_s$

if $\underset{\sim}{x}(\xi)$ is linear $s \leq \alpha$

$\leq$ const $h^\kappa \|u\|_{\kappa+1}$ if

is

ie $p(x,y) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 y + \alpha_4 y^2 + \alpha$

since highest degree is 2 $\Rightarrow$ lagrange

on a side $\therefore$ 9 node.

20

2. (20 points) To determine if a new finite element for 2-dimensional heat conduction is convergent, a patch test is performed. The results are as follows:

a	x_a	y_a	d_a Prob. 1	d_a Prob. 2	d_a Prob. 3
1	1.0	0.0	0.0	1.0	1
2	2.0	0.0	0.0	2.0	1
3	3.0	0.0	0.0	3.0	1
4	1.0	1.0	1.0	1.0	1
5 (interior node)	2.1	1.2	1.2	2.0	1
6	3.0	1.0	1.0	3.0	1
7	1.0	3.0	3.0	1.0	1
8	2.0	3.0	3.0	2.0	1
9	3.0	3.0	3.0	3.0	1

Ya

Nodes 1-4 and 6-9 are boundary nodes.

Does the element pass the patch test? Justify your answer.

no. since x_5 from prob 2 ≠ x coord of node 5

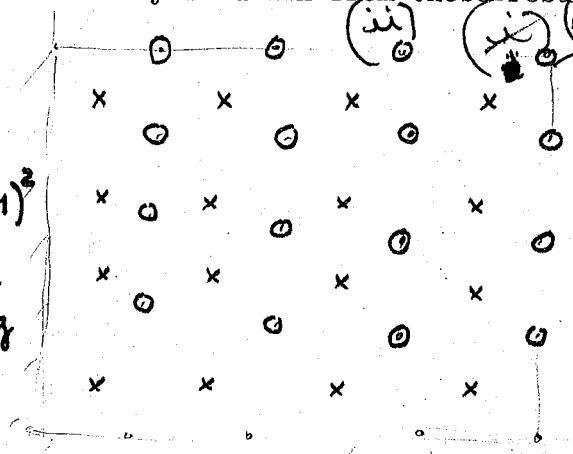
~~20~~ 10

3. (30 points) Consider the Lagrange quadrilaterals in two-dimensional elasticity. Take the case of the "general" element with n_{es} nodes per side. (Thus the element has n_{es}^2 nodes) Determine formulas for the constraint ratio, assuming

   i) selective reduced integration

   ii) exact integration

What conclusions may be drawn from these results?



for selective

0 - 2 eqns/pt

$\therefore n_{eq} = 2(n_{es}-1)^2$

× - constraints for selective integ

$n_c = (n_{es}-1)^2$

CR = 2

for n_{es} nodes there are $2(n_{es}-1)^2$ eqns ✓ 10

there are $n_{es}^2 - 1$ constraints ✗

$\therefore$ for regular the $CR = \dfrac{2(n_{es}-1)^2}{n_{es}^2-1}$ ✓

$CR = 2$ ✗ ?

~~for reduced integ~~

~~$CR = 2(n_{es}-1)^2$~~

for n_{es} nodes ⇒ poly of degree $2n_{es}-1$

$\therefore$ no. of points for integ is $\dfrac{n_{es}}{2}$

Thus we will have $\left(\dfrac{n_{es}}{2}\right)^2 - 1$ constraints

n

~~(i)~~ ~~(ii)~~

for selective integ

for n_{es} nodes there are $2(n_{es}-1)^2$ eqns

there are $(n_{es}-1)^2$ integ points / side

1 - less

$\therefore n_c = (n_{es}-1)^2$

$\therefore CR = \dfrac{2(n_{es}-1)^2}{2n_{es}-3} \cong 2(n_{es}-1)$

$\therefore 2(n_{es}-1) - 1 =$ degree of poly for selec integ ⇒ $\therefore$ it has $2(n_{es}-1)$ coeff has $2n_{es}-2-1$ constraints

selective integration has a better constraint ratio — 10

for exact integ 0 - 2 eq/pt

$\therefore n_{eq} = 2(n_{es}-1)^2$

$n_{es} = n_{es}^2 - 1$

$\therefore CR = \dfrac{2(n_{es}-1)^2}{n_{es}^2-1} = 2\left(\dfrac{n_{es}-1}{\quad}\right)$

for n_{es} points

lagr. poly is of degree $n_{es}-1$ & has n_{es} terms

$\therefore$ poly of $(n_{es}-1)^{deg}$ in x, y has n_{es}^2 terms

and $n_c = n_{es}^2 - 1 \Leftarrow \nabla \cdot u = 0$

20

4. (30 points) Consider the following "fractional-step" algorithm for the model heat equation:

$$\frac{d_{n+1/2} - d_n}{(\Delta t/2)} + \lambda d_n = 0$$

$$\frac{d_{n+1} - d_{n+1/2}}{(\Delta t/2)} + \lambda d_{n+1} = 0$$

i) Obtain an expresion for the amplification factor.

ii) Determine the conditions under which the algorithms is stable.

iii) What is the rate-of-convergence of this algorithm with respect to solutions of $\dot{d} + \lambda d = 0$.

$d_{n+1/2} - d_n + \lambda \Delta t/2\, d_n = 0$

$d_{n+1} - d_{n+1/2} + \lambda \Delta t/2\, d_{n+1} = d_{n+1/2}$

$d_{n+1} + \lambda \Delta t/2\, d_{n+1} - d_n + \lambda \Delta t/2\, d_n = 0$

$d_{n+1}(1 + \lambda \Delta t/2) = d_n(1 - \lambda \Delta t/2)$

$\therefore$ (i) 10 $A = \dfrac{1 - \lambda \Delta t/2}{1 + \lambda \Delta t/2}$ ✓ (ii) if $\left|\dfrac{1 - \lambda \Delta t/2}{1 + \lambda \Delta t/2}\right| < 1$ then this is unconditionally stable if $\lambda, \Delta t > 0$

(ii) ✓ 10 $-1 < \left|1 - \dfrac{2\lambda \Delta t/2}{1 + \lambda \Delta t/2}\right| < 1$

$\therefore -2\lambda \Delta t/2 < (1 + \lambda \Delta t/2) \cdot 0 = 0$ satisfied if $\Delta t, \lambda > 0$ or if $\Delta t, \lambda < 0$

$-1 < 1 - \dfrac{2\lambda \Delta t/2}{1 + \lambda \Delta t/2} \Rightarrow 1 > \dfrac{\lambda \Delta t/2}{1 + \lambda \Delta t/2}$ or $1 + \lambda \Delta t/2 > \lambda \Delta t/2$

$1 > 0$ ✓ $\forall\, \lambda, \Delta t$

$\therefore$ $|A| < 1$ then we get stability; for our normal system $\lambda, \Delta t > 0$ as "conditions"

say unconditionally stable

(iii) O this system is equivalent to the $\alpha = 1/2$ condition for $\dot{d} + \lambda d = 0$ (generalized trap. approx. of)

so what is one. ?

FOR $\alpha = 1/2$ $|\tau| < C\Delta t^{k+1}$ w/ $k = 2$

ME235B
Finite Element Methods
Spring Quarter 1981

Instructor: T. Hughes

Midterm Solutions

1. $$\|\underset{\sim}{e}\|_1 = \|\underset{\sim}{u}^h - \underset{\sim}{u}\|_1 \leq c\, h^k \|\underset{\sim}{u}\|_{k+1}$$

$k=2 \Rightarrow$ 9-node element

N.B. $\underset{\sim}{e} = \underset{\sim}{u}^h - \underset{\sim}{u}$ not $\underset{\sim}{U}^h - \underset{\sim}{u}$.

best approx
$\|\underset{\sim}{u}^h - \underset{\sim}{u}\|_1 \leq \text{const}\, \|\underset{\sim}{U}^h - \underset{\sim}{u}\|_1 \leq \text{const}\, h^k \|\underset{\sim}{u}\|_{k+1}$
$\|e\|_1$

2. Prob. 1 $d_a = y_a$ OK

Prob. 2 $d_a = x_a$ no good, i.e. $d_5 = 2.1 \neq 2.0$

Prob. 3 $d_a = 1$ OK

ans. NO

3. (i) $n_{eq} = 2(n_{es}-1)^2$

$n_c = (n_{es}-1)^2$ "standard" reduced quadrature of λ-term (see handout sheet)

$\therefore$ CR = 2 (selective reduced integration)

(ii) $n_c = n_{es}^2 - 1$ (exact int.)

$$CR = \frac{2(n_{es}-1)^2}{n_{es}^2-1} = \frac{2(n_{es}-1)}{n_{es}+1}$$

3. (cont'd)

For exact integration, CR improves the higher the order of the element. For example,

n_{es}	CR
2	2/3
3	1
4	6/5
⋮	⋮
∞	2

Thus the necessity of reduced integration decreases as the order of the element increases. (This is observed in practice.)

4. $$d_{n+1/2} - d_n + \lambda \frac{\Delta t}{2} d_n = 0 \qquad (1)$$

$$d_{n+1} - d_{n+1/2} + \lambda \frac{\Delta t}{2} d_{n+1} = 0 \qquad (2)$$

(i) solve (1) for $d_{n+1/2}$ and eliminate from (2) to arrive at:

$$d_{n+1} = A\, d_n \; ; \quad A = \frac{1 - \lambda \Delta t/2}{1 + \lambda \Delta t/2}$$

4. (cont'd) Observe that the foregoing A is identical to the one obtained for $\alpha = 1/2$ (i.e. trapezoidal rule) in the generalized trapezoidal family (see class notes)

(iii) unconditionally stable

(iiii) $k = 2$ (rate of conv.)

ME 235B
Finite Element Methods
Spring Quarter 1981

Stanford University
Instructor: T. Hughes

1 50
2 60
3 75
4 10
5 60
255

Course A+

Final Exam

Time = 3 Hours

Total Points = 300

Instructions: Open notes and homework are allowed, You may use all results previously obtained in the course lectures and homework, so be as brief as possible.

CESAR LEVY
Name

1. (60 points) The initial-boundary-value problem for an elastic membrane on a Winkler foundation is stated as follows:

Given $f:\Omega\times]0,\ T[\to\mathbb{R}$, $g:\Gamma_g\times]0,\ T[\to\mathbb{R}$, $h:\Gamma_h\times]0,\ T[\to\mathbb{R}$, and $w_0:\Omega\to\mathbb{R}$, and $\dot{w}_0:\Omega\to\mathbb{R}$; find $w:\Omega\times[0,\ T]\to\mathbb{R}$ such that

$$w_{,ii} - \alpha w + f = w_{,tt} \quad \text{on } \Omega\times]0,\ T[$$

$$w = g \qquad x\in\Gamma_g \quad , \quad t\in]0,\ T[$$

$$w_{,n} = h \qquad x\in\Gamma_h \quad , \quad t\in]0,\ T[$$

$$w = w_0 \qquad x\in\Omega \quad , \quad t = 0$$

$$w_{,t} = \dot{w}_0 \qquad x\in\Omega \quad , \quad t = 0$$

where $w_{,n} = \partial w/\partial n$ is the derivative in the direction of the outward unit normal vector and $\alpha>0$.

Set up the Following finite element paraphernalia:

(a) A weak form of the problem (i.e. "W") in which the h-boundary condition is "natural."

(b) The galerkin form of the problem (i.e. "G").

(c) The matrix initial-value ordinary differential equation problem (i.e. "M").

2. (70 points) The problem under consideration consists of

$$\underset{\sim}{M}\dot{\underset{\sim}{d}} + \underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}$$

$$\underset{\sim}{d}(0) = \underset{\sim}{d}_0$$

where $\underset{\sim}{M}$ and $\underset{\sim}{K}$ are symmetric and positive definite matrices $\underset{\sim}{F} = \underset{\sim}{F}(t)$ and $\underset{\sim}{d}_0$ are given. Consider the following one-parameter (α) family of predictor-corrector algorithms:

$$\underset{\sim}{M}\underset{\sim}{v}_{n+1} + \underset{\sim}{K}\tilde{\underset{\sim}{d}}_{n+1} = \underset{\sim}{F}_{n+1}$$

$$\tilde{\underset{\sim}{d}}_{n+1} = \underset{\sim}{d}_n + \Delta t(1 - \alpha)\underset{\sim}{v}_n$$

$$\underset{\sim}{d}_{n+1} = \tilde{\underset{\sim}{d}}_{n+1} + \Delta t\alpha \underset{\sim}{v}_{n+1}$$

Assume $\alpha \in]0, 1]$.

(a) Determine an expression for the amplification factor.

(b) Determine under what circumstances the algorithms are stable.

(c) Obtain an expression for the local truncation error.

(d) Determine the rate of convergence.

Hint The equations may be combined to form

$$(\underset{\sim}{M} - \alpha\Delta t\ \underset{\sim}{K})\underset{\sim}{v}_{n+1} + \underset{\sim}{K}\ \underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1}$$

which is useful in the analysis.

3. Consider the second-order ordinary differential equation

$$\ddot{d} + \omega^2 d = 0 \qquad (1)$$

(a) (25 points) The forward Euler method for (1) consists of the following equations:

$$a_n + \omega^2 d_n = 0$$

$$d_{n+1} = d_n + \Delta t\, v_n$$

$$v_{n+1} = v_n + \Delta t\, a_n$$

Set up the 2 x 2 amplification matrix for this method and determine the stability behavior.

(b) (50 points) The backward Euler method for (1) consists of:

$$a_{n+1} + \omega^2 d_{n+1} = 0$$

$$d_{n+1} = d_n + \Delta t\, v_{n+1}$$

$$v_{n+1} = v_n + \Delta t\, a_{n+1}$$

Determine the stability behavior and order-of-accuracy of this method.

4. (35 points) Consider a 4-node bilinear element which is used in an elastodynamics claculation. Assume that the Jacobian determinant is constant and equal to $A^e/4$, where A^e is the element area. The quadrature rule used to evaluate the element mass matrix is the "product trapezoidal rule," namely

ℓ	$\tilde{\xi}_\ell$	$\tilde{\eta}_\ell$	W_ℓ
1	-1	-1	1
2	1	-1	1
3	1	1	1
4	-1	1	1

Obtain exact expressions for all entries in the element mass matrix.

5. (60 points) Consider the 4-node, bilinear, isoparametric quadrilateral element. Show that $u^h_{i,i}$ at $\xi = \eta = 0$ is the mean value of $u^h_{i,i}$ over the element. That is, show

$$0 = \int_{-1}^{+1} \int_{-1}^{+1} \left(u^h_{i,i}(\xi, \eta) - u^h_{i,i}(0, 0) \right) j(\xi, \eta) d\xi \, d\eta$$

1. (a) 20 Given f, g, h, w_0 and $\dot{w}_0$ as in (S) Find: w s.t. $\forall u \in V$ (ie $w = v + g$)

where $w: [0,T] \times \bar{\Omega} \to S_{g(t)}$ and $u = 0$ on Γ_{g_i} $\forall t \in [0,t]$

$$\int_\Omega (u \nabla^2 w - \alpha w u + f u - w_{,tt} u)\, d\Omega = 0$$

$$\int_\Omega [(u w_{,i})_{,i} - u_{,i} w_{,i}]\, d\Omega = \int_\Omega \alpha w u\, d\Omega - \int_\Omega f u\, d\Omega + \int_\Omega u\, w_{,tt}\, d\Omega.$$

$$\int_\Gamma (n \cdot \nabla w) u\, d\Gamma - \int u_{,i} w_{,i}\, d\Omega = \dots$$

$$\int_\Gamma w_{,n} u\, d\Gamma - \int w_{,i} u_{,i}\, d\Omega = \dots$$

now $w_{,n} = h$ on Γ_h

$$\therefore \int_\Omega w_{,i} u_{,i}\, d\Omega = \int_{\Gamma_h} h u\, d\Gamma - \int_\Omega \alpha w u\, d\Omega + \int_\Omega f u\, d\Omega - \int u\, w_{,tt}\, d\Omega$$

$$\rightarrow \boxed{(u, \ddot{w}) + a(w,u) + (\alpha w, u) = (u,h)_{\Gamma_h} + (u,f)} \checkmark$$

where $a(w,u) = \int_\Omega w_{,i} u_{,i}\, d\Omega$ $\quad (u,h)_{\Gamma_h} = \int_{\Gamma_h} h u\, d\Gamma$ $\quad (A,B) = \int_\Omega AB\, d\Omega$

$$\boxed{\begin{array}{ll} \text{also} & (u, w(0) - w_0) = 0 \\ \text{also} & (u, \dot{w}(0) - \dot{w}_0) = 0 \end{array}} \checkmark$$

1. (b) 20 Given $f, g, h, w_0, \dot{w}_0$ as in (S)

Find a $v^h: [0,T] \to V^h \subset V$ where $w^h = v^h + g^h$, $g^h: [0,T] \to S^h$

s.t. $\forall u^h \in V^h$ then

$$(\alpha v, u) + (u^h, \ddot{v}^h) + a(v^h, u^h) = (u^h, h)_{\Gamma_h} + (u^h, f) - (\alpha g^h, u^h) - (u^h, \ddot{g}^h) - a(g^h, u^h).$$

and $\quad (u^h, v(0)^h) = (u^h, w_0) - (u^h, g^h(0))$ ✓

$$(u^h, \dot{v}(0)^h) = (u^h, \dot{w}_0) - (u^h, \dot{g}^h(0)) \checkmark$$

1. (c) 10 let $g^h(x,t) = \sum_{\eta_g} N_A(x)\, g_A(t) \in S^h$; $v^h(x,t) = \sum_{\eta - \eta_g} N_A(x)\, d_A(t) \in V^h$ find $d: [0,T] \to \mathbb{R}$

Then letting $M = \mathop{A}_{e=1}^{nel} (m^e_{pq})$ $\quad m^e = [m^e_{pq}]$ $\quad m^e_{pq} = \int_{\Omega^e} N_A N_B\, d\Omega$ $\quad p = \text{~~scrambled~~}\ a$

$K = \mathop{A}_{e=1}^{nel} (k^e)$ $\quad k^e = [k^e_{pq}]$ $\quad k^e_{pq} = \int_{\Omega^e} \text{~~B}^T D B\ d\Omega$ $\quad q = \text{~~scrambled~~}\ b$

$\leftarrow N_{a,i} N_{b,i}$

$$\underset{\sim}{F}(t) = \underset{e=1}{\overset{Nel}{A}} \left(\underset{\sim}{f}^e(t) \right) \qquad \underset{\sim}{f}^e = [f_p^e] \qquad f_p^e = \int_{\Omega^e} N_a f \, d\Omega + \int_{\Gamma^e} N_a h \, d\Gamma - \sum_{q=1}^{n_{en}} \left(k_{pq}^e g_q^e + c_{pq}^e \dot{g}_q^e + m_{pq}^e \ddot{g}_q^e \right).$$

where $\underset{\sim}{c}^e = [c_{pq}^e] \qquad \underset{\sim}{C} = \underset{e=1}{\overset{nel}{A}} (\underset{\sim}{c}^e) \qquad c_{pq}^e = \int_{\Omega^e} \alpha N_a N_\beta \, d\Omega$

if $\alpha = const$ then $c_{pq}^e = \alpha m_{pq}^e$ ✓

∴ our eqns become

$$\underset{\sim}{M}\ddot{\underset{\sim}{d}} + (\underset{\sim}{C} + \underset{\sim}{K})\underset{\sim}{d} = \underset{\sim}{F} \quad ✓$$

---

and $\underset{\sim}{d}_0 = \underset{\sim}{\bar{M}}^{-1} \left\{ \underset{e=1}{\overset{nel}{A}} (\bar{\underset{\sim}{d}}^e) \right\} \qquad \underset{\sim}{M} = A(\underset{\sim}{\bar{m}}^e) \qquad m_{pq}^e = \int_{\Omega^e} N_a N_\beta \delta_{ij} \, d\Omega$

$\bar{\underset{\sim}{d}}^e = \{ d_p^e \} \qquad d_p^e = \int_{\Omega^e} N_a \ddot{w}_{0,i} \, d\Omega - \sum_q \bar{m}_{pq}^e g_q^e(0)$

$\dot{\underset{\sim}{d}}_0 = \underset{\sim}{\bar{M}}^{-1} \left\{ \underset{e=1}{\overset{nel}{A}} (\bar{\dot{\underset{\sim}{d}}}^e) \right\}$ where $\dot{d}_p^e = \int_{\Omega^e} N_a \dot{w}_{0,i} \, d\Omega - \sum \bar{m}_{pq}^e \dot{g}_q^e(0)$

---

2. $\left(\underset{\sim}{M} - \alpha\Delta t\, \underset{\sim}{K}\right) \underset{\sim}{v}_{n+1} + \underset{\sim}{K}\, \underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1}$

now $\underset{\sim}{M}\,\underset{\sim}{v}_n + \underset{\sim}{K}\,\underset{\sim}{d}_n = \underset{\sim}{F}_n$

$$\underset{\sim}{M}\,\underset{\sim}{v}_n + \underset{\sim}{K}\left[\underset{\sim}{d}_n + \Delta t\,(1-\alpha)\,\underset{\sim}{v}_n\right] = \underset{\sim}{F}_n$$

$$\left[\underset{\sim}{M} + K\,\Delta t\,(1-\alpha)\right]\underset{\sim}{v}_n + \underset{\sim}{K}\,\underset{\sim}{d}_n = \underset{\sim}{F}_n \qquad (2)$$

and $\underset{\sim}{d}_{n+1} - \Delta t\,\alpha\,\underset{\sim}{v}_{n+1} = \underset{\sim}{d}_n + \Delta t\,(1-\alpha)\,\underset{\sim}{v}_n$

$$\Delta t\,\alpha\,\underset{\sim}{v}_{n+1} = \underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n - \Delta t\,(1-\alpha)\,\underset{\sim}{v}_n$$

$$\left(\underset{\sim}{M} - \alpha\Delta t\underset{\sim}{K}\right)\left[\underset{\sim}{d}_{n+1} - \underset{\sim}{d}_n - \Delta t\,(1-\alpha)\,\underset{\sim}{v}_n\right] + \Delta t\,\underset{\sim}{K}\,\underset{\sim}{d}_{n+1} = \underset{\sim}{F}_{n+1}\,\Delta t\,\alpha \qquad (1)$$

applying the fact that $\underset{\sim}{\psi}_\ell^T\,\underset{\sim}{K}\,\underset{\sim}{\psi}_m = \lambda_\ell\,\delta_{\ell m}$ & $\underset{\sim}{\psi}_\ell^T\,\underset{\sim}{M}\,\underset{\sim}{\psi}_m = \delta_{\ell m}$

& $\underset{\sim}{d}_{n+1} = \sum_m d_{m,n+1}(t)\,\underset{\sim}{\psi}_m \qquad \underset{\sim}{d}_n = \sum d_m(t)\,\underset{\sim}{\psi}_m \qquad \dot{\underset{\sim}{d}}_n = \sum \dot{d}_m(t)\,\underset{\sim}{\psi}_m$

then we get $\left[d_{\ell,n+1}\,(1-\alpha\Delta t\lambda_\ell) - d_{\ell n}\,(1-\alpha\Delta t\lambda_\ell) - \Delta t\,(1-\alpha)\,\dot{d}_{\ell n}\,(1-\alpha\Delta t\lambda_\ell)\right]$

$$+\Delta t\,\alpha\,\lambda_\ell\, d_{n+1} = F_{\ell,n+1}\,\Delta t\,\alpha \qquad \text{from (1)}$$

and $\left[1 + \lambda_\ell\,\Delta t\,(1-\alpha)\right]\dot{d}_{\ell n} + \lambda_\ell\, d_{\ell n} = F_{\ell n} \qquad \text{from (2)}$

then $\dot{d}_{\ell n} = \dfrac{F_{\ell n} - \lambda_\ell d_{\ell n}}{1 + \lambda\,\Delta t\,(1-\alpha)}$ solving for $\dot{d}_{\ell n}$

Put into (1)

$$\therefore \quad d_{\ell,n+1}\,(1-\alpha\Delta t\lambda_\ell + \Delta t\,\alpha\,\lambda_\ell) - d_\ell\,(1-\alpha\Delta t\,\lambda_\ell) - \Delta t\,(1-\alpha)\,(1-\alpha\Delta t\lambda)\left(\frac{F_{\ell n} - \lambda d_\ell}{1+\lambda\Delta t(1-\alpha)}\right)$$

$$= F_{\ell,n+1}\,\Delta t\,\alpha$$

$$d_{\ell,n+1} = d_\ell\left\{(1-\alpha\Delta t\lambda_\ell) - \frac{\lambda\Delta t(1-\alpha)(1-\alpha\Delta t\lambda)}{1+\lambda\Delta t(1-\alpha)}\right\} + F_{\ell,n+1}\,\Delta t\,\alpha + \frac{F_\ell\,(\Delta t)(1-\alpha)(1-\alpha\Delta t\lambda)}{1+\lambda\Delta t(1-\alpha)}$$

$$d_\ell\left\{\frac{(1-\alpha\Delta t\lambda)\left[1+\lambda\Delta t(1-\alpha) - \lambda\Delta t(1-\alpha)\right]}{1+\lambda\Delta t(1-\alpha)}\right\} + \text{Lin}$$

$$d_{\ell,n+1} = d_\ell\left[\frac{1-\alpha\Delta t\lambda}{1-\alpha\lambda\Delta t}\cdot(1+\lambda\Delta t)\right]$$

(a) 30 $A = 1 - \lambda\Delta t$ ✓

(b) 15 if $|A|<1$ i.e. $-1 < 1-\lambda\Delta t < 1$ $\Rightarrow$ if $\Delta t < \frac{2}{\lambda}$ ✓

for $\lambda, \Delta t > 0$

(c) if $\lambda \Delta t > 0$ & $\lambda \Delta t < 2$ or $\boxed{\Delta t < 2/\lambda}$ ie $\alpha = 0$

~~(d)~~ (c) $\boxed{|\tau| \leq C \Delta t^2}$ (d) $k = 1$ 10

5 ✓ ✓

wanted an expression -10

75

3. $a_n + \omega^2 d_n = 0 \quad \Rightarrow a_n = -\omega^2 d_n$

(a) 25

$$\begin{pmatrix} d_{n+1} \\ v_{n+1} \end{pmatrix} = \begin{matrix} d_n + \Delta t v_n \\ v_n - \Delta t \omega^2 d_n \end{matrix} = \begin{pmatrix} 1 & \Delta t \\ -\Delta t \omega^2 & 1 \end{pmatrix} \begin{pmatrix} d_n \\ v_n \end{pmatrix}$$

$$\underset{\sim}{A} = \begin{pmatrix} 1 & \Delta t \\ -\Delta t \omega^2 & 1 \end{pmatrix} \qquad A_1 = \tfrac{1}{2} \operatorname{tr} \underset{\sim}{A} = \tfrac{1}{2}(2) = 1$$

$$A_2 = \det \underset{\sim}{A} = 1 + \Delta t^2 \omega^2$$

$$\lambda_\pm = A_1 \pm \sqrt{A_1^2 - A_2} = 1 \pm \sqrt{1 - 1 - \Delta t^2 \omega^2} = 1 \pm i\omega\Delta t \quad \checkmark$$

$$|\lambda|^{1/2} = (1 + \omega^2 \Delta t^2)^{1/2} > 1 \quad \checkmark \quad \forall\ \Delta t \neq 0 \quad \omega \neq 0 \quad \therefore \text{ system blows up}$$

✓ not stable. Even though ✓ $A_1 \leq (A_2+1)/2 \qquad A_2 \not\leq 1.$

(b) 50 $\quad a_{n+1} + \omega^2 d_{n+1} = 0 \qquad a_{n+1} = -\omega^2 d_{n+1}$

$$d_{n+1} = d_n + \Delta t\, v_{n+1}$$

$$v_{n+1} = v_n + (\Delta t\, \omega^2\, d_{n+1}) \quad \Rightarrow \begin{pmatrix} 1 & -\Delta t \\ \Delta t \omega^2 & 1 \end{pmatrix} \begin{matrix} d_{n+1} \\ v_{n+1} \end{matrix} = \begin{pmatrix} d_n \\ v_n \end{pmatrix}$$

$$A^{-1} = \begin{pmatrix} 1 & -\Delta t \\ \Delta t \omega^2 & 1 \end{pmatrix} \qquad A = \frac{\begin{pmatrix} 1 & \Delta t \\ -\omega^2 \Delta t & 1 \end{pmatrix}}{1 + \Delta t^2 \omega^2}$$

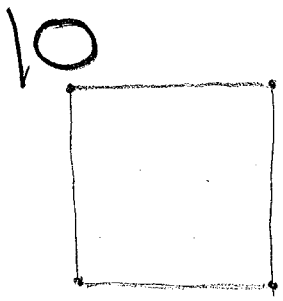
$$A_1 = \tfrac{1}{2} \operatorname{tr} A = \frac{1}{1 + \Delta t^2 \omega^2} \qquad A_2 = \det \underset{\sim}{A} = 1$$

$$\therefore \quad \lambda_\pm = \frac{1}{1 + \Delta t^2 \omega^2} \pm \sqrt{\frac{1 - (1 + \Delta t^2 \omega^2)^2}{(1 + \Delta t^2 \omega^2)^2}}$$

$$\frac{1}{1 + \Delta t^2 \omega^2} \pm i \sqrt{\frac{(\Delta t^2 \omega^2)(\omega^2 \Delta t^2 + 2)}{(1 + \Delta t^2 \omega^2)^2}} \quad \text{Complex}$$

✓ now since $-1 < A_1 < 1$ and $A_2 = 1$ we have spectral stability

since backward Euler $\alpha = 1 \quad \therefore \quad |\tau(t)| \leq c\, \Delta t^{k+1}$ where $k = 1$



$$\underset{\sim}{m}^e = \rho \int_{\Omega^e} N_A N_B \, d\Omega = \frac{A}{4} \int_{-1}^{1}\int_{-1}^{1} N_A N_B \, d\xi \, d\eta = \frac{A}{4} \left\{ N_A N_B \text{ evaluated at the 4 pts} \times \text{wts} \right\}$$

$$= \frac{A}{4} \left\{ N_A(-a,-a) N_B(-a,-a) + \dots \right\}$$

-15 δ_{ij}

where

	$\frac{(-1,-1)}{1}$	$\frac{(+1,-1)}{0}$	$\frac{(1,1)}{0}$	$\frac{(-1,1)}{0}$
$N_1 = \frac{(1-\xi)(1-\eta)}{4}$				
$N_2 = \frac{(1+\xi)(1-\eta)}{4}$	0	1	0	0
$N_3 = \frac{(1+\xi)(1+\eta)}{4}$	0	0	1	0
$N_4 = \frac{(1-\xi)(1+\eta)}{4}$	0	0	0	1

$$\underset{\sim}{m}^e = \frac{A}{4} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

didn't understand problem well enough to ask questions on it

5. 60

$$u^h_{1,1} = \sum_{a=1}^{4} N_{a,x}\, d_{xa} \qquad u^h_{2,2} = \sum_{a=1}^{4} N_{a,y}\, d_{ya}$$

$$= \sum_{a=1}^{4} \left[N_{a,\xi}\, y_{,\eta} - N_{a,\eta}\, y_{,\xi} \right] / j \; d_{xa} \qquad \sum_{a=1}^{4} -\left(N_{a,\xi}\, x_{,\eta} - N_{a,\eta}\, x_{,\xi} \right) / j \; d_{ya}$$

$$\therefore \iint (u^h_{1,1} + u^h_{2,2})\, j \, d\xi d\eta = \iint \sum_{a=1}^{4} N_{a,\xi} \left(d_{xa} y_{,\eta} - x_{,\xi} d_{ya} \right) d\xi d\eta + \sum N_{a,\eta} \left(d_{ya} x_{,\xi} - y_{,\eta} d_{xa} \right) d\xi d\eta$$

but $\quad y_{,\eta} = \sum_{a=1}^{4} N_a(\xi)_{,\eta}\, y_a \qquad x_{,\eta} = \sum N_a(\xi)_{,\eta}\, x_a \quad$ etc.

$$\iint \nabla \cdot \underset{\sim}{u}\, j \, d\xi d\eta = \iint d\xi d\eta \sum_{a=1}^{4} N_{a,\xi} \sum_{b=1}^{4} \left(d_{xa} N_{b,\eta}\, y_b - d_{ya} N_{b,\xi}\, x_b \right) + \sum N_{a,\eta} \sum \left(d_{ya} N_{b,\xi}\, x_b - d_{xa} N_{b,\eta}\, y_b \right)$$

Now since $N_a(\xi)$ are linear in $\underset{\sim}{\xi}$ then $N_{a,\xi}$ will be linear in η and vice versa $N_{a,\eta}$ will be linear in ξ. When integrated from $(-1, 1)$ the linear term (constant in η) is quadratic and gives no contrib $\therefore$ this is same as setting $\xi, \eta = 0$ ✓

$\therefore \int_{-1}^{1} \int_{-1}^{1} \left[u^h_{i,i}(\xi, \eta) - u^h_{i,i}(0,0) \right] j(\xi, \eta)\, d\xi d\eta = 0$. All other terms ie

$d_{ya}, y_a, x_b, x_a, y_b, d_{xa}$ are constants and do not change overall result

$1 \pm \eta \qquad \eta \pm \eta^2 \Big|_{-1}^{1}$

MF 235 B, Finite Element Methods. Stanford U.
Spring Quarter 1981 Instructor: T. Hughes

Final Exam Solutions

1 a. Given $f: \Omega \times]0,T[\to \mathbb{R}$, $g: \Gamma_g \times]0,T[\to \mathbb{R}$, $h: \Gamma_h \times]0,T[\to \mathbb{R}$, $w_0: \Omega \to \mathbb{R}$, and $\dot{w}_0: \Omega \to \mathbb{R}$

Find $w: \overline{\Omega} \times [0,T] \to \mathbb{R}$ $(w \in \mathcal{S})$ (assume)
$\ni \forall \bar{w}: \overline{\Omega} \times [0,T] \to \mathbb{R}$ $(\bar{w} \in \mathcal{V})$ $\ni$

$$(\bar{w}, \ddot{w}) + a(\bar{w}, w) = (\bar{w}, f) + (\bar{w}, h)_\Gamma$$

$$(\bar{w}, w(0) - w_0) = 0$$
$$(\bar{w}, \dot{w}(0) - \dot{w}_0) = 0$$

where: $(\bar{w}, \ddot{w}) = \int_\Omega \bar{w} \ddot{w} \, d\Omega$

$$a(\bar{w}, w) = \int_\Omega \{ \bar{w}_{,i} w_{,i} + \alpha \bar{w} w \} d\Omega$$

$$(\bar{w}, f) = \int_\Omega \bar{w} f \, d\Omega$$

$$(\bar{w}, h)_\Gamma = \int_{\Gamma_h} \bar{w} h \, d\Gamma, \text{ etc.}$$

b. Given ... ; Find $w^h = \overset{\mathcal{U}^h}{w^h} + g^h \in \mathcal{S}^h$
$\ni \forall \bar{w}^h \in \mathcal{V}^h$ (as usual)

$$(\bar{w}^h, \ddot{w}^h) + a(\bar{w}^h, w^h) = (\bar{w}^h, f) + (\bar{w}, h)_\Gamma$$

1b (cont'd)

$$(\tilde{w}^h, \ddot{u}^h) + a(\tilde{w}^h, w^h) = (\tilde{w}^h, f) + (\tilde{w}^h, h)_\Gamma - (\tilde{w}^h, \ddot{g}^h) - a(\tilde{w}^h, g^h)$$

(discussion for initial conditions)

1c $$\underset{\sim}{M}\,\underset{\sim}{\ddot{d}} + \underset{\sim}{K}\,\underset{\sim}{d} = \underset{\sim}{F} \; ; \; \underset{\sim}{d}(0) = \underset{\sim}{d}_0 \; ; \; \dot{\underset{\sim}{d}}(0) = \underset{\sim}{\dot{d}}_0$$

where

$$m^e_{ab} = \int_{\Omega^e} N_a N_b \, d\Omega$$

$$k^e_{ab} = \int_{\Omega^e} N_{a,i} N_{b,i} + \alpha N_a N_b \, d\Omega$$

$$f^e_a = \int_{\Omega^e} N_a f \, d\Omega + \int_{\Gamma^e_h} N_a h \, d\Gamma - \sum_{b=1}^{n_{en}} \left(m^e_{ab} \ddot{g}_b + k^e_{ab} g_b \right)$$

(usual assembly algo. ; + initial condition)

2. The modal eqs:

$$\dot{v}_{n+1} + \lambda \tilde{d}_{n+1} = F_{n+1} \quad (1)$$

$$\tilde{d}_{n+1} = d_n + \Delta t (1-\alpha) v_n \quad (2)$$

$$d_{n+1} = \tilde{d}_{n+1} + \Delta t \alpha v_{n+1} \quad (3)$$

(1) and (3) $\Rightarrow$ $(1-\alpha\Delta t\lambda)v_{n+1} + \lambda d_{n+1} = F_{n+1}$ (4)

eliminate at n: $(1-\alpha\Delta t\lambda)v_n + \lambda d_n = F_n$ (5)

(2) and (3) $\Rightarrow$ $d_{n+1} = d_n + \Delta t(1-\alpha)v_n + \Delta t \alpha v_{n+1}$ (6)

mult. (6) by $(1-\alpha\Delta t\lambda)$:

$$(1-\alpha\Delta t\lambda)d_{n+1} = (1-\alpha\Delta t\lambda)d_n + \Delta t(1-\alpha)\underbrace{(1-\alpha\Delta t\lambda)v_n}_{(5)} + \Delta t\alpha \underbrace{(1-\alpha\Delta t\lambda)v_{n+1}}_{(4)}$$

$$(1-\alpha\Delta t\lambda)d_{n+1} = (1-\alpha\Delta t\lambda)d_n + \Delta t(1-\alpha)[F_n - \lambda d_n] + \Delta t\alpha[F_{n+1} - \lambda d_{n+1}]$$

$$d_{n+1} = (1-\lambda\Delta t)d_n + \Delta t \underbrace{[(1-\alpha)F_n + \alpha F_{n+1}]}_{F_{n+\alpha}}$$

(30) a. $A = 1-\lambda\Delta t$

(15) b. $|A| < 1 \Rightarrow \lambda\Delta t < 2$.

(15) c. $T(t_n) = d(t_{n+1}) - (1-\lambda\Delta t)\, d(t_n) - \Delta t F_{n+\alpha}$

$$= \Delta t \Big(\underbrace{\dot{d}(t_n) + \lambda d(t_n) - F(t_n)}_{0} \Big) + \Delta t^2 \Big(\frac{\ddot{d}(t_n)}{2} - \alpha \dot{F}(t_n) \Big) + \dots$$

$\therefore |T(t)| \le c\,\Delta t^2$

(10) d. $k = 1$ (first-order)

$$\text{3. e. } \begin{Bmatrix} d_{n+1} \\ v_{n+1} \end{Bmatrix} = \underset{\sim}{A} \begin{Bmatrix} d_n \\ v_n \end{Bmatrix} \quad (25)$$

$$\underset{\sim}{A} = \begin{bmatrix} 1 & \Delta t \\ -\Delta t \omega^2 & 1 \end{bmatrix}$$

$$g(\underset{\sim}{A}) = (1+\Omega^2)^{1/2} \quad ; \quad \Omega = \omega \Delta t$$

$$> 1 \quad \forall \Omega > 0 \quad \therefore \text{ unconditionally unstable}$$

$$\text{5. } g(\underset{\sim}{A}) = 1/(1+\Omega^2)^{1/2} < 1, \quad \forall \Omega > 0 \quad (50)$$

$\therefore$ unconditionally stable

first-order accurate $\left[\underset{\sim}{T} = O(\Delta t^2) \right]$

$$\text{4. } m_{pq} = \delta_{ij} \int_{\Omega^e} \rho N_a N_b \, d\Omega_\xi \quad (35)$$

$$p = (a-1)n_{sd} + i \qquad q = (b-1)n_{sd} + j$$

$$= \delta_{ij} \int_{-1}^{+1}\int_{-1}^{+1} \rho N_a N_b |J| \, d\xi_1 d\xi_2 = \delta_{ij} \frac{A^e}{4} \int_{-1}^{+1}\int_{-1}^{+1} \rho N_a N_b \, d\xi_1 d\xi_2$$

two-pt rule

$$= \delta_{ij} \frac{A^e}{4} \left(\delta_{a1}\delta_{b1}\rho(\xi_{1,1},\xi_{2,1}) + \cdots + \delta_{a4}\delta_{b4}\rho(\xi_{1,4},\xi_{2,4}) \right)$$

$$= \delta_{ij} \frac{A^e}{4} \left(\delta_{a1}\delta_{b1}\rho(-1,-1) + \cdots + \delta_{a4}\delta_{b4}\rho(-1,1) \right)$$

$$[m_{pq}] = \frac{A^e}{4} \begin{bmatrix} \rho(-1,-1) & & & & & & & \\ & \rho(-1,-1) & & & & & & \\ & & \rho(1,-1) & & & & & \\ & & & \rho(1,-1) & & & & \\ & & & & \rho(1,1) & & & \\ & & & & & \rho(1,1) & & \\ & & & & & & \rho(-1,1) & \\ & & & & & & & \rho(-1,1) \end{bmatrix}$$

diagonal $\Rightarrow$ zero shear.

oo)

5. $u_i^h = \sum_{a=1}^{4} N_a \, d_{ia}$

$$\int_{-1}^{+1}\int_{-1}^{+1} \left(u^h_{i,i}(\xi,\zeta) - u^h_{i,i}(0,0)\right) j(\xi,\zeta)\, d\xi\, d\zeta$$

$$= \sum_{a=1}^{4} d_{ia} \left\{ \int_{-1}^{+1}\int_{-1}^{+1} \left(N_{a,i} - N_{a,i}\big|_{(0,0)}\right) j \, d\xi\, d\zeta \right\}$$

recall:

$$\begin{bmatrix} \xi_{,x} & \xi_{,y} \\ \zeta_{,x} & \zeta_{,y} \end{bmatrix} = \begin{bmatrix} x_{,\xi} & x_{,\zeta} \\ y_{,\xi} & y_{,\zeta} \end{bmatrix}^{-1} = j^{-1} \begin{bmatrix} y_{,\zeta} & -x_{,\zeta} \\ -y_{,\xi} & x_{,\xi} \end{bmatrix}$$

consider:

$$N_{a,x} = (N_{a,\xi} N_{b,\zeta} - N_{a,\zeta} N_{b,\xi})\, y_b / j$$

(sum b)

$$= \left[\xi_a (1+\zeta_a\zeta)\zeta_b(1+\xi_b\xi) - \zeta_a(1+\xi_a\xi)\xi_b(1+\zeta_b\zeta)\right] y_b/(16j)$$

(sum b).

$$= (\xi_a\zeta_b - \zeta_a\xi_b + \text{sum of bilin* terms in } \xi,\zeta)\, y_b/(16j)$$

$$\therefore \int_{-1}^{+1}\int_{-1}^{+1} N_{a,x}\, j \, d\xi\, d\zeta = (\xi_a\zeta_b - \zeta_a\xi_b) y_b/4$$

(sum b)

---

* The bilin. terms actually cancel, but this is not needed in the proof.

5. (cont'd) The preceding result derives from the fact that the j's cancel and the η bilinear terms integrate to zero.

$$N_{a,x}\Big|_{(0,0)} = (\xi_a z_b - z_a \xi_b) y_b \Big/ (16 j(0,0)) \qquad \text{sum } b$$

$$\int_{-1}^{+1}\int_{-1}^{+1} N_{a,x}\Big|_{(0,0)} j\, d\xi\, d\eta = N_{a,x}\Big|_{(0,0)} \int_{-1}^{+1}\int_{-1}^{+1} j\, d\xi\, d\eta$$

Thus to prove $\quad 0 = \int_{-1}^{+1}\int_{-1}^{+1} (N_{a,x} - N_{a,x}\Big|_{(0,0)}) j\, d\xi\, d\eta$

it suffices to show that

$$\int_{-1}^{+1}\int_{-1}^{+1} j\, d\xi\, d\eta = 4 j(0,0)$$

$$j = x_{,\xi} y_{,\eta} - x_{,\eta} y_{,\xi}$$
$$= (N_{a,\xi} N_{b,\eta} - N_{a,\eta} N_{b,\xi}) x_a y_b \qquad (\text{sum } a,b)$$
$$= (\xi_a z_b - z_a \xi_b + \text{bilinear terms in } \xi, \eta) x_a y_b \qquad (\text{sum } a,b)$$

$$\therefore \int_{-1}^{+1}\int_{-1}^{+1} j\, d\xi\, d\eta = 4 (\xi_a z_b - z_a \xi_b) x_a y_b$$

and $j(0,0) = (\xi_a z_b - z_a \xi_b) x_a y_b$. ∎

Is there a simpler way to see this?

---

* See footnote on p. 5.

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 15, 733–751 (1980)

TECHNIQUES FOR DEVELOPING 'SPECIAL' FINITE ELEMENT SHAPE FUNCTIONS WITH PARTICULAR REFERENCE TO SINGULARITIES

THOMAS J. R. HUGHES†

Division of Engineering and Applied Science, California Institute of Technology, Pasadena, California 91125, U.S.A.

AND

J. E. AKIN‡

Department of Engineering Science and Mechanics, University of Tennessee, Knoxville, Tennessee 37916, U.S.A.

SUMMARY

A concise and efficient algorithm is presented for deriving finite element shape functions from an arbitrary set of independent functions. Based on special one-dimensional interpolatory schemes derived via the algorithm, a variety of two- and three-dimensional interpolatory schemes are developed which are useful for modelling singular behaviour. The methodology presented is general and may be fruitfully applied to the development of 'special' finite element shape functions for a variety of other situations.

INTRODUCTION

In many areas of finite element analysis, elements with special properties are required to achieve maximal accuracy. As examples, we may mention 'upwind' finite elements for treating convection operators;[1–3] infinite elements for the representation of spatial domains which extend to infinity;[4,5] boundary-layer elements;[6] and singular elements for modelling point and line singularities engendered by geometric features such as re-entrant corners and cracks.[7–9] In this paper we are concerned with techniques for developing shape (i.e. interpolation) functions for special elements in general, but with particular emphasis on singularities.

Geometric features of the spatial domain may give rise to highly singular response in the solution of a field problem.[10] Eigenfunction analyses are frequently available for purposes of characterizing the asymptotic strength of the singularity.[11] However, the actual solution depends upon the nature of the loading and there are situations when the singularity does not contribute significantly.[12] For this reason it has been suggested that, ideally, singular finite elements should strike some balance between being able to represent smooth behaviour and potential singular behaviour.[12] In this way the element will remain effective under all loading conditions. Translated into analytical terms, the finite element shape functions should be designed to represent the standard low-order polynomials as well as the singular functions emanating from asymptotic analysis. To construct shape functions of this type gives rise to an interpolation problem which involves a system of linear simultaneous equations. In special circumstances explicit representations are available for the solution of the interpolation problem. A most noteworthy example is the Lagrange interpolation formula (see, e.g., Davis[13])

† Associate Professor of Structural Mechanics.
‡ Professor of Engineering Science and Mechanics.

0029–5981/80/0515–0733$01.00

Received 14 May 1979
Revised 24 July 1979

which is the solution of the interpolation problem for polynomials passing through distinct points. It appears, unfortunately, that no such representation exists when different classes of functions are merged, as seems appropriate for singular elements. Although, in principle, the interpolation problem could be solved numerically for each singular element during formation, this would be highly inefficient due to the increased number of computations required. Furthermore, numerical sensitivity may manifest itself in certain configurations. Consequently, it appears desirable to develop techniques for systematically defining shape functions for singularity modelling (and for developing special elements in general) which circumvent the interpolation problem. This is further attested to by the fact that very few elements developed so far achieve the desired properties delineated above (examples of ones that do may be found in References 7 and 14), and those that do have been developed by *ad hoc* means; thus generalization to higher order elements, or more complex situations, is by no means obvious.

In the next section we present a highly concise algorithm for generating shape functions from an arbitrary starting set of independent functions. A novel feature of the procedure is that *no* coefficient matrix is involved (i.e. no system of simultaneous equations need be solved).

Then we develop some useful one-dimensional interpolatory schemes for modelling singularities by employing the interpolation algorithm. These schemes are the basis for constructing two- and three-dimensional elements for modelling point and line singularities. A number of examples are presented which illustrate the methodology of constructing multi-dimensional elements. Several of these examples appear to be new and are of interest in their own right.

Conclusions are presented in the final section.

PRELIMINARY RESULTS

Let $\{\xi_a | a = 1, 2, \ldots, n\}$ denote a set of distinct points (i.e. 'nodes') in the interval $[-1, 1]$. A generic point in $[-1, 1]$ is denoted ξ. Let $r = r(\xi) = (1+\xi)/2$ and $r_a = r(\xi_a)$. Note that $r(\xi) \in [0, 1]$ for all $\xi \in [-1, 1]$.† The physical co-ordinate is denoted x.

In multi-dimensional applications, we use bold-faced notations analogous to the preceding ones. For example, in three dimensions $\boldsymbol{\xi} = (\xi, \eta, \zeta)$; $\mathbf{r} = (r, s, t)$; where $s = (1+\eta)/2$; $t = (1+\zeta)/2$; $\eta, \zeta \in [-1, 1]$, and $\mathbf{x} = (x, y, z)$.

Lagrange interpolation formula

The *Lagrange polynomials* are defined by

$$l_a(f) = \frac{\prod_{\substack{b=1 \\ b \neq a}}^{m} (f - f_b)}{\prod_{\substack{b=1 \\ b \neq a}}^{m} (f_a - f_b)}, \qquad a = 1, 2, \ldots, m \tag{1}$$

where the f_a's are distinct points in $\mathbb{R}$. It may be seen from (1) that $l_a(f)$ is an m-1st order polynomial in f which satisfies the 'interpolation property' (i.e. $l_a(f_b) = \delta_{ab}$, the Kronecker delta). *Note that f may be taken to be ξ, r, r^α, or any other convenient function of ξ.*

† Both 'r' and 'ξ' co-ordinate descriptions are commonly used in the formulation of finite-element properties. The former seems preferable in modelling singularities, whereas the latter is often favoured because quadrature rules are generally tabulated with respect to the interval $[-1, 1]$. In this paper we find it convenient to employ the r-description. Results may be translated to the ξ-description via change of variables.

Algorithm for constructing interpolation functions

Consider an n-node finite element. We assume we are given a set of 'preliminary' shape functions, N_a, $a = 1, 2, \ldots, n$, which satisfy the interpolation property on the first m nodes only; viz. $N_a(\mathbf{r}_b) = \delta_{ab}$, where $a, b = 1, 2, \ldots, m < n$. The idea is to systematically modify the N_a's such that the interpolation property is satisfied on all n nodes. The procedure is given as follows:†

Step 1
$$N_{m+1}(\mathbf{r}) \leftarrow \frac{N_{m+1}(\mathbf{r}) - \sum_{a=1}^{m} N_{m+1}(\mathbf{r}_a) N_a(\mathbf{r})}{N_{m+1}(\mathbf{r}_{m+1}) - \sum_{a=1}^{m} N_{m+1}(\mathbf{r}_a) N_a(\mathbf{r}_{m+1})} \tag{2}$$

Step 2
$$N_a(\mathbf{r}) \leftarrow N_a(\mathbf{r}) - N_a(\mathbf{r}_{m+1}) N_{m+1}(\mathbf{r}), \qquad a = 1, 2, \ldots, m \tag{3}$$

Step 3 If $m+1 < n$, replace m by $m+1$ and repeat steps 1–3.
If $m+1 = n$, stop.

Remarks

I. After completing steps 1 and 2, the shape functions satisfy the interpolation property on the first $m+1$ nodes. Clearly, after completing the entire procedure, the desired properties are achieved.

II. In many cases, the algorithm enables the explicit hand calculation of shape functions. *However, there is no need to do this in practice, as the algorithm itself may be programmed as part of a shape function subroutine.*‡

III. The derivatives of shape functions may be constructed in similar fashion. The analogues of steps 1 and 2 are (respectively)

$$\partial N_{m+1}(r) \leftarrow \frac{\partial N_{m+1}(\mathbf{r}) - \sum_{a=1}^{m} N_{m+1}(\mathbf{r}_a)\, \partial N_a(\mathbf{r})}{N_{m+1}(\mathbf{r}_{m+1}) - \sum_{a=1}^{m} N_{m+1}(\mathbf{r}_a) N_a(\mathbf{r}_{m+1})} \tag{4}$$

and

$$\partial N_a(\mathbf{r}) \leftarrow \partial N_a(\mathbf{r}) - N_a(\mathbf{r}_{m+1})\, \partial N_{m+1}(\mathbf{r}) \tag{5}$$

In (4) and (5), '∂' represents the partial differentiation operator with respect to any of the co-ordinates (e.g. in three dimensions, ∂ may represent $\partial/\partial r$, $\partial/\partial s$ or $\partial/\partial t$).

IV. Interpolation problems are generally formulated in terms of a system of simultaneous linear equations. It may be noted that, if $m = 1$, the algorithm encompassed by steps 1–3 solves the interpolation problem *without* engendering any coefficient matrix. Thus no 'storage' is required (beyond that for the N_a's) in programming the procedure. Another nice feature of the algorithm is that the interpolation property possessed by the preliminary shape functions is fully exploited in that only $n - m$ passes through steps 1–3 need be performed.

In practice, the Lagrange polynomial formula may be employed to generate a partial set of preliminary shape functinns which satisfies the interpolation property on the first m nodes. This will be illustrated in the following examples.

EXAMPLES

1. Consider a one-dimensional three-node element with nodes given by $r_1 = 0$, $r_2 = \frac{1}{2}$ and $r_3 = 1$. We wish to construct shape functions capable of exactly representing 1, r and r^α. (If $\alpha = 2$, we

† The arrow, $\leftarrow$, reads: 'is replaced by'.

‡ By a 'shape function subroutine', we mean a subprogram which, upon given a value of $\mathbf{r}$, returns the N_a's and/or derivatives of the N_a's, evaluated at $\mathbf{r}$.

have the usual three-node quadratic element. Other values of α enable the modelling of singularities.) We begin with the following preliminary shape functions:

$$N_1(r) = 1-2r; \qquad N_2(r) = 2r; \qquad N_3(r) = r^\alpha \tag{6}$$

Observe that $N_a(r_b) = \delta_{ab}$, $1 \leq a, b \leq 2$. In terms of our algorithm, $m = 2$ and $n = 3$, so only one pass through steps 1 and 2 is required:

$$N_3(r) \leftarrow \frac{N_3(r) - N_3(\frac{1}{2})N_2(r)}{N_3(1) - N_3(\frac{1}{2})N_2(1)} = \frac{r^\alpha - 2(\frac{1}{2})^\alpha r}{1-2(\frac{1}{2})^\alpha} \tag{7}$$

$$N_1(r) \leftarrow N_1(r) - N_1(1)N_3(r) = 1-2r+\left[\frac{r^\alpha - 2(\frac{1}{2})^\alpha r}{1-2(\frac{1}{2})^\alpha}\right] \tag{8}$$

$$N_2(r) \leftarrow N_2(r) - N_2(1)N_3(r) = 2r-2\left[\frac{r^\alpha - 2(\frac{1}{2})^\alpha r}{1-2(\frac{1}{2})^\alpha}\right] \tag{9}$$

Likewise, the derivatives are given by (in this case ∂N is taken to mean $\partial N/\partial r$):

$$\partial N_3(r) \leftarrow \frac{\partial N_3(r) - N_3(\frac{1}{2})\,\partial N_2(r)}{N_3(1) - N_3(\frac{1}{2})N_2(1)} = \frac{\alpha r^{\alpha-1} - 2(\frac{1}{2})^\alpha}{1-2(\frac{1}{2})^\alpha} \tag{10}$$

$$\partial N_1(r) \leftarrow \partial N_1(r) - N_1(1)\,\partial N_3(r) = -2+\left[\frac{\alpha r^{\alpha-1} - 2(\frac{1}{2})^\alpha}{1-2(\frac{1}{2})^\alpha}\right] \tag{11}$$

$$\partial N_2(r) \leftarrow \partial N_2(r) - N_2(1)\,\partial N_3(r) = 2-2\left[\frac{\alpha r^{\alpha-1} - 2(\frac{1}{2})^\alpha}{1-2(\frac{1}{2})^\alpha}\right] \tag{12}$$

2. Consider a one-dimensional four-node element for which the nodes are given by $r_1 = 0$, $r_2 = \frac{1}{3}$, $r_3 = \frac{2}{3}$ and $r_4 = 1$. We wish to construct shape functions capable of exactly representing 1, r^α, $r^{2\alpha}$ and r^β. The following special cases are of practical importance:

$\alpha = 1, \beta = 3$:	$1, r, r^2, r^3$	cubic polynomial in r
$\alpha = 1, \beta$ arbitrary:	$1, r, r^2, r^\beta$	quadratic polynomial in r plus linear polynomial in r^β
α arbitrary, $\beta = 1$:	$1, r, r^\alpha, r^{2\alpha}$	linear polynomial in r plus quadratic polynomial in r^α
α arbitrary, $\beta = 3\alpha$:	$1, r^\alpha, r^{2\alpha}, r^{3\alpha}$	cubic polynomial in r^α

To obtain a starting set of shape functions, we may employ the Lagrange interpolation formula with $f = r^\alpha$ and $m = 3$, namely

$$N_1(r) = l_1(r^\alpha) = \frac{(r^\alpha - r_2^\alpha)(r^\alpha - r_3^\alpha)}{(r_1^\alpha - r_2^\alpha)(r_1^\alpha - r_3^\alpha)} = \frac{(3^\alpha r^\alpha - 1)(3^\alpha r^\alpha - 2^\alpha)}{2^\alpha} \tag{13}$$

$$N_2(r) = l_2(r^\alpha) = \frac{(r^\alpha - r_1^\alpha)(r^\alpha - r_3^\alpha)}{(r_2^\alpha - r_1^\alpha)(r_2^\alpha - r_3^\alpha)} = \frac{3^\alpha r^\alpha (3^\alpha r^\alpha - 2^\alpha)}{(1-2^\alpha)} \tag{14}$$

$$N_3(r) = l_3(r^\alpha) = \frac{(r^\alpha - r_1^\alpha)(r^\alpha - r_2^\alpha)}{(r_3^\alpha - r_1^\alpha)(r_3^\alpha - r_2^\alpha)} = \frac{3^\alpha r^\alpha (3^\alpha r^\alpha - 1)}{2^\alpha (2^\alpha - 1)} \tag{15}$$

It may be verified that $N_a(r_b) = \delta_{ab}$ for all $a, b = 1, 2, 3$. The last preliminary shape function is taken to be

$$N_4(r) = r^\beta \tag{16}$$

With regard to the algorithm, $m = 3$ and $n = 4$, so only one pass through steps 1 and 2 is needed, namely

$$N_4(r) \leftarrow \frac{N_4(r) - \sum_{a=2}^{3} N_4(r_a) N_a(r)}{1 - \sum_{a=2}^{3} N_4(r_a) N_a(1)} \tag{17}$$

$$N_a(r) \leftarrow N_a(r) - N_a(1) N_4(r), \qquad a = 1, 2, 3 \tag{18}$$

Equations (17) and (18) may be programmed in a shape function subroutine, along with corresponding expressions for derivatives.

3. As a final one-dimensional example, we consider a five-node element for which the nodes are given by $r_1 = 0$, $r_2 = \frac{1}{4}$, $r_3 = \frac{1}{2}$, $r_4 = \frac{3}{4}$ and $r_5 = 1$. The shape functions are to be capable of exactly representing 1, r^α, $r^{2\alpha}$, r^β and r^γ. The following special cases are of practical importance:

$\alpha = 1, \beta = 3, \gamma = 4$:	$1, r, r^2, r^3, r^4$	quartic polynomial in r
$\alpha = 1, \beta = 3, \gamma$ arbitrary:	$1, r, r^2, r^3, r^\gamma$	cubic polynomial in r plus linear polynomial in r^γ.
$\alpha = 1$, $\beta =$ arbitrary, $\gamma = 2\beta$:	$1, r, r^2, r^\beta, r^{2\beta}$	quadratic polynomial in r plus quadratic polynomial in r^β
α arbitrary, $\beta = 3\alpha$, $\gamma = 1$:	$1, r, r^\alpha, r^{2\alpha}, r^{3\alpha}$	linear polynomial in r plus cubic polynomial in r^α
α arbitrary, $\beta = 3\alpha$, $\gamma = 4\alpha$:	$1, r^\alpha, r^{2\alpha}, r^{3\alpha}, r^{4\alpha}$	quartic polynomial in r^α

As in the previous example, the most appropriate choice for a set of starting shape functions is given by (13)–(16) and

$$N_5(r) = r^\gamma \tag{19}$$

This time $m = 3$ and $n = 5$, so two passes through steps 1–3 are required. The following relations need be executed, in turn, for $m = 3$ and $m = 4$:

$$N_{m+1}(r) \leftarrow \frac{N_{m+1}(r) - \sum_{a=2}^{m} N_{m+1}(r_a) N_a(r)}{N_{m+1}(r_{m+1}) - \sum_{a=2}^{m} N_{m+1}(r_a) N_a(r_{m+1})} \tag{20}$$

$$N_a(r) \leftarrow N_a(r) - N_a(r_{m+1}) N_{m+1}(r), \qquad a = 1, 2, \ldots, m \tag{21}$$

The above expressions, and their counterparts for derivatives, may be programmed in shape function subroutines.

Remarks

V. The preceding one-dimensional examples cover most cases of practical interest. If the need arises, higher order shape functions may be constructed along similar lines. The basic philosophy is to maximize m so as to minimize calculations in the shape function routines.

VI. For completeness, we wish to mention the simplest one-dimensional shape functions which allow the modelling of singularities; namely the two-node element with $r_1 = 0$, $r_2 = 1$, $N_1(r) = 1 - r^\alpha$ and $N_2(r) = r^\alpha$.

VII. Let $u(r) = \sum_{a=1}^{n} N_a(r) d_a$, where (by construction) $d_a = u(r_a)$. The function u is capable of exactly representing the powers of r which were used in deriving the N_a's. (The d_a's need only be set accordingly.) In practice, it is usually desired that the same powers of the physical

co-ordinate, $x = x(r)$, be representable. This will be attained if $x = x(r)$ is affine (i.e. of the form $x(r) = c_1 + c_2 r$). There are two ways of achieving this in practice:

The *first* depends upon 1 and r being present among the powers of r used in deriving the shape functions. In this case the isoparametric concept may be invoked; i.e. we take $x(r) = \sum_{a=1}^{n} N_a(r) x_a$, and equally space the nodal co-ordinates (i.e. x_a's in x-space).

The *second* procedure does not require that 1 and r be present, but involves an additional set of shape function calculations, so it is less efficient. In this case, we take $x(r) = \sum_{a=1}^{n} P_a(r) x_a$, where the P_a's are the standard polynomial shape functions, and the x_a's are, again, equally spaced.

To see the necessity of adopting the second procedure when 1 and r are absent from the N_a's, we need only consider the two-node element of Remark VI. If the isoparametric concept is adopted, then u varies linearly with x (rather than x^α). To see this we note that if $x_2 \neq x_1$ then we can write $d_a = c_1 + c_2 x_a$, where the c's are constants. Consequently

$$\begin{aligned} u &= \sum_{a=1}^{2} N_a d_a \\ &= \sum_{a=1}^{2} N_a (c_1 + c_2 x_a) \\ &= c_1 \left(\sum_{a=1}^{2} N_a \right) + c_2 \left(\sum_{a=1}^{2} N_a x_a \right) \\ &= c_1 + c_2 x \quad \blacksquare \end{aligned} \tag{22}$$

We shall now employ the above results in deriving two-dimensional finite element shape functions for modelling line and point singularities.

Examples

4. Consider the four-node quadrilateral illustrated in Figure 1(a). The shape functions may be constructed from products of linear polynomial shape functions in the s-direction and the

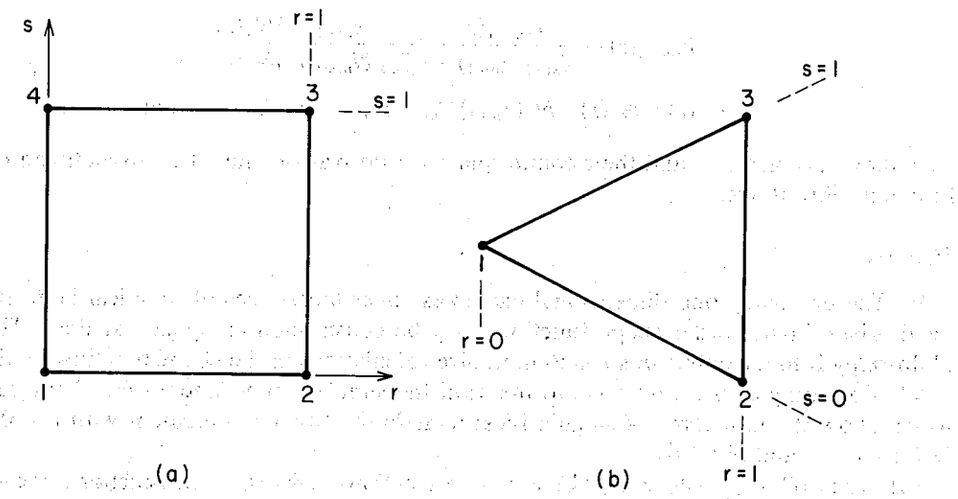


Figure 1. (a) Four-node quadrilateral elements. (b) Three-node triangular element formed by degenerating the four-node quadrilateral

shape functions of Remark VI in the r-direction, namely

$$N_1(r, s) = (1-r^\alpha)(1-s) \tag{23}$$

$$N_2(r, s) = r^\alpha(1-s) \tag{24}$$

$$N_3(r, s) = r^\alpha s \tag{25}$$

$$N_4(r, s) = (1-r^\alpha)s \tag{26}$$

Thus $u(r, s) = \sum_{a=1}^{4} N_a(r, s)d_a$ is capable of exactly representing a line singularity of order r^α along edge $\widehat{14}$. By virtue of Remark VII the standard polynomial shape functions need be employed to define the geometry. That is $\mathbf{x}(r, s) = \sum_{a=1}^{4} P_a(r, s)\mathbf{x}_a$, where

$$P_1(r, s) = (1-r)(1-s) \tag{27}$$

$$P_2(r, s) = r(1-s) \tag{28}$$

$$P_3(r, s) = rs \tag{29}$$

$$P_4(r, s) = (1-r)s \tag{30}$$

This element was first proposed by Akin.[15]

5. A three-node triangular element, see Figure 1(b), may be developed by combining the shape functions associated with nodes 3 and 4 in the previous example, namely

$$N_1(r, s) \leftarrow N_1(r, s) + N_4(r, s) = 1 - r^\alpha \tag{31}$$

$$P_1(r, s) \leftarrow P_1(r, s) + P_4(r, s) = 1 - r \tag{32}$$

The shape functions of nodes 2 and 3 remain the same. This element is capable of exactly representing a point singularity of order r^α. Similar interpolations were used in References 9 and 16.

The procedure employed in this example is often referred to as *element degeneration.*

6. Consider the nine-node Lagrange-type quadrilateral illustrated in Figure 2(a). The shape functions shall be constructed from products of the three-node one-dimensional shape functions

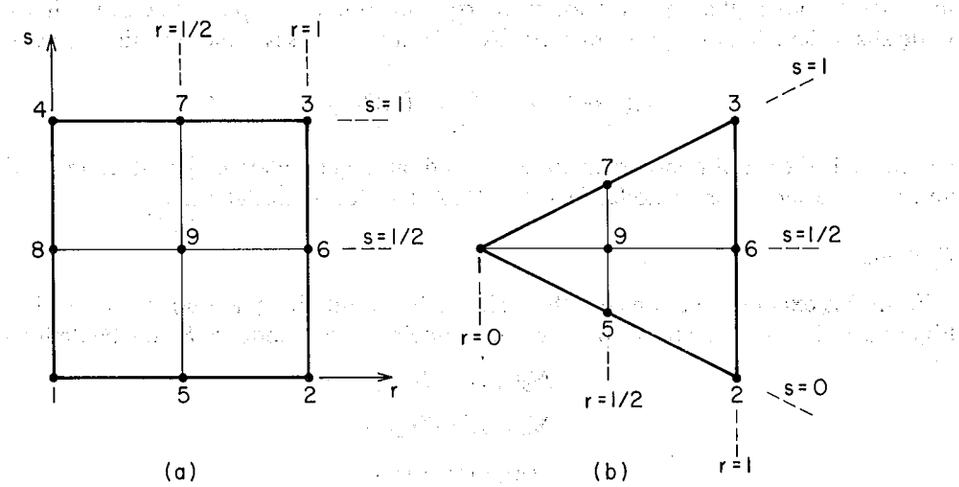


Figure 2. (a) Nine-node Lagrange quadrilateral element. Node 9 is omitted for the eight-node serendipity quadrilateral. (b) Seven-node triangular element formed by degenerating the nine-node Lagrange quadrilateral. In the six-node triangle, node 9 is omitted

of Example 1. That is, we define $N_a(r)$, $a = 1, 2, 3$, by (7)–(9); and $P_a(s)$ by the same expressions with r replaced by s and $\alpha = 2$. The two-dimensional shape functions may then be defined as follows:

$$N_1(r, s) = N_1(r)P_1(s) \tag{33}$$

$$N_2(r, s) = N_3(r)P_1(s) \tag{34}$$

$$N_3(r, s) = N_3(r)P_3(s) \tag{35}$$

$$N_4(r, s) = N_1(r)P_3(s) \tag{36}$$

$$N_5(r, s) = N_2(r)P_1(s) \tag{37}$$

$$N_6(r, s) = N_3(r)P_2(s) \tag{38}$$

$$N_7(r, s) = N_2(r)P_3(s) \tag{39}$$

$$N_8(r, s) = N_1(r)P_2(s) \tag{40}$$

$$N_9(r, s) = N_2(r)P_2(s) \tag{41}$$

In this case $u(r, s) = \sum_{a=1}^{9} N_a(r, s)d_a$ is capable of exactly representing a line singularity of order r^α along edge $\overline{14}$. Furthermore, the following monomials may be exactly represented: $1, r, s, r^\alpha, rs, s^2, r^\alpha s, s^2 r$ and $r^\alpha s^2$. By virtue of the presence of 1, r and s, either the isoparametric concept may be employed to define the geometry (i.e. $\mathbf{x}(r, s) = \sum_{a=1}^{9} N_a(r, s)\mathbf{x}_a$), or recourse may be made to the standard Lagrange polynomials (i.e. $\mathbf{x}(r, s) = \sum_{a=1}^{9} P_a(r, s)\mathbf{x}_a$, where the $P_a(r, s)$'s may be obtained from (33)–(41) by replacing $N_a(r)$ by $P_a(r)$, $a = 1, 2, 3$).

Remark

VIII. Serendipity-type nodal patterns for quadrilaterals are the same as Lagrange nodal patterns, except that all internal nodes are omitted. Our procedure for developing shape functions for serendipity-type elements is to first establish shape functions for the non-corner nodes which satisfy the interpolation property (it turns out that this may easily be done), then establish a preliminary set of shape functions for the four corner nodes which are systematically modified to satisfy the interpolation property. For example, suppose the preliminary corner-node shape functions are given by (23)–(26). Then the necessary modifications may be written

$$N_a(\mathbf{r}) \leftarrow N_a(\mathbf{r}) - \sum_{b \in B_a} N_a(\mathbf{r}_b)N_b(\mathbf{r}), \qquad a \in C \tag{42}$$

where B_a is the set of non-corner nodes located on the boundary edges which meet at node a, and C is the set of corner nodes (e.g. $C = \{1, 2, 3, 4\}$ for quadrilaterals).

Examples

7. In this example, we consider the eight-node serendipity quadrilateral in which node 9 of Figure 2(a) is omitted. The shape functions associated with nodes 5–8 may be defined by

$$N_5(r, s) = N_2(r)(1 - s) \tag{43}$$

$$N_6(r, s) = r^\alpha P_2(s) \tag{44}$$

$$N_7(r, s) = N_2(r)s \tag{45}$$

$$N_8(r, s) = (1 - r^\alpha)P_2(s) \tag{46}$$

In (43)–(46), $N_2(r)$ and $P_2(s)$ are defined as in the previous example. The shape functions for nodes 1–4 may be defined by (42), where the preliminary corner-node shape functions are given by (23)–(26); explicitly:

$$N_1(r,s) \leftarrow N_1(r,s) - \tfrac{1}{2}N_8(r,s) - \left(1 - \frac{1}{2^\alpha}\right) N_5(r,s) \tag{47}$$

$$N_2(r,s) \leftarrow N_2(r,s) - \frac{1}{2^\alpha} N_5(r,s) - \tfrac{1}{2}N_6(r,s) \tag{48}$$

$$N_3(r,s) \leftarrow N_3(r,s) - \tfrac{1}{2}N_6(r,s) - \frac{1}{2^\alpha} N_7(r,s) \tag{49}$$

$$N_4(r,s) \leftarrow N_4(r,s) - \left(1 - \frac{1}{2^\alpha}\right) N_7(r,s) - \tfrac{1}{2}N_8(r,s) \tag{50}$$

The serendipity shape functions defined above are capable of exactly representing the same monomials as the nine-node Lagrange shape functions of Example 6, except for rs^2, which is no longer present.

An alternative definition of serendipity shape functions may be constructed by replacing r^α by r in (23)–(26), (44) and (46). In this case the preliminary corner-node shape functions reduce to the standard bilinear shape functions and (42) yields in place of (47)–(50):

$$N_1(r,s) \leftarrow N_1(r,s) - [N_8(r,s) + N_5(r,s)]/2 \tag{51}$$

$$N_2(r,s) \leftarrow N_2(r,s) - [N_5(r,s) + N_6(r,s)]/2 \tag{52}$$

$$N_3(r,s) \leftarrow N_3(r,s) - [N_6(r,s) + N_7(r,s)]/2 \tag{53}$$

$$N_4(r,s) \leftarrow N_4(r,s) - [N_7(r,s) + N_8(r,s)]/2 \tag{54}$$

This alternative definition of serendipity shape functions is capable of exactly representing the same monomials as the nine-node Lagrange shape functions, except for $r^\alpha s^2$.

This example is typical in that the definition of serendipity shape functions from a given set of one-dimensional shape functions is non-unique. The choice must be made on the basis of the appropriateness of the functions for the intended application.

Again, either the $N_a(r,s)$'s, or the standard polynomial serendipity shape functions, may be used for purposes of defining the geometry. (The polynomial serendipity shape functions may be obtained from (43)–(46) and (51)–(54) by replacing $N_2(r)$ by $P_2(r)$ in (43) and (45) and by replacing r^α by r in (44) and (46).)

8. Seven-node and six-node triangular elements, see e.g. Figure 2(b), for modelling point singularities, may be obtained from the quadrilaterals of Examples 6 and 7, respectively, by combining the shape functions associated with nodes 1, 4 and 8, that is

$$N_1(r,s) \leftarrow N_1(r,s) + N_4(r,s) + N_8(r,s) \tag{55}$$

(If the standard polynomial shape functions are used to define the geometry, they must be similarly combined.) The shape functions associated with the remaining nodes are the same as in Examples 6 and 7. Shape functions N_4 and N_8 play no further role in the calculations, but typically would be set to zero in a shape function subroutine.

9. A 16-node Lagrange quadrilateral, see Figure 3(a), may be formed by taking products of the four-node one-dimensional shape functions of Example 2. In this case, we define $N_a(r)$, $a = 1, 2, 3, 4$, by (13)–(18); and $P_a(s)$ by the same expressions in which $\alpha = 1$ and $\beta = 3$. The results

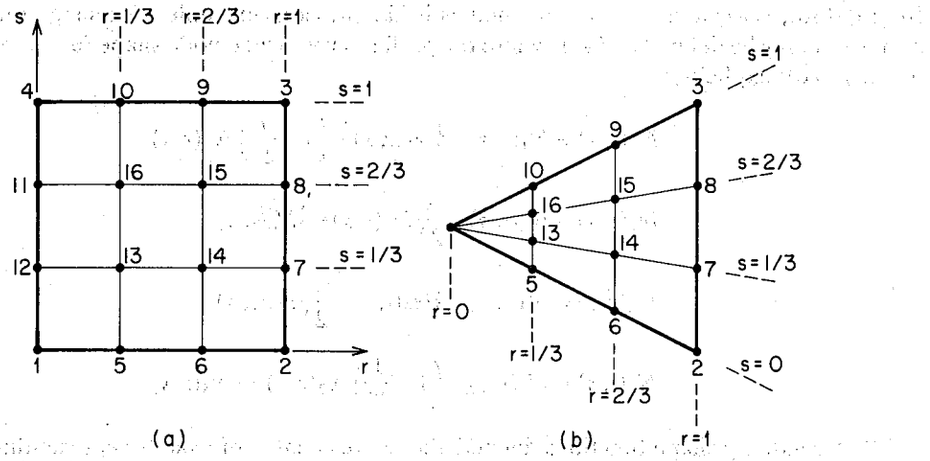


Figure 3. (a) 16-node Lagrange quadrilateral element. Nodes 13–16 are omitted for the 12-node serendipity quadrilateral. (b) 13-node triangular element formed by degenerating the 16-node Lagrange quadrilateral. In the nine-node serendipity triangle, nodes 13–16 are omitted

are

$$N_1(r, s) = N_1(r)P_1(s) \tag{56}$$

$$N_2(r, s) = N_4(r)P_1(s) \tag{57}$$

$$N_3(r, s) = N_4(r)P_4(s) \tag{58}$$

$$N_4(r, s) = N_1(r)P_4(s) \tag{59}$$

$$N_5(r, s) = N_2(r)P_1(s) \tag{60}$$

$$N_6(r, s) = N_3(r)P_1(s) \tag{61}$$

$$N_7(r, s) = N_4(r)P_2(s) \tag{62}$$

$$N_8(r, s) = N_4(r)P_3(s) \tag{63}$$

$$N_9(r, s) = N_3(r)P_4(s) \tag{64}$$

$$N_{10}(r, s) = N_2(r)P_4(s) \tag{65}$$

$$N_{11}(r, s) = N_1(r)P_3(s) \tag{66}$$

$$N_{12}(r, s) = N_1(r)P_2(s) \tag{67}$$

$$N_{13}(r, s) = N_2(r)P_2(s) \tag{68}$$

$$N_{14}(r, s) = N_3(r)P_2(s) \tag{69}$$

$$N_{15}(r, s) = N_3(r)P_3(s) \tag{70}$$

$$N_{16}(r, s) = N_2(r)P_3(s) \tag{71}$$

The monomials present this time are illustrated in Figure 4 by way of a 'Pascal triangle'. As long as either α or β equals 1, the isoparametric concept may be employed to define the geometry. Otherwise, the standard 16-node Lagrange polynomials may be employed. (These may be obtained from (56)–(71) by replacing $N_a(r)$ by $P_a(r)$, $a = 1, 2, 3, 4$.)

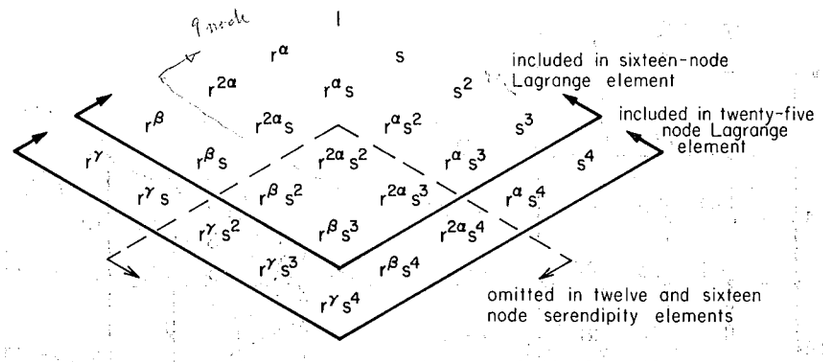


Figure 4. Pascal triangle for Examples 9, 10, 12 and 13

10. A 12-node serendipity quadrilateral in which nodes 13–16 of Figure 3(a) are omitted may be obtained as follows. Define the non-corner node shape functions via

$$N_5(r,s) = N_2(r)(1-s) \tag{72}$$

$$N_6(r,s) = N_3(r)(1-s) \tag{73}$$

$$N_7(r,s) = r^\alpha P_2(s) \tag{74}$$

$$N_8(r,s) = r^\alpha P_3(s) \tag{75}$$

$$N_9(r,s) = N_3(r)s \tag{76}$$

$$N_{10}(r,s) = N_2(r)s \tag{77}$$

$$N_{11}(r,s) = (1-r^\alpha)P_3(s) \tag{78}$$

$$N_{12}(r,s) = (1-r^\alpha)P_2(s) \tag{79}$$

In (72)–(79), $N_2(r)$, $N_3(r)$, $P_2(s)$ and $P_3(s)$ are defined as in the previous example. The shape functions associated with nodes 1–4 may be obtained from (42). The monomials present may be gleaned from Figure 4.

As in Example 7, the one-dimensional shape functions under consideration may be combined differently to form other serendipity-type elements. For example, we may replace r^α by r^β in (23)–(26), (74), (75), (78) and (79). In this case, the monomials present may be deduced from Figure 4 by interchanging r^α and r^β (the $r^{2\alpha}$ terms remain unchanged).

11. Triangles, see Figure 3(b), may likewise be constructed from the results of Examples 9 and 10 by coalescing the nodes along edge $\widehat{14}$. The shape functions need be combined in the usual way:

$$N_1(r,s) \leftarrow N_1(r,s) + N_4(r,s) + N_{11}(r,s) + N_{12}(r,s) \tag{80}$$

and likewise for the polynomial shape functions, if employed to define the geometry. The remaining shape functions are as given in Examples 9 and 10.

Remark

IX. At this point, the algorithm for defining serendipity-type shape functions should be clear from Examples 7 and 10. Likewise, the procedures for defining Lagrange-type quadrilaterals, and for degenerating quadrilaterals into triangles, should also be clear. Consequently, we shall

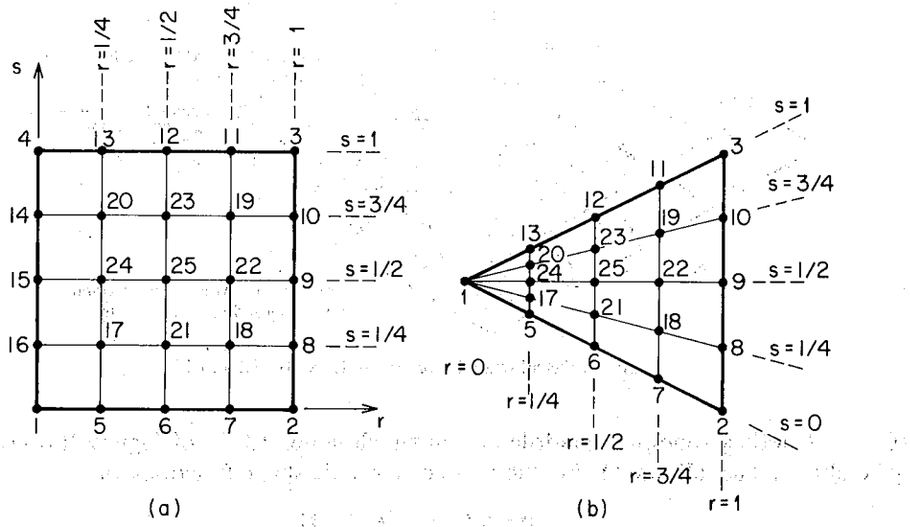


Figure 5. (a) 25-node Lagrange quadrilateral element. Nodes 17–25 are omitted for the 16-node serendipity quadrilateral. (b) 21-node triangular element formed by degenerating the 25-node Lagrange quadrilateral. In the 12-node serendipity triangle, nodes 17–25 are omitted

present without further elaboration the results for the 25-node Lagrange quadrilateral, 16-node serendipity quadrilateral, and corresponding triangular elements (see Figure 5). In these cases, the one-dimensional shape functions of Example 3 are employed.

Examples

12. *25-node Lagrange quadrilateral*

$$N_1(r,s) = N_1(r)P_1(s) \tag{81}$$
$$N_2(r,s) = N_5(r)P_1(s) \tag{82}$$
$$N_3(r,s) = N_5(r)P_5(s) \tag{83}$$
$$N_4(r,s) = N_1(r)P_5(s) \tag{84}$$
$$N_5(r,s) = N_2(r)P_1(s) \tag{85}$$
$$N_6(r,s) = N_3(r)P_1(s) \tag{86}$$
$$N_7(r,s) = N_4(r)P_1(s) \tag{87}$$
$$N_8(r,s) = N_5(r)P_2(s) \tag{88}$$
$$N_9(r,s) = N_5(r)P_3(s) \tag{89}$$
$$N_{10}(r,s) = N_5(r)P_4(s) \tag{90}$$
$$N_{11}(r,s) = N_4(r)P_5(s) \tag{91}$$
$$N_{12}(r,s) = N_3(r)P_5(s) \tag{92}$$
$$N_{13}(r,s) = N_2(r)P_5(s) \tag{93}$$

$$N_{14}(r, s) = N_1(r)P_4(s) \tag{94}$$

$$N_{15}(r, s) = N_1(r)P_3(s) \tag{95}$$

$$N_{16}(r, s) = N_1(r)P_2(s) \tag{96}$$

$$N_{17}(r, s) = N_2(r)P_2(s) \tag{97}$$

$$N_{18}(r, s) = N_4(r)P_2(s) \tag{98}$$

$$N_{19}(r, s) = N_4(r)P_4(s) \tag{99}$$

$$N_{20}(r, s) = N_2(r)P_4(s) \tag{100}$$

$$N_{21}(r, s) = N_3(r)P_2(s) \tag{101}$$

$$N_{22}(r, s) = N_4(r)P_3(s) \tag{102}$$

$$N_{23}(r, s) = N_3(r)P_4(s) \tag{103}$$

$$N_{24}(r, s) = N_2(r)P_3(s) \tag{104}$$

$$N_{25}(r, s) = N_3(r)P_3(s) \tag{105}$$

In the above, the $N_a(r)$'s are defined by (13)–(16) and (19)–(21); the $P_a(s)$'s are the standard quartic polynomials, which may be obtained from the $N_a(r)$'s by replacing r with s, and setting $\alpha = 1$, $\beta = 3$ and $\gamma = 4$.

13. *16-node serendipity quadrilateral*

$$N_5(r, s) = N_2(r)(1-s) \tag{106}$$

$$N_6(r, s) = N_3(r)(1-s) \tag{107}$$

$$N_7(r, s) = N_4(r)(1-s) \tag{108}$$

$$N_8(r, s) = r^\alpha P_2(s) \tag{109}$$

$$N_9(r, s) = r^\alpha P_3(s) \tag{110}$$

$$N_{10}(r, s) = r^\alpha P_4(s) \tag{111}$$

$$N_{11}(r, s) = N_4(r)s \tag{112}$$

$$N_{12}(r, s) = N_3(r)s \tag{113}$$

$$N_{13}(r, s) = N_2(r)s \tag{114}$$

$$N_{14}(r, s) = (1-r^\alpha)P_4(s) \tag{115}$$

$$N_{15}(r, s) = (1-r^\alpha)P_3(s) \tag{116}$$

$$N_{16}(r, s) = (1-r^\alpha)P_2(s) \tag{117}$$

The $N_a(r)$'s and $P_a(s)$'s in (106)–(117) are the same as used in the previous example. Equation (42) may be employed to define $N_a(r, s)$, $a = 1, 2, 3, 4$. Again we wish to point out that alternative definitions of the serendipity shape functions may also be constructed (cf. Examples 7 and 10).

14. *21- and 12-node triangles*

The shape functions of Examples 12 and 13 are combined in the usual way

$$N_1(r, s) \leftarrow N_1(r, s) + N_4(r, s) + N_{14}(r, s) + N_{15}(r, s) + N_{16}(r, s) \tag{118}$$

The remaining functions are defined in Examples 12 and 13. ■

We shall now illustrate, by way of examples, how to use the previously defined two-dimensional shape functions to develop three-dimensional functions for modelling line singularities. We shall assume that the singularity lies along the r-axis.

Examples

15. Consider the wedge-shaped six-node element in Figure 6. Let $N_a(r, s)$, $a = 1, 2, 3$, denote the three-node triangle functions defined in Example 5. Shape functions for the wedge may be

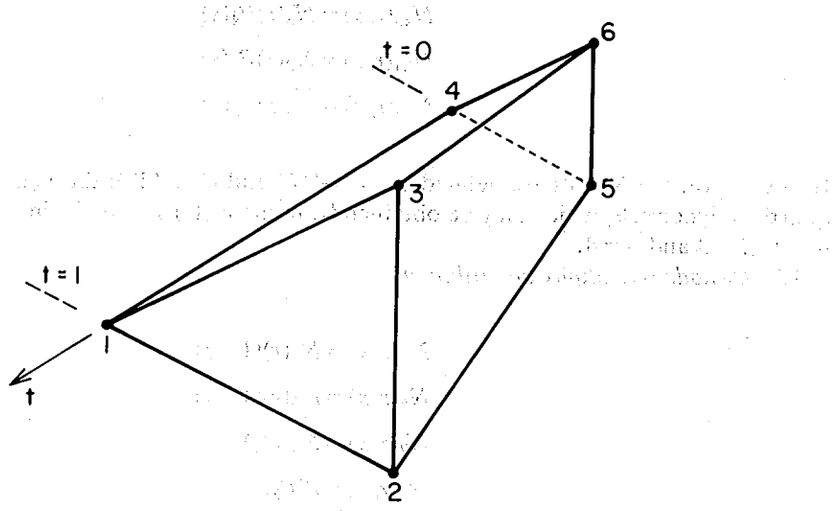


Figure 6. Six-node wedge element

defined as follows:

$$N_1(r, s, t) = N_1(r, s)t \tag{119}$$

$$N_2(r, s, t) = N_2(r, s)t \tag{120}$$

$$N_3(r, s, t) = N_3(r, s)t \tag{121}$$

$$N_4(r, s, t) = N_1(r, s)(1-t) \tag{122}$$

$$N_5(r, s, t) = N_2(r, s)(1-t) \tag{123}$$

$$N_6(r, s, t) = N_3(r, s)(1-t) \tag{124}$$

For the geometric mapping, the $N_a(r, s)$'s in (119)–(124) should be replaced by the standard polynomial shape functions (i.e. $P_a(r, s)$'s in Example 5).

16. Consider the 18-node wedge illustrated in Figure 7(a). Let $\tilde{N}_a(r, s)$, $a = 1, 2, \ldots, 6$, denote the shape functions of the six-node triangle which were defined in Example 8, and are

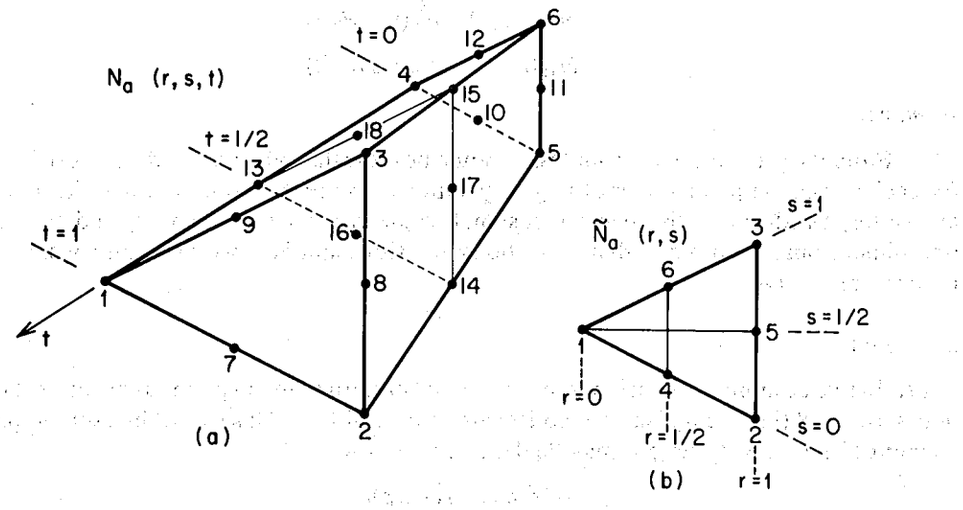


Figure 7. (a) 18-node wedge element. Nodes 16–18 are omitted for the 15-node wedge. (b) Six-node triangular element

renumbered here for convenience—see Figure 7(b). Let $P_a(t)$, $a = 1, 2, 3$, be the standard one-dimensional polynomial shape functions of the three-node element. The three-dimensional shape functions may be defined as follows:

$$N_1(r, s, t) = \tilde{N}_1(r, s)P_3(t) \tag{125}$$
$$N_2(r, s, t) = \tilde{N}_2(r, s)P_3(t) \tag{126}$$
$$N_3(r, s, t) = \tilde{N}_3(r, s)P_3(t) \tag{127}$$
$$N_4(r, s, t) = \tilde{N}_1(r, s)P_1(t) \tag{128}$$
$$N_5(r, s, t) = \tilde{N}_2(r, s)P_1(t) \tag{129}$$
$$N_6(r, s, t) = \tilde{N}_3(r, s)P_1(t) \tag{130}$$
$$N_7(r, s, t) = \tilde{N}_4(r, s)P_3(t) \tag{131}$$
$$N_8(r, s, t) = \tilde{N}_5(r, s)P_3(t) \tag{132}$$
$$N_9(r, s, t) = \tilde{N}_6(r, s)P_3(t) \tag{133}$$
$$N_{10}(r, s, t) = \tilde{N}_4(r, s)P_1(t) \tag{134}$$
$$N_{11}(r, s, t) = \tilde{N}_5(r, s)P_1(t) \tag{135}$$
$$N_{12}(r, s, t) = \tilde{N}_6(r, s)P_1(t) \tag{136}$$
$$N_{13}(r, s, t) = \tilde{N}_1(r, s)P_2(t) \tag{137}$$
$$N_{14}(r, s, t) = \tilde{N}_2(r, s)P_2(t) \tag{138}$$
$$N_{15}(r, s, t) = \tilde{N}_3(r, s)P_2(t) \tag{139}$$
$$N_{16}(r, s, t) = \tilde{N}_4(r, s)P_2(t) \tag{140}$$

$$N_{17}(r, s, t) = \tilde{N}_5(r, s)P_2(t) \tag{141}$$

$$N_{18}(r, s, t) = \tilde{N}_6(r, s)P_2(t) \tag{142}$$

Remark

X. From the previous two examples, it should be clear that when the nodal pattern is the same for each constant-t plane (i.e. is of Lagrange type), then the three-dimensional shape functions are simply products of the two-dimensional shape functions in r and s, and the standard one-dimensional polynomial shape functions in t. Consequently, we shall not consider further patterns of this type.

Examples

17. In this example we consider the same nodal pattern as in the previous example; however, nodes 16 to 18 (i.e. the midsurface nodes) are to be omitted. (This results in a serendipity-type element.) The shape functions may be defined as follows:

$$N_7(r, s, t) = \tilde{N}_4(r, s)t \tag{143}$$

$$N_8(r, s, t) = \tilde{N}_5(r, s)t \tag{144}$$

$$N_9(r, s, t) = \tilde{N}_6(r, s)t \tag{145}$$

$$N_{10}(r, s, t) = \tilde{N}_4(r, s)(1-t) \tag{146}$$

$$N_{11}(r, s, t) = \tilde{N}_5(r, s)(1-t) \tag{147}$$

$$N_{12}(r, s, t) = \tilde{N}_6(r, s)(1-t) \tag{148}$$

$$N_{13}(r, s, t) = \tilde{N}_1(r, s)P_2(t) \tag{149}$$

$$N_{14}(r, s, t) = \tilde{N}_2(r, s)P_2(t) \tag{150}$$

$$N_{15}(r, s, t) = \tilde{N}_3(r, s)P_2(t) \tag{151}$$

In the above, the $N_a(r, s)$'s, $\tilde{N}_a(r, s)$'s and $P_2(t)$ are the same as defined in the previous two examples. A set of preliminary shape functions for the corner nodes are given by

$$N_1(r, s, t) = N_1(r, s)t \tag{152}$$

$$N_2(r, s, t) = N_2(r, s)t \tag{153}$$

$$N_3(r, s, t) = N_3(r, s)t \tag{154}$$

$$N_4(r, s, t) = N_1(r, s)(1-t) \tag{155}$$

$$N_5(r, s, t) = N_2(r, s)(1-t) \tag{156}$$

$$N_6(r, s, t) = N_3(r, s)(1-t) \tag{157}$$

The preceding definitions need to be modified to satisfy the interpolation property. To this end the $N_a(r, s, t)$'s are used in (42), in which $C = \{1, 2, \ldots, 6\}$.

We may again mention that the definition of serendipity-type elements is not unique in the present context and other possibilities arise.

18. Consider the 24-node serendipity-type wedge depicted in Figure 8(a). The three-dimensional shape functions are to be generated from the two-dimensional ones associated with the nodal pattern of Figure 8(b) (cf. Example 11). The non-corner shape functions are defined

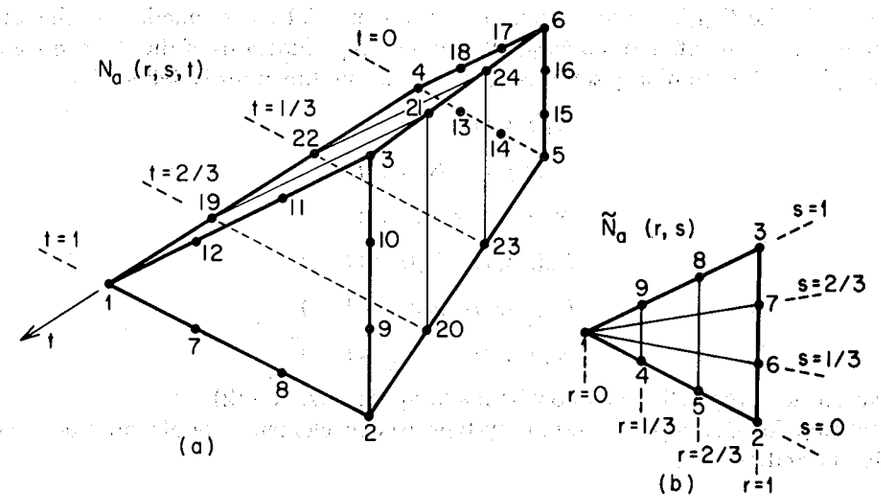


Figure 8. (a) 24-node wedge element. (b) Nine-node triangular element

by:

$$N_7(r, s, t) = \tilde{N}_4(r, s)t \tag{158}$$

$$N_8(r, s, t) = \tilde{N}_5(r, s)t \tag{159}$$

$$N_9(r, s, t) = \tilde{N}_6(r, s)t \tag{160}$$

$$N_{10}(r, s, t) = \tilde{N}_7(r, s)t \tag{161}$$

$$N_{11}(r, s, t) = \tilde{N}_8(r, s)t \tag{162}$$

$$N_{12}(r, s, t) = \tilde{N}_9(r, s)t \tag{163}$$

$$N_{13}(r, s, t) = \tilde{N}_4(r, s)(1-t) \tag{164}$$

$$N_{14}(r, s, t) = \tilde{N}_5(r, s)(1-t) \tag{165}$$

$$N_{15}(r, s, t) = \tilde{N}_6(r, s)(1-t) \tag{166}$$

$$N_{16}(r, s, t) = \tilde{N}_7(r, s)(1-t) \tag{167}$$

$$N_{17}(r, s, t) = \tilde{N}_8(r, s)(1-t) \tag{168}$$

$$N_{18}(r, s, t) = \tilde{N}_9(r, s)(1-t) \tag{169}$$

$$N_{19}(r, s, t) = N_1(r, s)P_3(t) \tag{170}$$

$$N_{20}(r, s, t) = N_2(r, s)P_3(t) \tag{171}$$

$$N_{21}(r, s, t) = N_3(r, s)P_3(t) \tag{172}$$

$$N_{22}(r, s, t) = N_1(r, s)P_2(t) \tag{173}$$

$$N_{23}(r, s, t) = N_2(r, s)P_2(t) \tag{174}$$

$$N_{24}(r, s, t) = N_3(r, s)P_2(t) \tag{175}$$

In (158)–(175), the $P_a(t)$'s are the standard cubic polynomial shape functions of the four-node one-dimensional element, and the $N_a(r, s)$'s are the shape functions of the three-node triangle (i.e. Example 5). A preliminary set of corner-node shape functions is defined by

$$N_1(r, s, t) = N_1(r, s)t \tag{176}$$

$$N_2(r, s, t) = N_2(r, s)t \tag{177}$$

$$N_3(r, s, t) = N_3(r, s)t \tag{178}$$

$$N_4(r, s, t) = N_1(r, s)(1-t) \tag{179}$$

$$N_5(r, s, t) = N_2(r, s)(1-t) \tag{180}$$

$$N_6(r, s, t) = N_3(r, s)(1-t) \tag{181}$$

Modifications to (176)–(181) may now be made by recourse to (42).

The scheme for developing serendipity-type wedge elements should now be clear from Examples 17 and 18.

Remark on implementation

The simplest elements can have their shape functions developed in closed form, as illustrated in the previous section. The automatic generation of these shape functions is complicated by the normalizing terms in the denominators of (2) and (4).

A practical approach is to include one r^β term in addition to the Lagrange family of r^α terms. Then the final function would contain the $1, r^\alpha, \ldots, r^{m\alpha}, r^\beta$ powers. This would allow for standard polynomials as well as the common practical cases cited earlier. Such a procedure yields more efficient computations and does not overly restrict the application of the procedure.

It may also be mentioned that the 'variable-number-of-nodes' concept (see e.g. Reference 17) may be fruitfully employed to unify the implementation of families of special elements.

CONCLUSIONS

In this paper we have constructed a concise and efficient algorithm for generating shape functions from an arbitrary set of independent functions. The algorithm does not involve a coefficient matrix and thus eliminates the burdens of equation solving usually associated with interpolation problems.

The algorithm has been employed to generate one-dimensional shape functions which serve as a basis for modelling singularities. A variety of two- and three-dimensional elements have been constructed, and methodology for developing more general elements of the type considered has been described.

The techniques developed herein may also be applied fruitfully to the development of 'special' finite element interpolatory schemes for other purposes. Areas which may be mentioned in this regard are 'upwind' elements, infinite elements, boundary-layer elements and other singular situations (e.g. ones requiring higher order continuity.)

REFERENCES

1. T. J. R. Hughes, 'A simple scheme for developing "upwind" finite elements', *Int. J. num. Meth. Engng*, **12**, 1359–1365 (1978).
2. T. J. R. Hughes, W. K. Liu and A. Brooks, 'Finite element analysis of incompressible viscous flows by the penalty function formulation', *J. Comp. Phys.* **30**, 1–60 (1979).

3. J. C. Heinrich, P. S. Huyakorn, A. R. Mitchell and O. C. Zienkiewicz, 'An upwind finite element scheme for two-dimensional convective transport equations', *Int. J. num. Meth. Engng*, **11**, 131–143 (1977).
4. D. Gartling and E. B. Becker, 'Computationally efficient finite element analysis of viscous flow problems', in *Computational Methods in Nonlinear Mechanics* (J. T. Oden, Ed.), 1974.
5. P. Bettess, 'Infinite elements', *Int. J. num. Meth. Engng*, **11**, 53–64 (1977).
6. J. E. Akin, 'Physical bases for the design of special finite element interpolation functions', in *Recent Advances in Engineering Science* (G. C. Sih, Ed.), Lehigh University Press, 1977, pp. 879–884.
7. W. S. Blackburn, 'Calculation of stress intensity factors at crack tips using special finite elements', in *Mathematics of Finite Elements and Applications* (J. R. Whiteman, Ed.), Academic Press, 1973, pp. 327–336.
8. R. D. Henshell and K. G. Shaw, 'Crack tip elements are unnecessary', *Int. J. num. Meth. Engng*, **9**, 495–507 (1975).
9. J. E. Akin, 'Generation of elements with singularities', *Int. J. num. Meth. Engng*, **10**, 1249–1259 (1976).
10. R. E. Barnhill and J. R. Whiteman, 'Error analysis of finite element methods with triangles for elliptic boundary value problems', in *Mathematics of Finite Element Methods and Applications* (J. R. Whiteman, Ed.), Academic Press, 1973, pp. 83–112.
11. A. K. Rao, 'Stress concentrations and singularities at interface corners', *ZAMM*, **51**, 395–406 (1971).
12. R. S. Barsoum, 'On the use of isoparametric finite elements in linear fracture mechanics', *Int. J. num. Meth. Engng*, **10**, 25–37 (1976).
13. P. J. Davis, *Interpolation and Approximation*, Blaisdell Press, New York, 1963.
14. M. Stern, 'Families of consistent conforming elements with singular derivative fields', *Int. J. num. Meth. Engng*, **14**, 409–421 (1979).
15. J. E. Akin, 'Elements for problems with line singularities', *Mathematics of Finite Elements and Applications, vol. III* (J. R. Whiteman, Ed.), Academic Press, London, 1978.
16. D. M. Tracey and T. S. Cook, 'Analysis of power type singularities using finite elements', *Int. J. num. Meth. Engng*, **10**, 1249–1259 (1976).
17. K. J. Bathe and E. L. Wilson, *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1976.

LEARN - a Linear Static Finite Element Analysis Program

by

Thomas J. R. Hughes
Division of Engineering and Applied Science
California Institute of Technology
Pasadena, California 91125

---

*Prepared for Ae/AM 108, Finite Element Methods, Fall 1976, California Institute of Technology

Contents

	Page
Introduction	1
Description of Coding Techniques Used in LEARN	3
Adding New Elements to LEARN	18
Users' Manual	20
Flowchart	21
Data Set-up	26
Individual Data Cards	27
Listing	44
Solved Problems	72
Truss Problem	73
Plane Strain Problem	79

Introduction

This report describes "LEARN," a simple finite element code for linear static analysis. For instructional purposes it may be considered a "first example" in that taping (disk reading and writing) is minimized, and blocking (segmenting large jobs via taping) and overlay structures are avoided. Also the class of capabilities--linear static analysis--is the simplest. Nevertheless many sophisticated procedures are employed (namely compacted column solution and data structures, element routines, etc.) which are used in more advanced situations.

LEARN is used in courses Ae/AM108a-c, Finite Element Methods, and serves the following three purposes:

(1) It is felt that the first step in learning to implement finite element methods is to add a new element to an existing program. In this regard, LEARN is written so that it is very easy to add new elements. (Instructions are given in the section entitled "Adding New Elements to LEARN.") Students in the class are assigned projects, depending on their interests and level of competence in programming, which usually involve adding a new element to LEARN. For this reason, only two sets of element routines are included with the listing so that many standard element types (e.g. 3D and axisymmetric elasticity elements; beam, plate, and shell elements; heat conduction elements; etc.) are available for project assignments. The course lectures provide much additional information regarding coding element routines.

(2) LEARN makes available to the student a good, first, finite element "software package." The subroutines are efficiently and simply coded and, as the student becomes more advanced, may be used as a set of building blocks for more sophisticated applications. With this idea in mind it was decided to include a very efficient equation solver in LEARN which employs compacted column storage mode (see Section 2).

(3) The style of coding in LEARN derives heavily from the "Berkeley school" of finite element programming which the author learned by studying the works of E. L. Wilson, R. L. Taylor and many former colleagues and students, too numerous to mention, in the Division of Structural Engineering and Structural Mechanics, Department of Civil Engineering, University of California, Berkeley. As this style of programming is employed in many large-scale production codes that are used extensively throughout the world, it is felt to be a very worthwhile style for the student to become familiar with.

The reader of this report is forewarned that the level of documentation is not considered self-sufficient (nor is it intended to be) for fully understanding all aspects of the program, but rather as supplementary to the course lectures.

If any errors are detected in the code or documentation, please report them to T. Hughes, 221 Thomas Laboratory, x1232.

Description of Coding Techniques Used in LEARN

To understand the workings of a computer program, and in order to work with a computer program, one must be thoroughly familiar with its data structure.

Definition. The data structure is the mode of storage, location and history of existence of the data employed.

In LEARN all but the element data is stored in core (central memory) in blank common. The basic features of the data structure are:

- compacted column architecture

  This feature is engendered by the method of equation solving employed. The solver is called variously a "profile," "skyline" or "active column" solver. Since, in large problems, the dominant portion of storage is that devoted to the stiffness matrix, $\underset{\sim}{K}$, the storage scheme for $\underset{\sim}{K}$ is the most important feature of the storage.

- dynamic storage allocation

  Because of the variable sizes of the arrays needed in different problems, it would be very inefficient to "fix" the dimensioning. In the dynamic storage allocation concept the dimensions of arrays are set in the program at the time of execution.

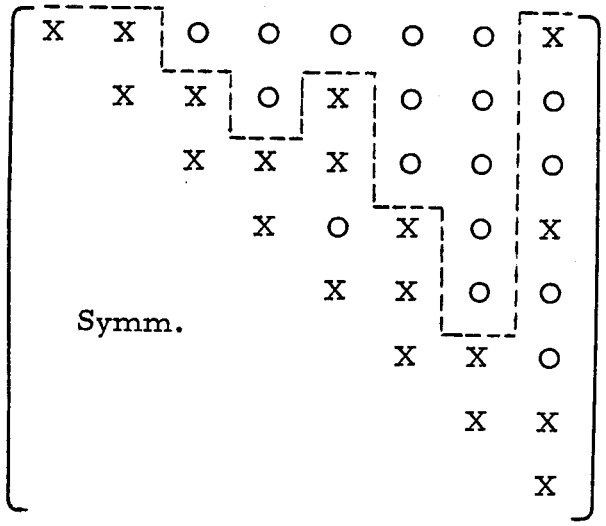
- element groups

Many features of finite element computer programs are the same even though the intended applications may be quite different. To some extent this is also true of the individual finite element subroutines. However, the data structure and options available may vary considerably from one element to another. To accommodate these differences, the elements are read in, stored and operated upon in groups. The data structure for the group is set up via dynamic storage allocation in individual element subroutines.

Example. Compacted column storage scheme.

Assume the symmetric stiffness matrix $\underset{\sim}{K}$ has dimension 8×8 and has zero and nonzero elements indicated by O and X, respectively, in the figure below.

$$\underset{\sim}{K} = [K_{ij}] =$$



The dashed line enveloping the nonzero terms is sometimes called the "profile" or "skyline."

$\underset{\sim}{K}$ is stored in a one-dimensional array, A, column-wise beginning with the first nonzero element in each column and ending with the diagonal term. The locations of the diagonal terms in A are stored in a one-dimensional integer array IDIAG. The dimension of IDIAG is NEQ, the number of equations (e.g., for $\underset{\sim}{K}$ above NEQ=8). The dimension of A is NA, the sum of the "column heights." The column height is the number of terms in a column beginning with the first nonzero term and ending with the diagonal term (e.g., for the above matrix $\underset{\sim}{K}$, column 1 has height 1, column 4 has height 2, column 6 has height 3, etc.) NA for $\underset{\sim}{K}$ is 24. The arrays A and IDIAG corresponding to $\underset{\sim}{K}$ are given explicitly as follows:

$A(1) = K_{11}$ } col. 1

$A(2) = K_{12}$
$A(3) = K_{22}$ } col. 2

$A(4) = K_{23}$
$A(5) = K_{33}$ } col. 3

$A(6) = K_{34}$
$A(7) = K_{44}$ } col. 4

$A(8) = K_{25}$
$A(9) = K_{35}$
$A(10) = K_{45}$
$A(11) = K_{55}$ } col. 5

$A(12) = K_{46}$
$A(13) = K_{56}$
$A(14) = K_{66}$ } col. 6

$A(15)=K_{67}$ } col. 7
$A(16)=K_{77}$

$A(17)=K_{18}$
$A(18)=K_{28}$
$A(19)=K_{38}$
$A(20)=K_{48}$
$A(21)=K_{58}$ } col. 8
$A(22)=K_{68}$
$A(23)=K_{78}$
$A(24)=K_{88}$

IDIAG(1)=1
IDIAG(2)=3
IDIAG(3)=5
IDIAG(4)=7
IDIAG(5)=11
IDIAG(6)=14
IDIAG(7)=16
IDIAG(8)=24

The zero terms beneath the skyline must be stored because they become nonzero during the solution process. All information in the original matrix $\underset{\sim}{K}$ is now contained in the arrays A and IDIAG in compact fashion. For large finite element stiffness matrices the saving of storage is considerable.

The heights of the columns in $\underset{\sim}{K}$ are determined by subroutine COLHT from the node numbers on element data cards. The column

ie LM

heights are initially stored in IDIAG. When the calculation of column heights is completed, subroutine DIAG determines the diagonal addresses and "over writes" them in IDIAG.

The right-hand side vector in

$$\underset{\sim}{K}\underset{\sim}{d} = \underset{\sim}{F}$$

is stored in an array B, of dimension NEQ, in the obvious way:

$$\begin{aligned} B(1) &= F_1 \\ B(2) &= F_2 \\ &\vdots \\ B(8) &= F_8 \end{aligned}$$

The elimination scheme employed is based upon the following theorem.

Theorem. Let $\underset{\sim}{K}$ be symmetric. Then there exists a nonsingular upper triangular matrix $\underset{\sim}{U}$, with unit diagonal entries, and a diagonal matrix $\underset{\sim}{D}$ such that

$$\underset{\sim}{K} = \underset{\sim}{U}^T \underset{\sim}{D} \underset{\sim}{U} \qquad \text{(factorization)}$$

Furthermore, the number of negative, zero and positive eigenvalues of $\underset{\sim}{K}$ is equal to the number of negative, zero and positive diagonal entries of $\underset{\sim}{D}$, respectively.

Remarks. 1. In the case when $\underset{\sim}{K}$ is positive definite, the diagonal entries of $\underset{\sim}{D}$ are all strictly positive. Thus examining the diagonal entries of $\underset{\sim}{D}$ enables us to determine the rank of $\underset{\sim}{K}$. This is the purpose of the "rank check option" in the program.

2. The matrix $\underset{\sim}{U}$ has the same profile as $\underset{\sim}{K}$. The fact that $\underset{\sim}{U}$ has diagonal entries equal to one allows us to save storage by placing the diagonal entries of $\underset{\sim}{D}$ in the diagonal entries of the array A. Thus after factorization the nonzero entries of both $\underset{\sim}{U}$ and $\underset{\sim}{D}$ are stored in array A. During the factorization procedure no auxiliary storage is needed.

Crout elimination

The procedure used for performing the factorization is called Crout elimination, a convenient variant of Gauss elimination. In the Crout algorithm one column at a time is completely factorized, beginning with the first column and not involving subsequent columns. (This feature proves convenient in nonlinear analysis in which only a small zone of the entire effective stiffness is nonlinear. The linear portion can be located in the first columns and factorized once and for all.)

For further information on the Crout elimination algorithm employed see R. L. Taylor, "Computer Procedures for Finite Element Analysis," Chapter 24, in O. C. Zienkiewicz, The Finite Element Method, Third Edition, McGraw-Hill, London, 1977.

Forward reduction and back substitution

After factorization the solution is carried out in three steps

$$\underset{\sim}{U}^T \underset{\sim}{x} = \underset{\sim}{F} \qquad \text{(forward reduction)}$$

$$\underset{\sim\sim}{D}\,\underset{\sim}{y} = \underset{\sim}{x}$$

$$\underset{\sim\sim}{U}\,\underset{\sim}{d} = \underset{\sim}{y} \qquad \text{(back substitution)}$$

In each step the solution is an explicit process, the coefficient matrix being triangular or diagonal. The intermediate vectors $\underset{\sim}{x}$ and $\underset{\sim}{y}$ are in turn stored in the array B. Thus again no additional storage besides that for A and B (and IDIAG) is needed to carry out the factorization and solution process.

Storage in blank common

Except for element group data, which is stored on a disk file and read in and out of core as needed, all of the main arrays are stored in blank common. This includes both integer and floating point data. In the main program and subroutine LEARN, blank common is denoted by the one-dimensional array A.

The locations and lengths of arrays in blank common are as follows:

Storage in Blank Common - Solution Phase

First Word	Array
N1 = 1	ID(NDOF, NUMNP)
N2 = N1 + NDOF*NUMNP	X(NSD, NUMNP)
N3 = N2 + NSD*NUMNP*IPREC	F(NDOF, NUMNP) (=D(NDOF, NUMNP))
N4 = N3 + NDOF*NUMNP*IPREC	IDIAG(NEQ)
N5* = N4 + NEQ	B(NEQ)
N6 = N5 + NEQ*IPREC	A(NA)
N7 = N6 + NA*IPREC	E(MAX)

N8 = N7 + MAX-1 (= total required storage; must be .LT. MTOT for execution.)

*During input phase, N1 to N5 are the same as above and element group data begins in N5. Individual element routines determine necessary storage for element groups.

where

ID(M,N)	=	equation number for degree of freedom M, node N
X(M,N)	=	Mth coordinate of node N
F(M,N) D(M,N)	=	Mth direction force/displacement component of node N
IDIAG(N)	=	location of Nth diagonal element in coefficient array A.*
B(N)	=	Nth force/displacement component
A(N)	=	Nth element of coefficient matrix stored in compacted column form
E(N)	=	Nth (single precision) word of element group array
NUMNP	=	number of nodes
NSD	=	number of space dimensions
NDOF	=	number of degrees of freedom
NEQ	=	number of equations
NA	=	number of terms in A
MAX	=	number of single-precision words in longest element group data file

*This array name is used for the coefficient matrix in various subroutines and should not be confused with the array A denoting blank common.

IPREC* = { 1 single-precision floating point data used throughout program
2 double-precision floating point data used throughout program

MTOT* = total length of blank common available

The above names are the ones most commonly used in individual subroutines for the arrays in question. During the input and solution phases different data than that indicated above may be present in some of the arrays to conserve storage.

---

*Set in the main program. Some FORTRAN compilers allow MTOT to be set internally by the program during execution. However, this is not a feature of standard FORTRAN.

Dynamic storage allocation

Examples

In subroutine LEARN (the driving program) storage is allocated for several of the main arrays. For example, on lines 014 to 016 the "pointers" are determined for the ID and X arrays. The calling statements

```
CALL COORD (A(N2),NSD,NUMNP)
```

and

```
CALL BC      (A(N1),NDOF,NUMNP,NEQ)
```

and corresponding subroutine statements

```
SUBROUTINE COORD(X,NSD,NUMNP)
```

and

```
SUBROUTINE BC (ID,NDOF,NUMNP,NEQ)
```

respectively, mean array X begins in address N2 of blank common and array ID begins in address N1(=1)of blank common.

In COORD and BC, the arrays X and ID are treated as two-dimensional, as evidenced by the DIMENSION statements, viz.

```
DIMENSION X (NSD, 1)
```

and

```
DIMENSION ID (NDOF, 1),
```

respectively. (It does not matter that X and ID correspond to portions of a one-dimensional array in the calling program.) The significance

of the dimensioning is as follows: The arrays X and ID are stored in blank common in consecutively addressed cells (words) by columns. Consequently the addresses of X(M,N) and ID(M,N) in A are given by (resp.)

```
IPREC*(NSD*(N-1)+M-1)  +  N2
```

and

```
NDOF*(N-1)+M-1  +  N1
```

Thus the only information necessary to determine the appropriate addresses in A are the starting addresses (i.e. N2 and N1, resp.), which are provided by the CALL statements, and the number of rows (i.e. NSD and NDOF, resp.), which are provided by the DIMENSION statements in the subroutines. This explains why it is not necessary to define the number of columns of two-dimensional arrays "passed" to subroutines. The 1's in the column slots of the DIMENSION statements could be replaced by any positive integer. This only serves notice that these arrays are to be regarded as two-dimensional in the subroutines--the integer is never used.

In the case of three-dimensional arrays, the first two dimensions must be precisely defined, whereas the third one may be set to 1.

Storage of element group data

The individual element routines determine the data storage for element groups. A description of the storage for the truss and plane elements follows:

Storage Requirements of Truss Element (NTYPE=1)

First Word	Array
N101 = NF*	E(NUMAT)
N102 = N101 + NUMAT*IPREC	AREA(NUMAT)
N103 = N102 + NUMAT*IPREC	WT(NUMAT)
N104 = N103 + NUMAT*IPREC	GRAV(3)
N105 = N104 + 3*IPREC	MAT(NUMEL)
N106 = N105 + NUMEL	IEN(2,NUMEL)
N107 = N106 + 2*NUMEL	LM(3,2,NUMEL)
NL = N107 + 6*NUMEL	

LENGTH = NL - NF (=total required storage for element group)

where

E(N) = Young's modulus for material number N

AREA(N) = cross-section area for material number N

*NF = N5 during input phase and NF = N7 during solution phase.

WT(N)	=	weight per unit length for material number N
GRAV(N)	=	multiplier of gravity load in Nth coordinate direction
MAT(N)	=	material number for element number N
IEN(M,N)	=	Mth node of element number N
LM(L,M,N)	=	equation number for Lth degree of freedom, Mth node, element N
NUMAT	=	number of geometric/material property sets for this group

Storage Requirements of Plane Stress/Strain Element (NTYPE=3)

First Word	Array
N101 = NF*	E(NUMAT)
N102 = N101 + NUMAT*IPREC	POIS(NUMAT)
N103 = N102 + NUMAT*IPREC	WT(NUMAT)
N104 = N103 + NUMAT*IPREC	TH(NUMAT)
N105 = N104 + NUMAT*IPREC	GRAV(2)
N106 = N105 + 2*IPREC	MAT(NUMEL)
N107 = N106 + NUMEL	IEN(4,NUMEL)
N108 = N107 + 4*NUMEL	LM(2,4,NUMEL)
N109 = N108 + 8*NUMEL	IELNO(NUMPR)
N110 = N109 + NUMPR	ISIDE(NUMPR)
N111 = N110 + NUMPR	PRES(NUMPR)
NL = N111 + NUMPR*IPREC	

LENGTH = NL - NF (=total required storage for element group)

The meaning of the arrays here may be deduced from the description in the section entitled "User's Manual."

*NF = N5 during input phase and NF = N7 during solution phase.

Adding an element to the program

Example Suppose we wish to add a shell element to the program. Let us call the element routine which sets up element group storage "SHELL."

The steps are as follows:

0. Study and understand thoroughly the subroutines TRUSS, TRUSSI, PLANE and PLANE1 before beginning.

1. Code a new version of subroutine ELMLIB to include a call to SHELL. The necessary modifications are indicated below:

```
      GO TO (1, 2, 3, 4                    ), I
1     CALL TRUSS
      RETURN
2     CALL BEAM
      RETURN
3     CALL PLANE
      RETURN
4     CALL SHELL
      RETURN
C
      END
```

2. Write a description for the User's Manual to be included in Section 6.0, ELEMENT DATA. Mimic the style of Sections 6.1 and 6.2, and use the same names of variables as far as possible.

3. Code SHELL in the style of TRUSS and PLANE. (These are the routines which set the element dynamic storage allocation.)

4. Code subroutine SHELL1 in the style of TRUSS1 and PLANE1. Take care to code attractive and complete output data formats.

5. Code any additional subroutines needed which are not already provided in the program.

Instructions for debugging* and running the new element

1. Compile all your new routines separately before including them in LEARN.

2. After all routines have compiled, run by loading your new routines with the data for an analysis. You can use the same job control cards as used in running LEARN. (In this way you will not change or destroy the program on permanent file.)

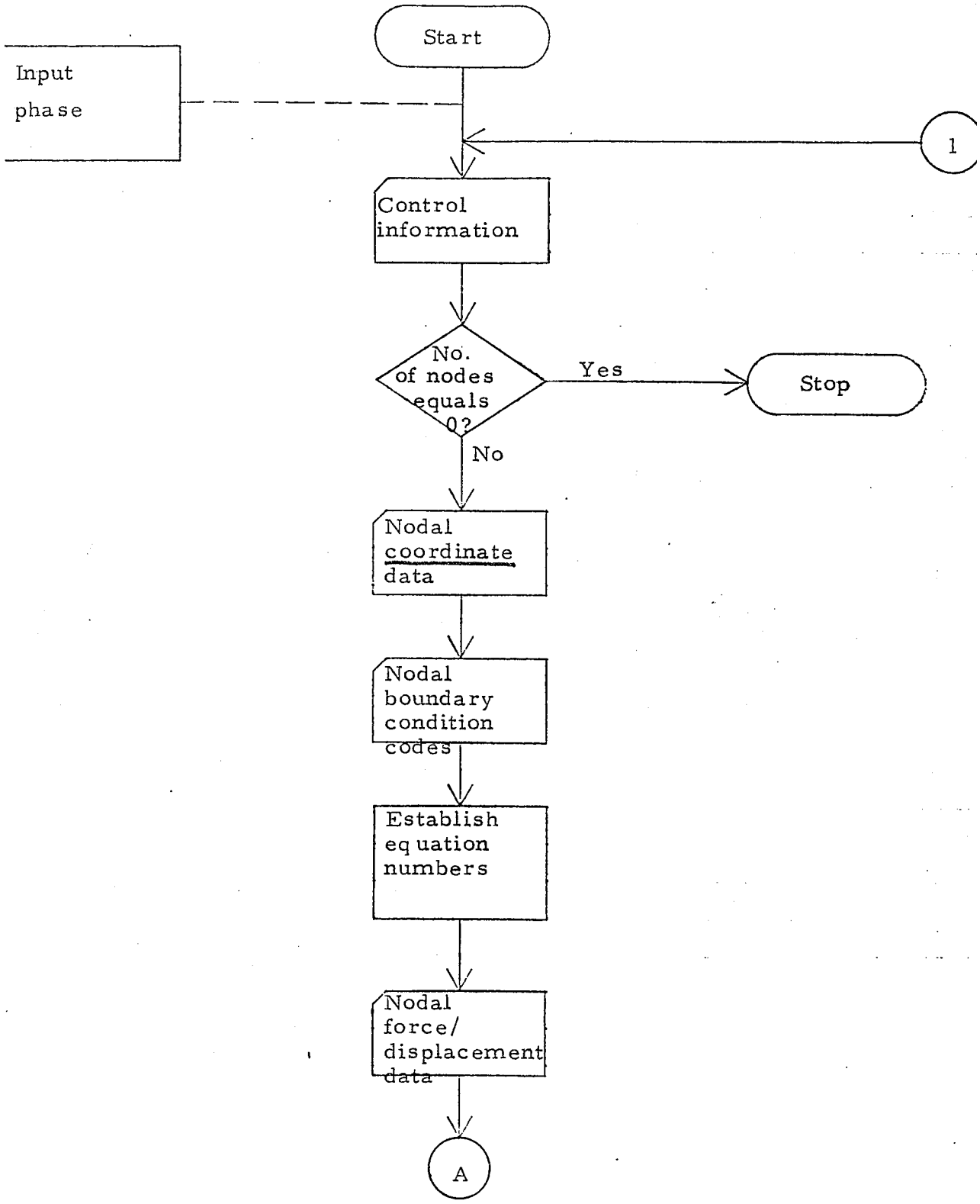
3. Initially you should run simple, one-element problems (e.g. homogeneous deformation states) which are easy to check. Check out all options (e.g., gravity loads, etc.) on the one element problems. Next proceed to "patch tests," which also should be easy to check. Finally, after you are convinced your element is working properly, try some interesting test problems for which you have exact analytical, or reliable experimental, data. Assess the accuracy of the element you have programmed.

---

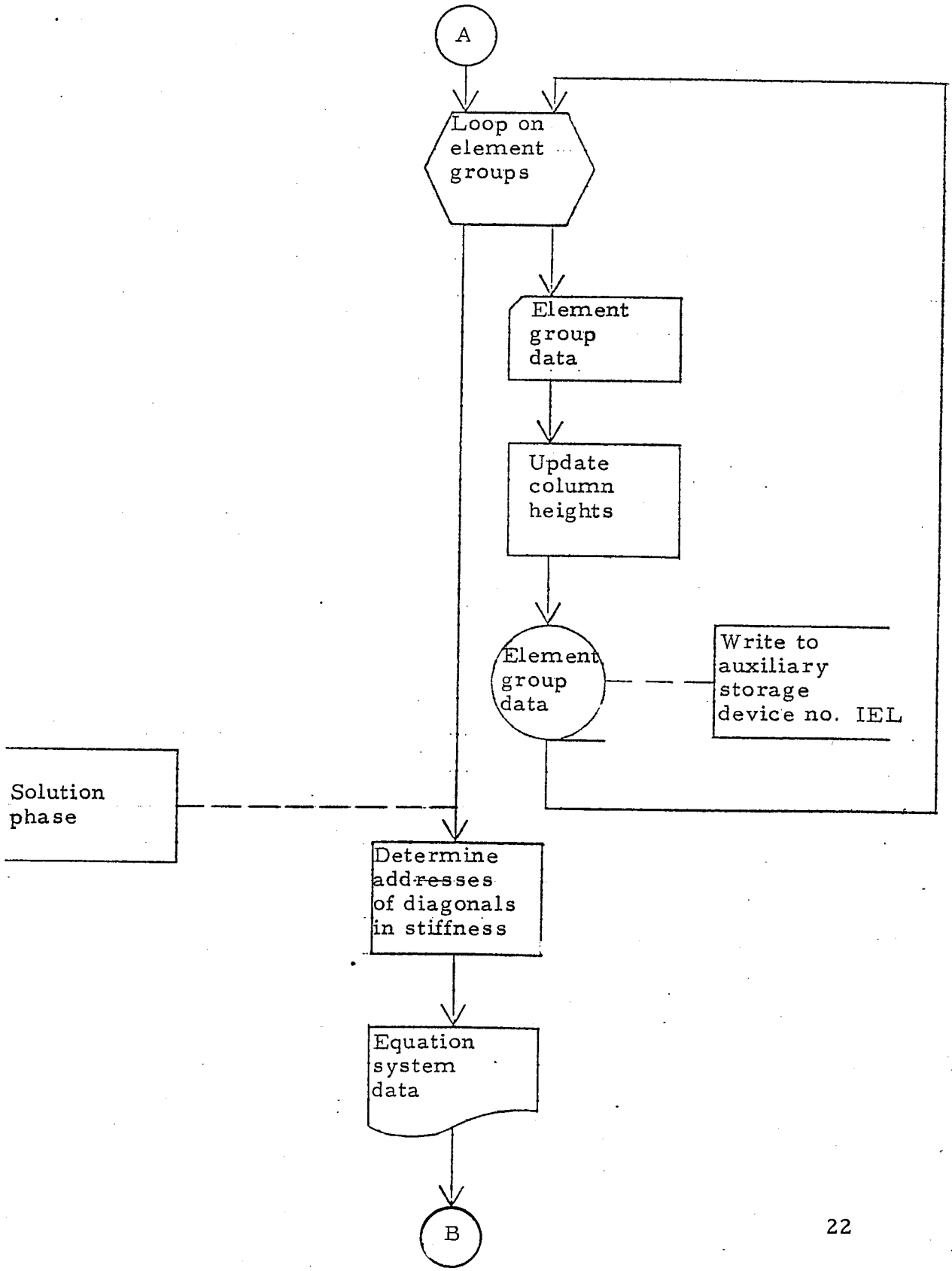
*Debugging is boring and frustrating. When you are feeling particularly depressed by this experience, remember everyone who has even programmed has experienced what you are feeling now. Misery loves company.

User's Manual

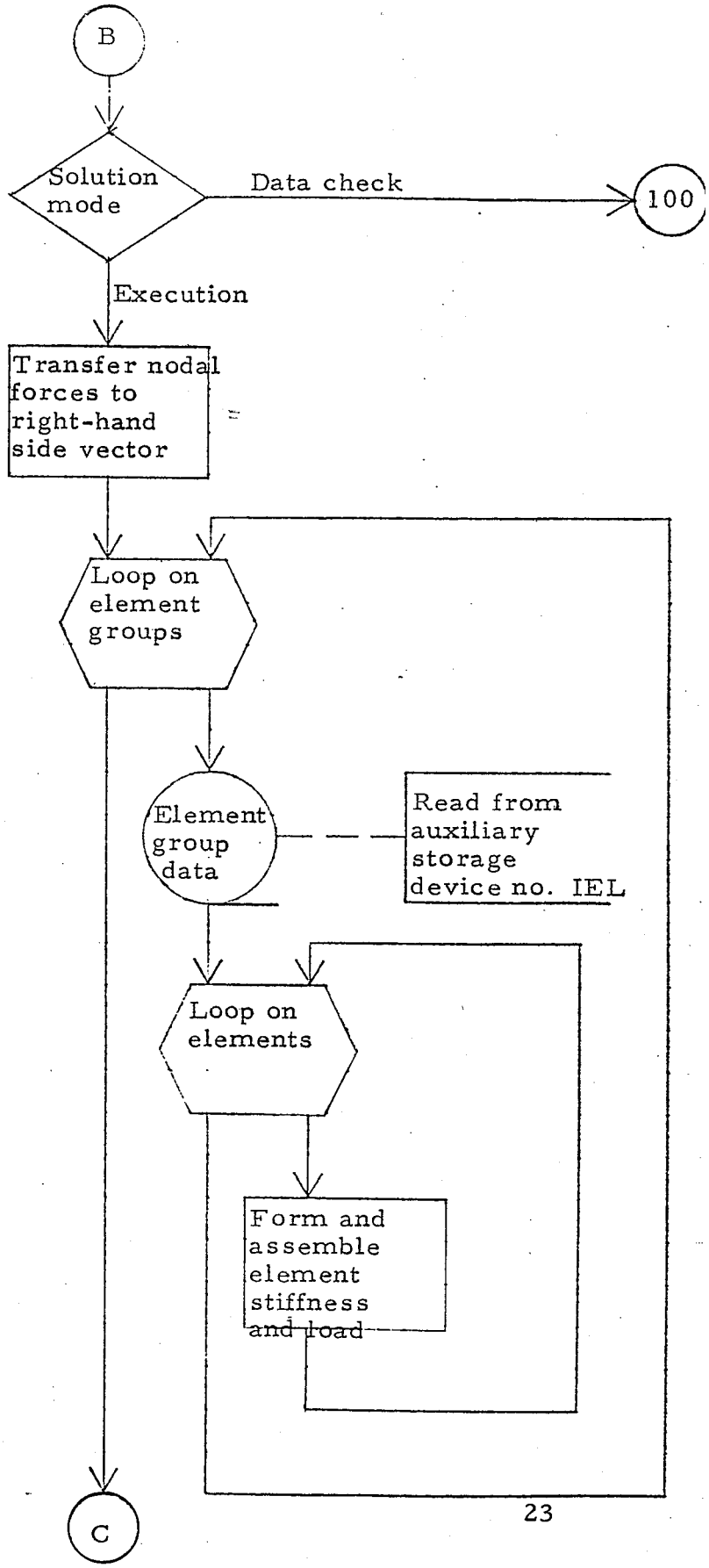
Flowchart


Start
Input phase
1
Control information
No. of nodes equals 0?
Yes
Stop
No
Nodal coordinate data
Nodal boundary condition codes
Establish eq uation numbers
Nodal force/ displacement data
A

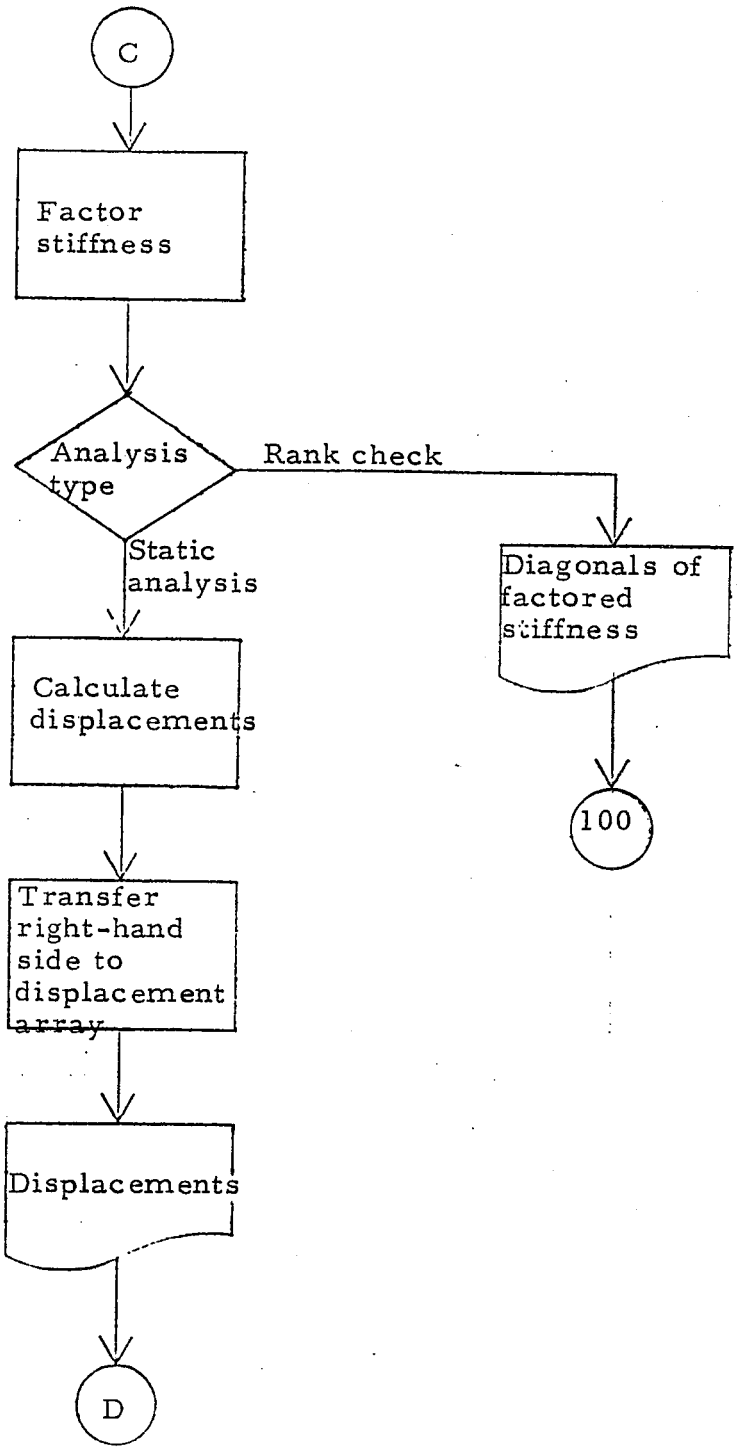
Flowchart (cont'd)



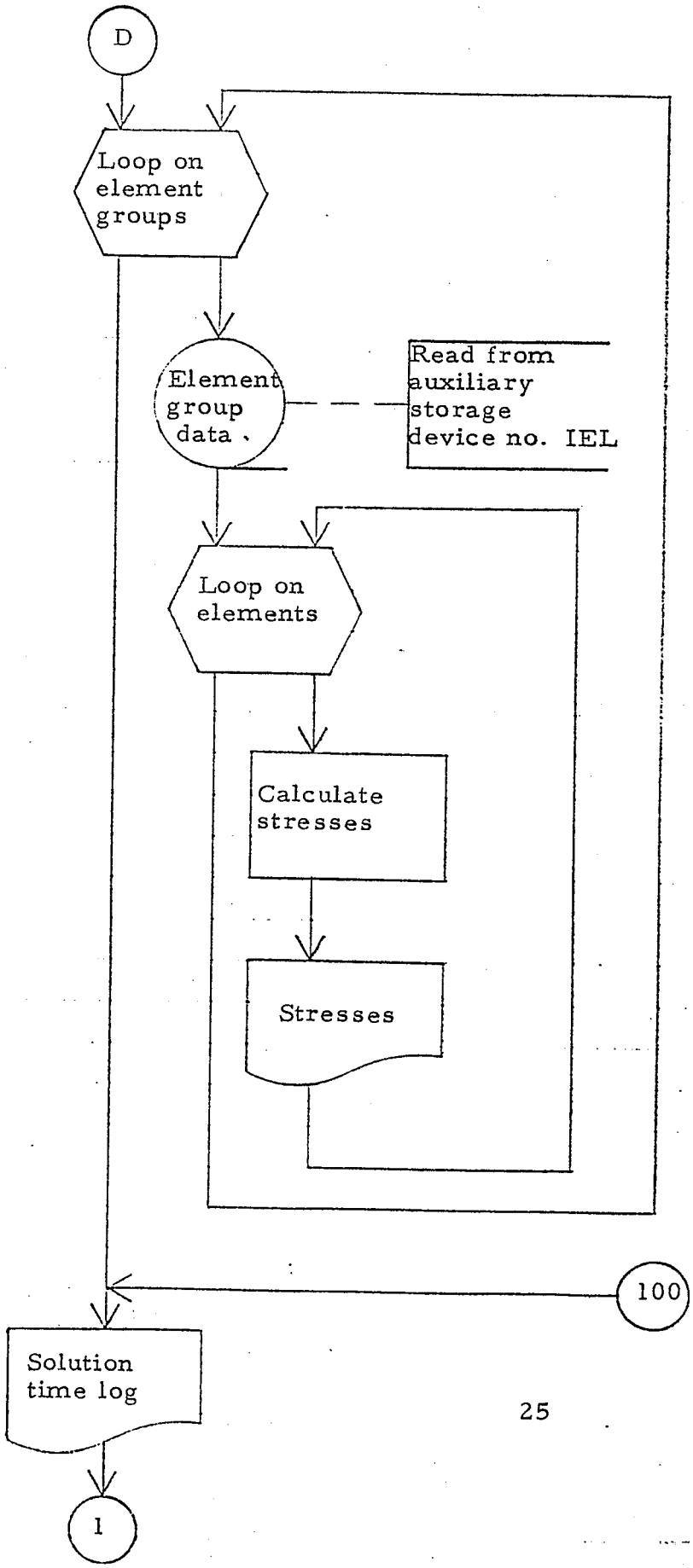
Flowchart (cont'd)



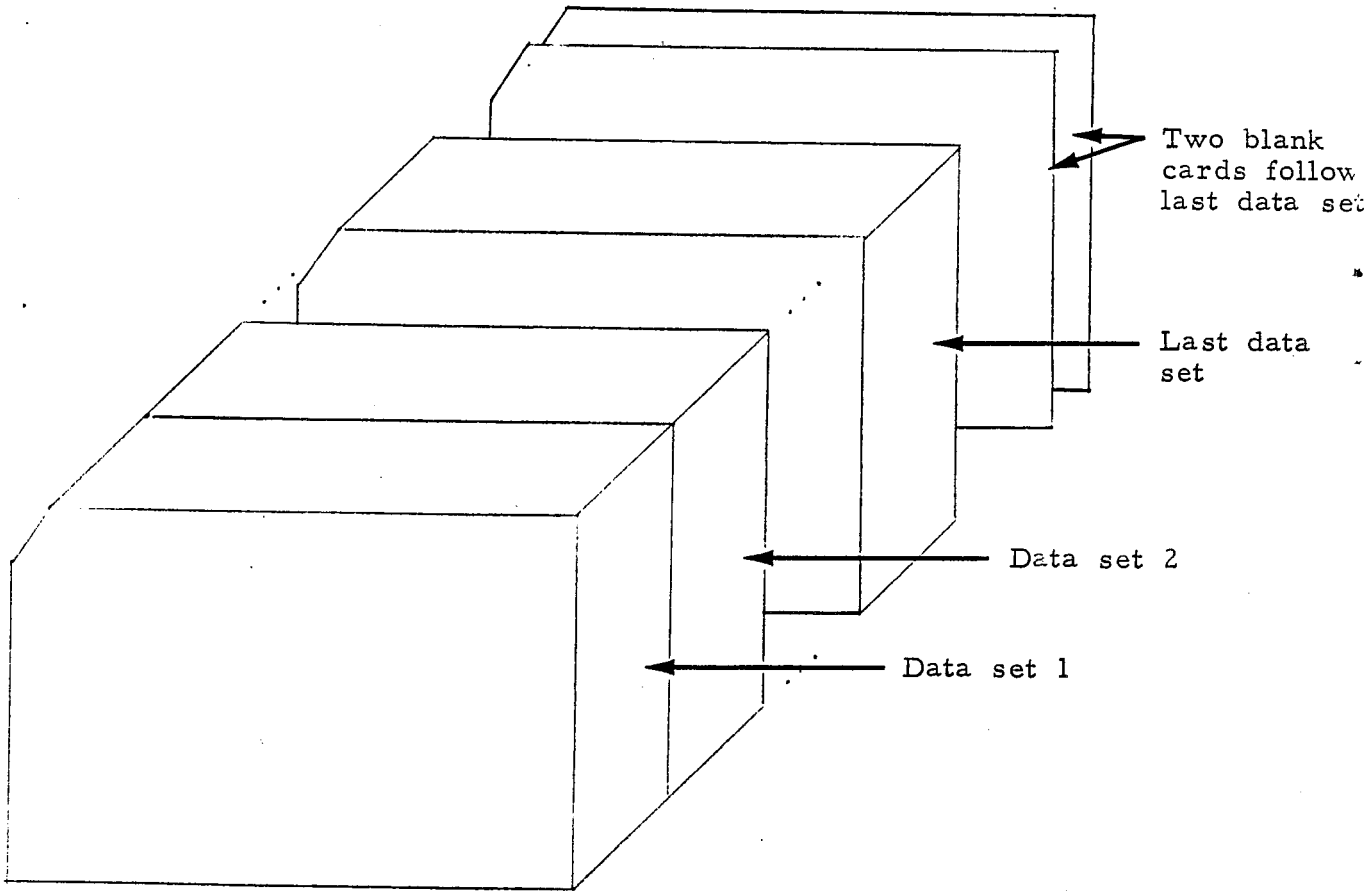
Flowchart (cont'd)



Flowchart (cont'd)



Data Set-Up


Two blank cards follow last data set
Last data set
Data set 2
Data set 1

Individual Data Cards

1.0 TITLE CARD (20A4)

Note	Columns	Variable	Description
(1)	1-80	TITLE(20)	Job title for heading the output

Notes/

(1) Each data set begins with a title card. Two blank cards follow the last data set.

2.0 CONTROL CARD (6I5)

Note	Columns	Variable	Description
	1-5	NUMNP	Number of nodes; EQ. 0, program stops
(1)	6-10	NSD	Number of spatial dimensions GE. 1 and LE. 3
(2)	11-15	NDOF	Number of degrees of freedom at each node; GE. 1 and LE. 6
(3)	16-20	NUMEG	Number of element groups
(4)	21-25	ICODE	Analysis code EQ. 0, static analysis EQ. 1, stiffness rank check
(5)	26-30	MODE	Execution mode EQ. 0, data check only EQ. 1, execution

Notes/

(1) Consult individual element routines in Section 6 for requirements.

(2) In problems for which there are different numbers of degrees of freedom at different nodes, NDOF should be set equal to the maximum number. Consult individual element routines in Section 6 for requirements. Superfluous degrees of freedom can then be eliminated by employing boundary condition codes (see Section 4.0).

(3) The elements are read in groups.

(4) A static analysis consists of solution of the equilibrium equations, displacement and stress output.

A rank check consists of factorization of the stiffness matrix, $K = U^T D U$, and printing the elements of the diagonal matrix D. The number of zero-energy modes can be deduced from the number of zeroes along the diagonal of D. This option allows for the checking of elements and meshes suspected of containing spurious zero-energy modes, such as those arising in so-called "underintegrated finite elements."

Notes Continued:

(5) In the data check mode input data is printed out and storage requirements are indicated. This mode should be employed prior to making expensive executions.

3.0 COORDINATE DATA (2I5, NSD*F10.0)

Note	Columns	Variable	Description
(1)	1-5	N	Node number; GE.1 and LE.NUMNP
(2)	6-10	NG	Generation increment
(3)	11-20	X(1,N)	x_1-coordinate of node N
	21-30	X(2,N)	x_2-coordinate of node N
	31-40	X(3,N)	x_3-coordinate of node N

Notes/

(1) The coordinates of each node must be defined, but need not be read in order. Terminate with a blank card.

(2) Coordinate data can be generated along straight lines by employing a two card sequence as follows:

Card 1: L, LG, X(1, L), . . . , X(NSD, L)

Card 2: N, NG, X(1, N), . . . , X(NSD, N)

N-L must be an integral multiple of LG. The coordinates of all nodes

L+LG, L+2*LG, . . . , N-LG

are generated by linearly interpolating the coordinates of nodes L and N. If LG is blank or zero no generation takes place between L and N.

(3) If more than one coordinate data card for node N is input, the last one read takes priority.

4.0 BOUNDARY CONDITION DATA ((2+NDOF)*I5)

Note	Columns	Variable	Description
(1)	1-5	N	Node number; GE. 1 and LE. NUMNP
(2)	6-10	NG	Generation increment
(3)	11-15	ID(1, N)	Degree of freedom 1 boundary code
	16-20	ID(2, N)	Degree of freedom 2 boundary code
	etc.	⋮	⋮
		ID(NDOF, N)	Degree of freedom NDOF boundary code

Notes/

(1) Boundary condition data must be input for each node which has one or more specified displacements. Cards need not be input in order. Terminate with a blank card.

(2) Boundary condition data can be generated by employing a two card sequence as follows:

Card 1: L, LG, ID(1, L), . . . , ID(NDOF, L)

Card 2: N, NG, ID(1, N), . . . , ID(NDOF, N)

The boundary codes of all nodes

L+LG, L+2*LG, . . . , N-MOD(N-L, LG)

(i.e., less than N) are set equal to those of node L. If LG is blank or zero no generation takes place between L and N.

(3) Boundary condition codes may be assigned the following values:

ID(I, N) = 0 , unspecified displacement

ID(I, N) = 1 , specified displacement

Notes Continued:

where I = 1, 2, . . . , NDOF. Specified displacements are assumed to be fixed (i.e., have the value 0.0) unless assigned a nonzero value as described in Section 5.0. If more than one boundary condition data card for node N is input, the last one read takes priority.

5.0 APPLIED NODAL FORCES AND PRESCRIBED DISPLACEMENTS (2I5, NDOF*F10.0)

Note	Columns	Variable	Description
(1)	1-5	N	Node number; GE.1 and LE.NUMNP
(2)	6-10	NG	Generation increment
(3)	11-20	F(1,N)	Degree of freedom 1 force or displacement
	21-30	F(2,N)	Degree of freedom 2 force or displacement
	etc.	⋮	⋮
		F(NDOF,N)	Degree of freedom NDOF force or displacement

Notes/

(1) Applied nodal force/prescribed displacement data must be included for each node subjected to a nonzero applied force or nonzero prescribed displacement. Cards need not be read in order. Terminate with a blank card.

(2) Applied nodal force and prescribed displacement data may be generated by employing a two card sequence as follows (generation is the same as for coordinate data):

Card 1: L, LG, F(1, L), . . . , F(NDOF, L)

Card 2: N, NG, F(1, N), . . . , F(NDOF, N)

N-L must be an integral multiple of LG. The applied nodal force and prescribed displacement data of all nodes

L+LG, L+2*LG, . . . , N-LG

are generated by linearly interpolating the data of nodes L and N. If LG is blank or zero no generation takes place between L and N.

(3) The elements of the array F(NDOF, NUMNP) are initialized to zero. If more than one applied nodal force/prescribed displacement data card for node N is input, the last one read takes priority.

6.0 ELEMENT DATA

6.1.0 Three-Dimensional Truss Element

Truss elements connect two points in space and transmit axial force only. There are three degrees of freedom at each of the two nodes, i.e. the x_1, x_2 and x_3 translations. When employing truss elements, NSD must be EQ.3 and NDOF must be GE.3 on the master control card (see Section 1.0). The following sequence of cards is used to describe truss elements:

6.1.1 Element Group Control Card (3I5)

Note	Columns	Variable	Description
	1-5	NPAR(1) (=NTYPE)	The number 1
	6-10	NPAR(2) (=NUMEL)	Number of elements in this group; GE.1
(1)	11-15	NPAR(3) (=NUMAT)	Number of geometric/ material sets in this group; GE.0; if EQ. 0, set internally to 1

Notes/

(1) Defines number of geometric/material properties cards to be input in Section 6.1.2.

6.1.2 Geometric/Material Properties Cards (I5,5X,3F10.0)

Note	Columns	Variable	Description
(1)	1-5	N	Material identification number
	11-20	E(N)	Young's modulus
	21-30	AREA(N)	Cross-section area
(2)	31-40	WT(N)	Weight per unit length

Notes/

(1) One card is required for each set of elements possessing the same cross-section and material properties.

(2) The product of weight per unit length and length defines gravity loads for the truss element.

6.1.3 Gravity Load Multiplier Card (3F10.0)

Note	Columns	Variable	Description
(1)	1-10	GRAV(1)	Multiplier of gravity load in the x_1 direction
	11-20	GRAV(2)	Multiplier of gravity load in the x_2 direction
	21-30	GRAV(3)	Multiplier of gravity load in the x_3 direction

Notes/

(1) Gravity load multipliers may be used to define the components of the gravity unit vector with respect to the global x_1, x_2, x_3 system. For example, if the gravity load multipliers are

0.0, 0.0, -1.0,

then the entire gravitational load acts in the $-x_3$ direction. (One card for element group.)

6.1.4 Element Data Cards (5I5)

Note	Columns	Variable	Description
(1)	1-5	N	Element number
	6-10	MAT(N)	Geometric/material properties set number
	11-15	IEN(1,N)	Number of 1st node
	16-20	IEN(2,N)	Number of 2nd node
(2)	21-25	NG	Generation increment

Notes/

(1) All elements must be input on an element data card or generated. Terminate with a blank card.

(2) Element data cards may be generated by employing a two card sequence as follows:

Card 1: L, MAT(L), IEN(1,L), IEN(2,L), LG

Card 2: N, MAT(N), IEN(1,N), IEN(2,L), NG

N must be greater than L. The geometric/material set number for all elements

$$L+1, L+2, \ldots, N-1$$

is set to MAT(L), and the node numbers are incremented by LG.

6.2.0 Two-Dimensional Isotropic Elasticity Element

The present element may be used in triangle (3 node) or quadrilateral (4 node) form for plane strain and plane stress. In each case the plane of analysis is assumed to be the x_1, x_2 plane. Two displacement degrees of freedom, in the x_1 and x_2 directions, are assigned to each node. Incompatible modes and selective numerical integration may be employed to improve element behavior in various situations. These options should be activated only by users fully knowledgeable in their use. The nodes of the element must be input in counterclockwise order and NSD must EQ.2 and NDOF must be GE.2 on the master control card (see Section 1.0).

Stresses/strains in the global coordinate system, and principal stresses/strains, maximum shear stress/strain and angle of inclination, in degrees, of principal states (see Figure 6.2.0.1) are output at the element centroid, which is generally the point of optimal accuracy. All shear strains are reported according to the "engineering" convention (i.e. twice the value of the tensor components).

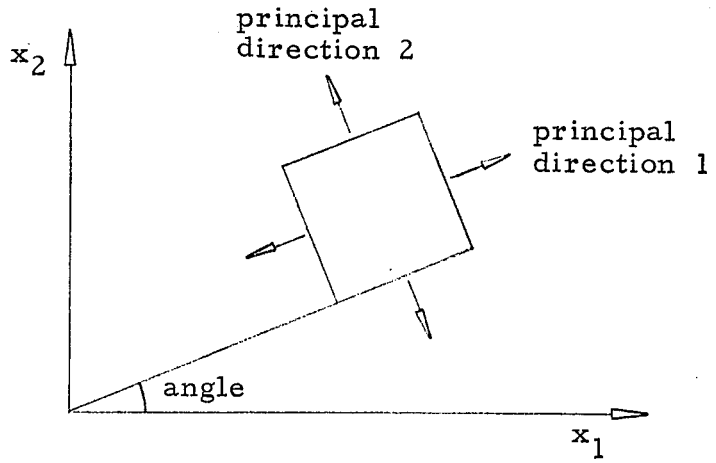


Figure 6.2.0.1

The following sequence of cards is used to describe the elements:

6.2.1 Element Group Control Card (9I5)

Note	Columns	Variable	Description
	1-5	NPAR(1) (=NTYPE)	The number 3
	6-10	NPAR(2) (=NUMEL)	Number of elements in this group; GE. 1
	11-15	NPAR(3) (=NUMAT)	Number of geometric/ material sets in this group; GE. 0; if EQ. 0, set internally to 1
	16-20	NPAR(4) (=IOPT)	Analysis option EQ. 0, plane strain EQ. 1, plane stress
(1)	21-25	NPAR(5) (=IL)	Lambda-term (λ) numerical integration code EQ. 0, two-by-two Gaussian quadrature EQ. 1, one-point Gaussian quadrature
(2)	26-30	NPAR(6) (=IM)	Mhu-term (μ) numerical integration code EQ. 0, two-by-two Gaussian quadrature EQ. 1, one-point Gaussian quadrature
(3)	31-35	NPAR(7) (=INC)	Incompatible modes code EQ. 0, incompatible modes neglected EQ. 1, incompatible modes added
(4)	36-40	NPAR(8) (=IHOUR)	Hourglass stiffness code EQ. 0, hourglass stiffness neglected EQ. 1, hourglass stiffness added
	41-45	NPAR(9) (=NUMPR)	Number of element pressure load cards

6.2.1 Continued

Notes/

(1) In problems involving nearly incompressible materials an effective approach is to use one-point Gaussian quadrature on the λ term, and two-by-two Gaussian quadrature on the μ term (see T.J.R. Hughes, "Equivalence of Finite Elements for Nearly-Incompressible Elasticity," LBL 5237, University of California, Berkeley, August 1976, or Journal of Applied Mechanics, Vol. 44, Series E, No. 1, p. 181, March 1977.

(2) The "standard" 4-node quadrilateral employs two-by-two Gaussian quadrature on both the λ and μ terms. However, it is ineffective in application to nearly-incompressible materials (see above note) and also in application to "bending" situations. One-point Gaussian quadrature on both terms produces a more accurate, but dangerous element. The danger is that zero energy modes of deformation, so-called hourglass or keystone modes, may be present, resulting in a singular stiffness matrix. If enough displacement boundary conditions are present these modes may be eliminated. However, one-point Gaussian quadrature on both terms should only be used if you know exactly what you are doing.

(3) The presence of incompatible modes produces an element effective in bending and in application to incompressible materials. Two-by-two Gaussian quadrature on both the λ and μ terms should be employed when using incompatible modes. Much pro and con has been written about incompatible modes. The present implementation is based upon R.L. Taylor, P.J. Beresford and E.L. Wilson, "A Non-Conforming Element for Stress Analysis," International Journal for Numerical Methods in Engineering, 10 (6) p. 1211, 1976.

(4) The hourglass stiffness counteracts the zero-energy hourglass modes when one-point Gaussian quadrature is used on both the λ and μ terms. The implementation is based upon D. Kosloff and G. A. Frazier, "Treatment of Hourglass Patterns in Low Order Finite Element Codes," Contribution No. 2895, Division of Geological and Planetary Sciences, California Institute of Technology, Pasadena, California, or Numerical and Analytical Methods in Geomechanics, Vol. 2, 57-72 (1978). The hourglass stiffness should be added only when one-point Gaussian quadrature is employed on both the λ and μ terms.

6.2.2 Geometric/Material Properties Cards (I5,5X,5F10.0)

Note	Columns	Variable	Description
	1-5	N	Material identification number
	11-20	E(N)	Young's modulus
(1)	21-30	POIS(N)	Poisson's ratio
(2)	31-40	WT(N)	Weight density
(3)	41-50	TH(N)	Thickness (if 0.0 defaults to 1.0)

Notes/

(1) Poisson's ratio cannot be set equal to 1/2 since it results in division by zero. A value close to 1/2, say .4999, can be employed for incompressible applications.

(2) The product of the weight density and element volume is used to compute gravity loads.

(3) In plane strain the thickness is automatically assumed to be 1.0.

6.2.3 Gravity Load Multiplier Cards (2F10.0)

Note	Columns	Variables	Description
(1)	1-10	GRAV(1)	Multiplier of gravity load in the x_1 direction
	11-20	GRAV(2)	Multiplier of gravity load in the x_2 direction

Notes/

(1) Gravity load multipliers are used to define the components of the gravity vector with respect to the global x_1, x_2 system.

6.2.4 Element Data Cards (7I5)

Note	Columns	Variables	Description
(1)	1-5	N	Element number
	6-10	MAT(N)	Geometric/material properties set number
	11-15	IEN(1,N)	Number of 1st node
	16-20	IEN(2,N)	Number of 2nd node
	21-25	IEN(3,N)	Number of 3rd node
(2)	26-30	IEN(4,N)	Number of 4th node
(3)	31-35	NG	Generation increment

Notes/

(1) All elements must be input on an element data card or generated. Terminate with a blank card.

(2) Node number IEN(4,N) must equal node number IEN(3,N) for triangular elements. Triangles should not be used for incompressible or nearly-incompressible materials.

6.2.4 Continued

Notes Continued:

(3) Element data cards may be generated by employing a two card sequence as follows:

Card 1: L, MAT(L), IEN(1, L), IEN(2, L), IEN(3, L), IEN(4, L), LG

Card 2: N, MAT(N), IEN(1, N), IEN(2, N), IEN(3, N), IEN(4, N), NG

N must be greater than L. The geometric/material set number for all

$$L+1, L+2, \ldots, N-1$$

is set to MAT(L) and the node numbers are incremented by LG.

6.2.5 Element Pressure Load Cards (2I5, 10X, F10.0)

Note	Columns	Variables	Description
(1)	1-5	IELNO(I)	Element number 1.LE.IELNO(I).LE.NUMEL
(2)	6-10	ISIDE(I)	Element side number 1.LE.ISIDE(I).LE.4
(3)	21-30	PRES(I)	Pressure (force/unit area)

Notes/

(1) Each element subjected to a pressure load must be input on an element pressure load card. If more than one side of the element is loaded, one card for each loaded side must be input. The total number of element pressure load cards must equal NUMPR read in on the element group control card (see Section 6.2.1). The index I in the arrays IELNO, ISIDE and PRES corresponds to the order read in; 1.LE.I.LE.NUMPR. The cards need not be read in any particular order; terminate with a blank card.

(2) The element side number is deduced as follows: Consider the element of Figure 6.2.1.1; I, J, K, L indicate the element node numbers as read in on the element data card (i.e., I is the

6.2.5 Continued

Notes Continued:

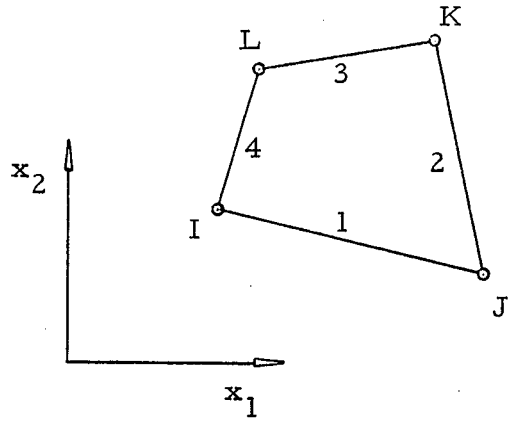


Figure 6.2.1.1

first node, J the second, etc.). Side 1 connects nodes I and J, side 2 connects nodes J and K, etc.; see Figure 6.2.1.1.

(3) The pressure is assumed to be positive pointing inward, as indicated in Figure 6.2.1.2

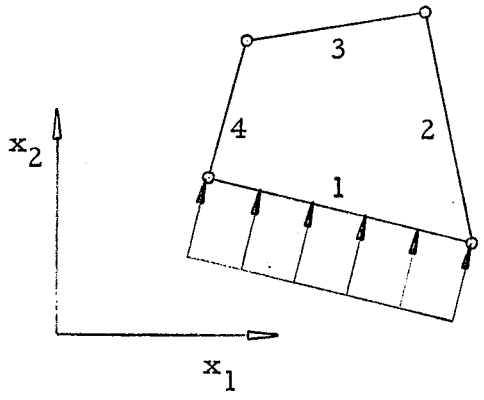
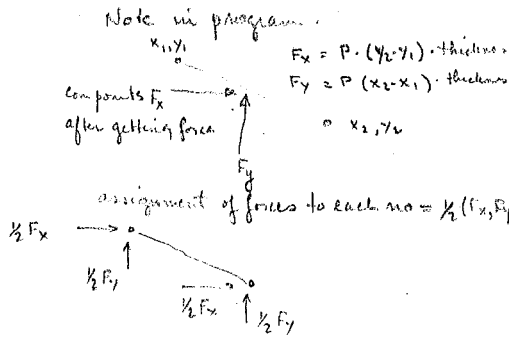


Figure 6.2.1.2



Listing

```
C
C.... MAIN PROGRAM                                                              1 MAIN
C                                                                               2 MAIN
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG          3 MAIN
      COMMON/DEVICE/ IEL,IIN,IOUT                                               4 MAIN
      COMMON A(5000)                                                            5 MAIN
      MTOT=5001                                                                 6 MAIN
      IPREC=2                                                                   7 MAIN
      IEL=1                                                                     8 MAIN
      IIN=5 ~ input file                                                        9 MAIN
      IOUT=6 ~ output file                                                     10 MAIN
      CALL LEARN                                                               11 MAIN
      END                                                                      12 MAIN
                                                                               13 MAIN


C
      SUBROUTINE LEARN                                                          1 LEARN
C                                                                               2 LEARN
C     ****************************************************************          3 LEARN
C     *                                                              *          4 LEARN
C     *                          L E A R N                           *          5 LEARN
C     *                                                              *          6 LEARN
C     *  A L I N E A R  S T A T I C  A N A L Y S I S  P R O G R A M  *          7 LEARN
C     *                                                              *          8 LEARN
C     *          T . H U G H E S  O C T O B E R  1 9 7 6             *          9 LEARN
C     *                                                              *         10 LEARN
C     *       ( D O U B L E  P R E C I S I O N  V E R S I O N )      *         11 LEARN
C     *                                                              *         12 LEARN
C     ****************************************************************         13 LEARN
C                                                                              14 LEARN
      COMMON/INFO/ICODE,MODE,NDOF,NSD,NUMNP                                    15 LEARN
      COMMON/DEVICE/ IEL,IIN,IOUT                                              16 LEARN
      COMMON/POINTS/N1,N2,N3,N4,N5,N6,N7                                       17 LEARN
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG         18 LEARN
      COMMON/LABELS/LABELC(3),LABELD(6)                                        19 LEARN
      DIMENSION TIME(5),TITLE(20)                                              20 LEARN
      COMMON A(1)                                                              21 LEARN
C                                                                              22 LEARN
C     DEVICE CODES                                                             23 LEARN
C        IIN= INPUT                                                            24 LEARN
C        IOUT=OUTPUT                                                           25 LEARN
C        IEL= ELEMENT DATA                                                     26 LEARN
C                                                                              27 LEARN
  1   MAX=0                                                                    28 LEARN
C                                                                              29 LEARN
C     ****************************************************************         30 LEARN
C     *                                                              *         31 LEARN
C     *                      I N P U T  P H A S E                    *         32 LEARN
C     *                                                              *         33 LEARN
C     ****************************************************************         34 LEARN
C                                                                              35 LEARN
C.... PROGRAM CONTROL INFORMATION                                              36 LEARN
C                                                                              37 LEARN
      CALL SECOND(TIME(1))                                                     38 LEARN
      READ(IIN,1000)TITLE,NUMNP,NSD,NDOF,NUMEG,ICODE,MODE                      39 LEARN
      IF(NUMNP.EQ.0) STOP                                                      40 LEARN
      WRITE(IOUT,2000) TITLE,NUMNP,NSD,NDOF,NUMEG,ICODE,MODE                   41 LEARN
C                                                                              42 LEARN
C.... INPUT NODAL DATA                                                         43 LEARN
C                                                                              44 LEARN
      N1=1                  (ID) = A(N1)                                       45 LEARN
      N2=N1+NUMNP*NDOF      X = A(N2)                                          46 LEARN
      N3=N2+NUMNP*NSD*IPREC F,D = A(N3)                                        47 LEARN
      IF(N3.GT.MTOT) CALL ERROR (N3-MTOT,1)  check size of array for nodal coord & bdy data  48 LEARN
      CALL COORD(A(N2),NSD,NUMNP)   generates nodal coords                     49 LEARN
      CALL BC(A(N1),NDOF,NUMNP,NEQ)   generates ID                             50 LEARN
C                                                                              51 LEARN
C.... INPUT NODAL APPLIED FORCE AND PRESCRIBED DISPLACEMENT DATA               52 LEARN
C                                                                              53 LEARN
      N4=N3+NUMNP*NDOF*IPREC   IDIAG=A(N4)                                     54 LEARN
      IF(N4.GT.MTOT) CALL ERROR(N4-MTOT,2)  check applied forces & displ data  55 LEARN
                                                                               56 LEARN
```

```
      CALL FORCE (A(N3),NDOF,NUMNP)   generate force
C
C.... INPUT ELEMENT DATA
C
      N5=N4+NEQ          A(N4)
      N6=N5
      N7=N5
      IF(N5.GT.MTOT)  CALL ERROR(N5-MTOT,3)
      CALL ICLEAR (A(N4),NEQ)
      IE=1
      CALL EL
      CALL SECOND(TIME(2))
C
C     ********************************************************************
C     *                                                                  *
C     *                    S O L U T I O N   P H A S E                   *
C     *                                                                  *
C     ********************************************************************
C
C.... DETERMINE ADDRESSES OF DIAGONALS IN STIFFNESS
C
      CALL DIAG(A(N4),NEQ,NA)
      N6=N5+NEQ*IPREC          B = A(N5)
      N7=N6+NA *IPREC          A = A(N6)
      N8=N7+MAX-1              E = A(N7)
      IF(N8.GT.MTOT) CALL ERROR(N8-MTOT,5)
C
C.... WRITE EQUATION SYSTEM  DATA
C
      MB=NA/NEQ
      WRITE(IOUT,2010) TITLE,NEQ,NA,MB,N8
C
C.... IF DATA CHECK MODE (MODE.EQ.0) SUBSEQUENT CALCULATIONS ARE IGNORED
C
      IF(MODE.EQ.1) GOTO 10
      CALL SECOND (TIME(3))
      CALL SECOND (TIME(4))
      CALL SECOND (TIME(5))
      GOTO 100
C
C.... ASSEMBLE STIFFNESS AND LOAD
C
   10 CALL CLEAR (A(N5),NEQ+NA)
      CALL LOAD(A(N1),A(N3),A(N5),NDOF,NUMNP)
      IE=2
      CALL ELREAD(A(N7))
      CALL SECOND(TIME(3))
C
C.... FACTOR STIFFNESS MATRIX
C                      A     B    IDIAG
      CALL SOLVE(A(N6),A(N5),A(N4),NEQ,.TRUE.,.FALSE.)
      CALL SECOND (TIME(4))
C
C.... IF RANK CHECK ANALYSIS (ICODE.EQ.1) SUBSEQUENT CALCULATIONS ARE
C        IGNORED
C
      IF(ICODE.EQ.0) GOTO 20
      CALL PRINTP(A(N6),A(N4),NEQ)
      CALL SECOND(TIME(5))
      GOTO 100
C
C.... CALCULATION  OF  DISPLACEMENTS
C                      A     B    IDIAG
   20 CALL SOLVE(A(N6),A(N5),A(N4),NEQ,.FALSE.,.TRUE.)
      CALL DISPTR(A(N5),A(N3),A(N1),NDOF,NUMNP)
      CALL PRINTD(A(N3),NDOF,NUMNP)
C
C.... CALCULATION  OF  STRESSES
C
      IE=3
      CALL ELREAD(A(N7))
      CALL SECOND(TIME(5))
```

```
 57 LEARN
 58 LEARN
 59 LEARN
 60 LEARN
 61 LEARN
 62 LEARN
 63 LEARN
 64 LEARN
 65 LEARN
 66 LEARN
 67 LEARN
 68 LEARN
 69 LEARN
 70 LEARN
 71 LEARN
 72 LEARN
 73 LEARN
 74 LEARN
 75 LEARN
 76 LEARN
 77 LEARN
 78 LEARN
 79 LEARN
 80 LEARN
 81 LEARN
 82 LEARN
 83 LEARN
 84 LEARN
 85 LEARN
 86 LEARN
 87 LEARN
 88 LEARN
 89 LEARN
 90 LEARN
 91 LEARN
 92 LEARN
 93 LEARN
 94 LEARN
 95 LEARN
 96 LEARN
 97 LEARN
 98 LEARN
 99 LEARN
100 LEARN
101 LEARN
102 LEARN
103 LEARN
104 LEARN
105 LEARN
106 LEARN
107 LEARN
108 LEARN
109 LEARN
110 LEARN
111 LEARN
112 LEARN
113 LEARN
114 LEARN
115 LEARN
116 LEARN
117 LEARN
118 LEARN
119 LEARN
120 LEARN
121 LEARN
122 LEARN
123 LEARN
124 LEARN
125 LEARN
126 LEARN
127 LEARN
128 LEARN
```

```
C
C.... PRINT SOLUTION TIMES                                                      129 LEARN
C                                                                               130 LEARN
  100 TOT=TIME(1)-TIME(5)                                                       131 LEARN
      DO 30 I=1,4                                                               132 LEARN
   30 TIME(I)=TIME(I)-TIME(I+1)                                                 133 LEARN
      WRITE(IOUT,2020)  TITLE,(TIME(I),I=1,4),TOT                               134 LEARN
C                                                                               135 LEARN
C.... READ NEXT ANALYSIS CASE                                                   136 LEARN
C                                                                               137 LEARN
      GOTO 1                                                                    138 LEARN
C                                                                               139 LEARN
 1000 FORMAT(20A4/6I5)                                                          140 LEARN
 2000 FORMAT(1H1,20A4///                                                        141 LEARN
     X' C O N T R O L   I N F O R M A T I O N                    '   //5X,       142 LEARN
     X' NUMBER OF NODAL POINTS . . . . . (NUMNP) =',I5//5X,                     143 LEARN
     X' NUMBER OF SPACE DIMENSIONS . . . . . (NSD  ) =',I5//5X,                 144 LEARN
     X' NUMBER OF DEGREES OF FREEDOM PER NODE . . . (NDOF ) =',I5//5X,          145 LEARN
     X' NUMBER OF ELEMENT GROUPS . . . . . (NUMEG) =',I5//5X,                   146 LEARN
     X' ANALYSIS CODE . . . . . (ICODE) =',I5/ 5X,                              147 LEARN
     X'    EQ.0, STATIC ANALYSIS                                       ',  / 5X,  148 LEARN
     X'    EQ.1, RANK CHECK                                            ',  //5X,  149 LEARN
     X' SOLUTION MODE . . . . . (MODE ) =',I5/ 5X,                              150 LEARN
     X'    EQ.0,.DATA CHECK                                            ',  / 5X,  151 LEARN
     X'    EQ.1, EXECUTION                                             '       )  152 LEARN
 2010 FORMAT(1H1,20A4///                                                        153 LEARN
     X' E Q U A T I O N   S Y S T E M   D A T A                  ',  //5X,       154 LEARN
     X' NUMBER OF EQUATIONS . . . . . (NEQ) =',I5//5X,                          155 LEARN
     X' NUMBER OF TERMS IN STIFFNESS . . . . . (NA ) =',I5//5X,                 156 LEARN
     X' MEAN HALF BANDWIDTH . . . . . (MB ) =',I5//5X,                          157 LEARN
     X' TOTAL LENGTH OF BLANK COMMON REQUIRED . . . . (N8 ) =',I5    )          158 LEARN
 2020 FORMAT(1H1,20A4///                                                        159 LEARN
     X' S O L U T I O N   T I M E   L O G   I N   S E C O N D S' //5X,          160 LEARN
     X' INPUT PHASE . . . . . =',F12.2//5X,                                     161 LEARN
     X' FORMATION OF STIFFNESS AND LOAD . . . . . =',F12.2//5X,                 162 LEARN
     X' FACTOR STIFFNESS . . . . . =',F12.2//5X,                                163 LEARN
     X' DISPLACEMENT AND STRESS ANALYSIS . . . . . =',F12.2//5X,                164 LEARN
     X' T O T A L   S O L U T I O N   T I M E . . . . . =',F12.2    )           165 LEARN
C                                                                               166 LEARN
      RETURN                                                                    167 LEARN
      END                                                                       168 LEARN
                                                                                169 LEARN


C                                                                                 1 BLOCK
      BLOCK DATA                                                                  2 BLOCK
C                                                                                 3 BLOCK
C.... PROGRAM  TO  DEFINE NODAL COORDINATES AND DEGREE OF FREEDOM OUTPUT          4 BLOCK
C        LABELS                                                                   5 BLOCK
C                                                                                 6 BLOCK
C     LABELC(3) = COORDINATES LABELS                                              7 BLOCK
C     LABELD(6) = DEGREE OF FREEDOM LABELS                                        8 BLOCK
C                                                                                 9 BLOCK
      COMMON/LABELS/LABELC(3),LABELD(6)                                          10 BLOCK
      DATA LABELC/' X1 ',' X2 ',' X3 '/                                          11 BLOCK
     X     LABELD/4HDOF1,4HDOF2,4HDOF3,4HDOF4,4HDOF5,4HDOF6/                     12 BLOCK
      END                                                                        13 BLOCK


C                                                                                 1 ADDSTF
      SUBROUTINE ADDSTF(A,B,S,P,IDIAG,LM,NEE)                                     2 ADDSTF
C                                                                                 3 ADDSTF
C.... PROGRAM TO ADD ELEMENT STIFFNESS AND FORCE TO GLOBAL ARRAYS                 4 ADDSTF
C                                                                                 5 ADDSTF
      IMPLICIT REAL*8(A-H,O-Z)                                                    6 ADDSTF
C                                                                                 7 ADDSTF
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                            8 ADDSTF
C                                                                                 9 ADDSTF
      DIMENSION A(1),B(1),S(NEE,1),P(1),IDIAG(1),LM(1)                           10 ADDSTF
      DO 10 J=1,NEE                                                              11 ADDSTF
      K=LM(J)                                                                    12 ADDSTF
```

```
      IF(K.EQ.0) GOTO 10                                                ADDSTF 13
      B(K)=B(K)+P(J)                                                    ADDSTF 14
      L=IDIAG(K)-K                                                      ADDSTF 15
      DO 20 I=1,NEE                                                     ADDSTF 16
      M=LM(I)                                                           ADDSTF 17
      IF(M.GT.K.OR.M.EQ.0) GOTO 20                                      ADDSTF 18
      M=L+M                                                             ADDSTF 19
      A(M)=A(M)+S(I,J)                                                  ADDSTF 20
   20 CONTINUE                                                          ADDSTF 21
   10 CONTINUE                                                          ADDSTF 22
C                                                                       ADDSTF 23
      RETURN                                                            ADDSTF 24
      END                                                               ADDSTF 25


C                                                                         BC  1
      SUBROUTINE BC(ID,NDOF,NUMNP,NEQ)                                    BC  2
C                                                                         BC  3
C.... PROGRAM FOR READING AND PRINTING BOUNDARY CONDITION DATA            BC  4
C        AND ESTABLISHING EQUATION NUMBERS                                BC  5
C                                                                         BC  6
C        ID(NDOF,NUMNP) = BOUNDARY CONDITION CODES                        BC  7
C                           EQ.0, FREE                                    BC  8
C                           EQ.1, DISPLACEMENT                            BC  9
C                                                                         BC 10
      COMMON/LABELS/LABELC(3),LABELD(6)                                   BC 11
      COMMON/DEVICE/IEL,IIN,IOUT                                          BC 12
      DIMENSION ID(NDOF,1)                                                BC 13
C                                                                         BC 14
      CALL ICLEAR(ID,NDOF*NUMNP)                                          BC 15
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                               BC 16
      CALL IGEN(ID,NDOF)                                                  BC 17
      NN=0                                                                BC 18
      DO 10 N=1,NUMNP                                                     BC 19
      IF(NN.LE.50) GOTO 20                                                BC 20
      NN=1                                                                BC 21
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                               BC 22
   20 DO 30 I=1,NDOF                                                      BC 23
      IF(ID(I,N)) 40,30,40                                                BC 24
   30 CONTINUE                                                            BC 25
      GOTO 10                                                             BC 26
   40 NN=NN+1                                                             BC 27
      WRITE(IOUT,2010) N,(ID(I,N),I=1,NDOF)                               BC 28
   10 CONTINUE                                                            BC 29
C                                                                         BC 30
C.... ESTABLISH EQUATION NUMBERS                                          BC 31
C                                                                         BC 32
      NEQ=0                                                               BC 33
      DO 100 N=1,NUMNP                                                    BC 34
      DO 100 I=1,NDOF                                                     BC 35
      IF(ID(I,N).EQ.0) GOTO 110                                           BC 36
      ID(I,N)=0                                                           BC 37
      GOTO 100                                                            BC 38
  110 NEQ=NEQ+1                                                           BC 39
      ID(I,N)=NEQ                                                         BC 40
  100 CONTINUE                                                            BC 41
      RETURN                                                              BC 42
C                                                                         BC 43
 2000 FORMAT(1H1,' N O D A L   B O U N D A R Y   C O N D I T I O N   C O  BC 44
     X D E S'///                                                          BC 45
     X      4X,' NODE NUMBER ',4X,6(6X,A4)/)                              BC 46
 2010 FORMAT(6X,I5,10X,6(5X,I5))                                          BC 47
      END                                                                 BC 48


C                                                                       COLHT  1
      SUBROUTINE COLHT(IDIAG,LM,NED,NEN,NUMEL)                          COLHT  2
C                                                                       COLHT  3
C.... PROGRAM TO COMPUTE COLUMN HEIGHTS                                 COLHT  4
C                                                                       COLHT  5
      DIMENSION IDIAG(1),LM(NED,NEN,1)                                  COLHT  6
```

```
C
      DO 10 K=1,NUMEL                                                    7 COLHT
      MIN=100000                                                         8 COLHT
      DO 20 I=1,NED                                                      9 COLHT
      DO 20 J=1,NEN                                                     10 COLHT
      IF(LM(I,J,K).EQ.0) GOTO 20   for each element find min global eqn no.  11 COLHT
      MIN=MINO(MIN,LM(I,J,K))                                           12 COLHT
   20 CONTINUE                                                          13 COLHT
      DO 30 I=1,NED                                                     14 COLHT
      DO 30 J=1,NEN                                                     15 COLHT
      II=LM(I,J,K)                                                      16 COLHT
      IF(II.EQ.0) GOTO 30   this then finds the length of the column    17 COLHT
      M=II-MIN                                                          18 COLHT
      IF(M.GT.IDIAG(II)) IDIAG(II)=M  - this assigns to the array IDIAG(J) the length  19 COLHT
   30 CONTINUE                                                          20 COLHT
   10 CONTINUE                                                          21 COLHT
C                                                                       22 COLHT
      RETURN                                                            23 COLHT
      END                                                               24 COLHT
                                                                        25 COLHT


C
      SUBROUTINE CLEAR(A,M)   ✓                                          1 CLEAR
C                                                                        2 CLEAR
C.... PROGRAM TO CLEAR A FLOATING POINT ARRAY                            3 CLEAR
C                                                                        4 CLEAR
      IMPLICIT REAL*8(A-H,O-Z)                                           5 CLEAR
C                                                                        6 CLEAR
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                   7 CLEAR
C                                                                        8 CLEAR
      DIMENSION A(1)                                                     9 CLEAR
      DO 10 I=1,M                                                       10 CLEAR
   10 A(I)=0.D0                                                         11 CLEAR
C                                                                       12 CLEAR
      RETURN                                                            13 CLEAR
      END                                                               14 CLEAR
                                                                        15 CLEAR


C
      SUBROUTINE COORD(X,NSD,NUMNP)   ✓  OK as is, don't change!         1 COORD
C                                                                        2 COORD
C.... PROGRAM FOR READING AND PRINTING COORDINATE DATA                   3 COORD
C                                                                        4 COORD
C        X(NSD,NUMNP) = COORDINATE ARRAY                                 5 COORD
C                                                                        6 COORD
      IMPLICIT REAL*8(A-H,O-Z)                                           7 COORD
C                                                                        8 COORD
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATON                    9 COORD
C                                                                       10 COORD
      COMMON/LABELS/LABELC(3),LABELD(6)                                 11 COORD
      COMMON/DEVICE/IEL,IIN,IOUT                                        12 COORD
      DIMENSION X(NSD,1)                                                13 COORD
C                                                                       14 COORD
      WRITE(IOUT,2000) (LABELC(I),I=1,NSD)                              15 COORD
      CALL FGEN(X,NSD)   generate X(NSD, NUMNP)                         16 COORD
      NN=0                                                              17 COORD
      DO 10 N=1,NUMNP                                                   18 COORD
      NN=NN+1                                                           19 COORD
      IF(NN.LE.50) GOTO 10   print 50 lines/page of data.               20 COORD
      NN=1                                                              21 COORD
      WRITE(IOUT,2000) (LABELC(I),I=1,NSD)                              22 COORD
   10 WRITE(IOUT,2010) N,(X(I,N),I=1,NSD)                               23 COORD
      RETURN                                                            24 COORD
C                                                                       25 COORD
 2000 FORMAT(1H1,' N O D A L   C O O R D I N A T E   D A T A '///       26 COORD
     X     4X,' NODE NUMBER ',10X,3(A4,13X)//)                          27 COORD
 2010 FORMAT(6X,I5,10X,3(1PE15.8,2X))                                   28 COORD
      END                                                               29 COORD
                                                                        30 COORD
```

```
C                                                                       1 DIAG
      SUBROUTINE DIAG(IDIAG,NEQ,NA)                                     2 DIAG
C                                                                       3 DIAG
C.... PROGRAM TO COMPUTE DIAGONAL ADDRESSES                             4 DIAG
C                                                                       5 DIAG
      DIMENSION IDIAG(1)                                                6 DIAG
C                                                                       7 DIAG
      NA=1                                                              8 DIAG
      IDIAG(1)=1                                                        9 DIAG
      IF(NEQ.EQ.1) RETURN                                              10 DIAG
      DO 10 I=2,NEQ                                                    11 DIAG
   10 IDIAG(I)=IDIAG(I)+IDIAG(I-1)+1                                   12 DIAG
      NA=IDIAG(NEQ)                                                    13 DIAG
C                                                                      14 DIAG
      RETURN                                                           15 DIAG
      END                                                              16 DIAG


C                                                                       1 DISBC
      SUBROUTINE DISBC(DL,P,S,NEE)                                      2 DISBC
C                                                                       3 DISBC
C.... PROGRAM TO ADJUST LOAD VECTOR FOR PRESCRIBED DISPLACEMENT         4 DISBC
C        BOUNDARY CONDITIONS                                            5 DISBC
C                                                                       6 DISBC
      IMPLICIT REAL*8 (A-H,O-Z)                                         7 DISBC
C                                                                       8 DISBC
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                  9 DISBC
C                                                                      10 DISBC
      DIMENSION DL(NEE),P(NEE),S(NEE,1)                                11 DISBC
C                                                                      12 DISBC
      DO 20 J=1,NEE                                                    13 DISBC
      IF(DL(J).EQ.0.) GOTO 20                                          14 DISBC
      DO 10 I=1,NEE                                                    15 DISBC
   10 P(I)=P(I)-S(I,J)*DL(J)                                           16 DISBC
   20 CONTINUE                                                         17 DISBC
C                                                                      18 DISBC
      RETURN                                                           19 DISBC
      END                                                              20 DISBC


C                                                                       1 DISPTR
      SUBROUTINE DISPTR(B,D,ID,NDOF,NUMNP)                              2 DISPTR
C                                                                       3 DISPTR
C.... PROGRAM TO TRANSFER COMPUTED NODAL DISPLACEMENTS INTO D ARRAY     4 DISPTR
C                                                                       5 DISPTR
      IMPLICIT REAL*8(A-H,O-Z)                                          6 DISPTR
C                                                                       7 DISPTR
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                  8 DISPTR
C                                                                       9 DISPTR
      DIMENSION B(1),D(NDOF,1),ID(NDOF,1)                              10 DISPTR
C                                                                      11 DISPTR
      DO 10 I=1,NDOF                                                   12 DISPTR
      DO 10 J=1,NUMNP                                                  13 DISPTR
      K=ID(I,J)                                                        14 DISPTR
      IF(K.GT.0) D(I,J)=B(K)                                           15 DISPTR
   10 CONTINUE                                                         16 DISPTR
C                                                                      17 DISPTR
      RETURN                                                           18 DISPTR
      END                                                              19 DISPTR


C                                                                       1 DOT
      FUNCTION DOT(A,B,N)                                               2 DOT
C                                                                       3 DOT
C.... PROGRAM TO PERFORM THE DOT PRODUCT OF TWO VECTORS                 4 DOT
C                                                                       5 DOT
      IMPLICIT REAL*8(A-H,O-Z)                                          6 DOT
C                                                                       7 DOT
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                  8 DOT
```

```
C                                                                       9 DOT
      DIMENSION A(1),B(1)                                              10 DOT
C                                                                      11 DOT
      DOT=0.DO                                                         12 DOT
      DO 10 I=1,N                                                      13 DOT
   10 DOT=DOT+A(I)*B(I)                                                14 DOT
C                                                                      15 DOT
      RETURN                                                           16 DOT
      END                                                              17 DOT


C                                                                       1 EL
      SUBROUTINE EL                                                     2 EL
C                                                                       3 EL
C.... PROGRAM TO INPUT AND WRITE ELEMENT GROUP DATA ON DISK             4 EL
C                                                                       5 EL
      COMMON/DEVICE/IEL,IIN,IOUT                                        6 EL
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG  7 EL
      COMMON A(1)                                                       8 EL
C                                                                       9 EL
      REWIND IEL                                                       10 EL
      WRITE(IOUT,2000)                                                 11 EL
      DO 10 NEG=1,NUMEG                                                12 EL
      IF(NEG.GT.1) WRITE(IOUT,2010)                                    13 EL
      WRITE(IOUT,2020) NEG                                             14 EL
      READ(IIN,1000) NPAR                                              15 EL
      CALL ELMLIB                                                      16 EL
      LENGTH=NL-NF+1                                                   17 EL
      IF(LENGTH.GT.MAX) MAX=LENGTH                                     18 EL
   10 WRITE(IEL) LENGTH,NPAR,  (A(I),I=NF,NL)                          19 EL
      RETURN                                                           20 EL
C                                                                      21 EL
 1000 FORMAT(16I5)                                                     22 EL
 2000 FORMAT(1H1,' E L E M E N T   G R O U P   D A T A '///)           23 EL
 2010 FORMAT(1H1)                                                      24 EL
 2020 FORMAT(1HO,' ELEMENT GROUP NUMBER . . . (NEG) = ',I5//)          25 EL
      END                                                              26 EL


C                                                                       1 ELGEN
      SUBROUTINE ELGEN (IEN,MAT,NEN)                                    2 ELGEN
C                                                                       3 ELGEN
C.... PROGRAM TO READ AND GENERATE ELEMENT NODAL DATA AND MATERIAL      4 ELGEN
C        SET NUMBERS                                                    5 ELGEN
C                                                                       6 ELGEN
C        IEN(NEN,NUMEL) = ELEMENT NODE NUMBERS                          7 ELGEN
C        MAT(NUMEL)     = ELEMENT MATERIAL SET                          8 ELGEN
C        NEN            = NUMBER OF NODES PER ELEMENT                   9 ELGEN
C        N              = ELEMENT NUMBER                               10 ELGEN
C        NG             = GENERATION PARAMETER                         11 ELGEN
C                                                                      12 ELGEN
      COMMON/DEVICE/ IEL,IIN,IOUT                                      13 ELGEN
      DIMENSION IEN(NEN,1),MAT(1)                                      14 ELGEN
      DIMENSION IA(32)                                                 15 ELGEN
C                                                                      16 ELGEN
      L=0                                                              17 ELGEN
      LG=0                                                             18 ELGEN
   10 READ(IIN,1000) N,MAT1,(IA(I),I=1,NEN),NG                         19 ELGEN
      IF(N.EQ.0) RETURN                                                20 ELGEN
      MAT(N)=MAT1                                                      21 ELGEN
      DO 30 I=1,NEN                                                    22 ELGEN
   30 IEN(I,N)=IA(I)                                                   23 ELGEN
      IF(LG.EQ.0) GOTO 100                                             24 ELGEN
C                                                                      25 ELGEN
C.... GENERATE DATA                                                    26 ELGEN
C                                                                      27 ELGEN
      N1=L+1                                                           28 ELGEN
      N2=N-1                                                           29 ELGEN
      DO 20 J=N1,N2                                                    30 ELGEN
      MAT(J)=MAT(L)                                                    31 ELGEN
      JJ=J-1                                                           32 ELGEN
```

```
      DO 20 I=1,NEN                                                    33 ELGEN
   20 IEN(I,J)=IEN(I,JJ)+LG                                            34 ELGEN
C                                                                      35 ELGEN
  100 L=N                                                              36 ELGEN
      LG=NG                                                            37 ELGEN
      GO TO 10                                                         38 ELGEN
C                                                                      39 ELGEN
 1000 FORMAT(16I5)                                                     40 ELGEN
      END                                                              41 ELGEN


C
      SUBROUTINE ELMLIB                                                1 ELMLIB
C                                                                      2 ELMLIB
C.... PROGRAM TO CALL THE ELEMENT ROUTINES                             3 ELMLIB
C                                                                      4 ELMLIB
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG 5 ELMLIB
C                                                                      6 ELMLIB
      I=NPAR(1)                                                        7 ELMLIB
      GOTO(1,2,3                                          ),I          8 ELMLIB
    1 CALL TRUSS                                                       9 ELMLIB
      RETURN                                                          10 ELMLIB
    2 CALL BEAM                                                       11 ELMLIB
      RETURN                                                          12 ELMLIB
    3 CALL PLANE                                                      13 ELMLIB
C                                                                     14 ELMLIB
      RETURN                                                          15 ELMLIB
      END                                                             16 ELMLIB
                                                                      17 ELMLIB


C
      SUBROUTINE ELREAD(E)                                             1 ELREAD
C                                                                      2 ELREAD
C.... PROGRAM TO READ ELEMENT GROUP DATA FROM AUXILIARY STORAGE DEVICE 3 ELREAD
C        NO. IEL                                                       4 ELREAD
C                                                                      5 ELREAD
      COMMON/DEVICE/ IEL,IIN,IOUT                                      6 ELREAD
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG 7 ELREAD
      DIMENSION E(1)                                                   8 ELREAD
C                                                                      9 ELREAD
      REWIND IEL                                                      10 ELREAD
C                                                                     11 ELREAD
      DO 10 NEG=1,NUMEG                                               12 ELREAD
      READ(IEL) J,NPAR,(E(I),I=1,J)                                   13 ELREAD
   10 CALL ELMLIB                                                     14 ELREAD
C                                                                     15 ELREAD
      RETURN                                                          16 ELREAD
      END                                                             17 ELREAD
                                                                      18 ELREAD


C
      SUBROUTINE ERROR (I,N)                                           1 ERROR
C                                                                      2 ERROR
C.... PROGRAM TO PRINT ERROR MESSAGES WHEN HIGH-SPEED STORAGE IS EXCEEDED 3 ERROR
C                                                                      4 ERROR
      COMMON/DEVICE/ IEL,IIN,IOUT                                      5 ERROR
C                                                                      6 ERROR
      WRITE(IOUT,2000) I                                               7 ERROR
      GOTO (1,2,3,4,5                                     ),N          8 ERROR
    1 WRITE(IOUT,2010)                                                 9 ERROR
      STOP                                                            10 ERROR
    2 WRITE(IOUT,2020)                                                11 ERROR
      STOP                                                            12 ERROR
    3 WRITE(IOUT,2030)                                                13 ERROR
      STOP                                                            14 ERROR
    4 WRITE(IOUT,2040)                                                15 ERROR
      STOP                                                            16 ERROR
    5 WRITE(IOUT,2050)                                                17 ERROR
      STOP                                                            18 ERROR
                                                                      19 ERROR
```

```
C                                                                       20 ERROR
 2000 FORMAT(//' AVAILABLE STORAGE INSUFFICIENT BY ',I10,               21 ERROR
     X'   SINGLE-PRECISION WORDS')                                      22 ERROR
 2010 FORMAT(/' FOR INPUT OF NODAL COORDINATE AND BOUNDARY DATA')       23 ERROR
 2020 FORMAT(/' FOR INPUT OF APPLIED FORCE AND PRESCRIBED DISPLACEMENT D 24 ERROR
     XATA')                                                             25 ERROR
 2030 FORMAT(/' FOR STORAGE IDIAG ARRAY')                               26 ERROR
 2040 FORMAT(/' FOR INPUT OF ELEMENT DATA')                             27 ERROR
 2050 FORMAT(/' FOR ASSEMBLY OF STIFFNESS AND LOAD')                    28 ERROR
C                                                                       29 ERROR
      RETURN                                                            30 ERROR
      END                                                               31 ERROR


C                                                                        1 FGEN
      SUBROUTINE FGEN(A,M)                                               2 FGEN
C                                                                        3 FGEN
C.... PROGRAM TO READ AND GENERATE FLOATING POINT NODAL DATA             4 FGEN
C                                                                        5 FGEN
C        A  = INPUT ARRAY                                                6 FGEN
C        M  = NUMBER OF ROWS IN A                                        7 FGEN
C        N  = NODE NUMBER                                                8 FGEN
C        NG = GENERATION INCREMENT                                       9 FGEN
C                                                                       10 FGEN
      IMPLICIT REAL*8(A-H,O-Z)                                          11 FGEN
C                                                                       12 FGEN
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                  13 FGEN
C                                                                       14 FGEN
      COMMON /DEVICE/ IEL,IIN,IOUT                                      15 FGEN
      DIMENSION A(M,1)                                                  16 FGEN
      DIMENSION F(7)                                                    17 FGEN
C                                                                       18 FGEN
      L=0                                                               19 FGEN
      LG=0                                                              20 FGEN
   10 READ(IIN,1000) N,NG,(F(I),I=1,M)                                  21 FGEN
      IF(N.EQ.0) RETURN                                                 22 FGEN
      DO 30 I=1,M                                                       23 FGEN
   30 A(I,N)=F(I)                                                       24 FGEN
      IF(LG.EQ.0) GOTO 100                                              25 FGEN
C                                                                       26 FGEN
C.... GENERATE DATA                                                     27 FGEN
C                                                                       28 FGEN
      NI=(N-L)/LG                                                       29 FGEN
      N1=L+LG                                                           30 FGEN
      N2=N-LG                                                           31 FGEN
      DO 20 I=1,M                                                       32 FGEN
      DA=(A(I,N)-A(I,L))/NI                                             33 FGEN
      DO 20 J=N1,N2,LG                                                  34 FGEN
      JJ=J-LG                                                           35 FGEN
   20 A(I,J)=A(I,JJ)+DA                                                 36 FGEN
C                                                                       37 FGEN
  100 L=N                                                               38 FGEN
      LG=NG                                                             39 FGEN
      GO TO 10                                                          40 FGEN
C                                                                       41 FGEN
 1000 FORMAT(2I5,7(F10.0))                                              42 FGEN
      END                                                               43 FGEN


C                                                                        1 FORCE
      SUBROUTINE FORCE(F,NDOF,NUMNP)                                     2 FORCE
C                                                                        3 FORCE
C.... PROGRAM FOR READING AND PRINTING APPLIED FORCE AND PRESCRIBED      4 FORCE
C        DISPLACEMENT DATA                                               5 FORCE
C        F(NDOF,NUMNP) = APPLIED FORCES AND PRESCRIBED DISPLACEMENTS     6 FORCE
C                                                                        7 FORCE
      IMPLICIT REAL*8(A-H,O-Z)                                           8 FORCE
C                                                                        9 FORCE
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                  10 FORCE
C                                                                       11 FORCE
```

```
      COMMON/LABELS/LABELC(3),LABELD(6)                                  12 FORCE
      COMMON/DEVICE/IEL,IIN,IOUT                                        13 FORCE
      DIMENSION F(NDOF,1)                                               14 FORCE
      CALL CLEAR(F,NDOF*NUMNP)                                          15 FORCE
C                                                                       16 FORCE
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                             17 FORCE
      CALL FGEN(F,NDOF)                                                 18 FORCE
      NN=0                                                              19 FORCE
      DO 10 N=1,NUMNP                                                   20 FORCE
      IF(NN.LE.50) GOTO 20                                              21 FORCE
      NN=1                                                              22 FORCE
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                             23 FORCE
   20 DO 30 I=1,NDOF                                                    24 FORCE
      IF(F(I,N)) 40,30,40                                               25 FORCE
   30 CONTINUE                                                          26 FORCE
      GOTO 10                                                           27 FORCE
   40 NN=NN+1                                                           28 FORCE
      WRITE(IOUT,2010) N,(F(I,N),I=1,NDOF)                              29 FORCE
   10 CONTINUE                                                          30 FORCE
      RETURN                                                            31 FORCE
C                                                                       32 FORCE
 2000 FORMAT(1H1,'  N O D A L   F O R C E S   A N D   D I S P L A C E M  33 FORCE
     X E N T S '///                                                     34 FORCE
     X    4X,' NODE NUMBER',10X,6(A4,13X)/)                             35 FORCE
 2010 FORMAT(6X,I5,10X,6(1PE15.8,2X))                                   36 FORCE
      END                                                               37 FORCE
```

```
C                                                                        1 ICLEAR
      SUBROUTINE ICLEAR(IA,M)                                            2 ICLEAR
C                                                                        3 ICLEAR
C.... PROGRAM TO CLEAR AN INTERGER ARRAY                                 4 ICLEAR
C                                                                        5 ICLEAR
      DIMENSION IA(1)                                                    6 ICLEAR
      DO 10 I=1,M                                                        7 ICLEAR
   10 IA(I)=0                                                            8 ICLEAR
C                                                                        9 ICLEAR
      RETURN                                                            10 ICLEAR
      END                                                               11 ICLEAR
```

```
C                                                                        1 IGEN
      SUBROUTINE IGEN(IA,M)                                              2 IGEN
C                                                                        3 IGEN
C.... PROGRAM TO READ AND GENERATE INTEGER NODAL DATA                    4 IGEN
C                                                                        5 IGEN
C        IA = INPUT ARRAY                                                6 IGEN
C        M  = NUMBER OF ROWS IN IA                                       7 IGEN
C        N  = NODE NUMBER                                                8 IGEN
C        NG = GENERATION INCREMENT                                       9 IGEN
C                                                                       10 IGEN
      COMMON/DEVICE/IEL,IIN,IOUT                                        11 IGEN
      DIMENSION IA(M,1)                                                 12 IGEN
      DIMENSION IB(32)                                                  13 IGEN
C                                                                       14 IGEN
      L=0                                                               15 IGEN
      LG=0                                                              16 IGEN
   10 READ(IIN,1000) N,NG,(IB(I),I=1,M)                                 17 IGEN
      IF(N.EQ.0) RETURN                                                 18 IGEN
      DO 30 I=1,M                                                       19 IGEN
   30 IA(I,N)=IB(I)                                                     20 IGEN
      IF(LG.EQ.0) GOTO 100                                              21 IGEN
C                                                                       22 IGEN
C.... GENERATE DATA                                                     23 IGEN
C                                                                       24 IGEN
      N1=L+LG                                                           25 IGEN
      N2=N-MOD(N-L,LG)                                                  26 IGEN
      IF(N2.EQ.N) N2=N-LG                                               27 IGEN
      DO 20 I=1,M                                                       28 IGEN
      DO 20 J=N1,N2,LG                                                  29 IGEN
   20 IA(I,J)=IA(I,L)                                                   30 IGEN
```

```
C
  100 L=N                                                                  31 IGEN
      LG=NG                                                                32 IGEN
      GO TO 10                                                             33 IGEN
C                                                                          34 IGEN
 1000 FORMAT(1615)                                                         35 IGEN
      END                                                                  36 IGEN
                                                                           37 IGEN


C
      SUBROUTINE LCOORD(IEN,X,XL,NEC,NEN,NSD)                               1 LCOORD
C                                                                           2 LCOORD
C.... PROGRAM TO LOCALIZE COORDINATE DATA                                   3 LCOORD
C                                                                           4 LCOORD
C        XL(NSD,NEN) = ELEMENT NODAL COORDINATES                            5 LCOORD
C                                                                           6 LCOORD
      IMPLICIT REAL*8 (A-H,O-Z)                                             7 LCOORD
C                                                                           8 LCOORD
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                      9 LCOORD
C                                                                          10 LCOORD
      DIMENSION  IEN(1),X(NSD,1),XL(NSD,1)                                 11 LCOORD
C                                                                          12 LCOORD
      CALL CLEAR(XL,NEC)                                                   13 LCOORD
      DO 10 J=1,NEN                                                        14 LCOORD
      K=IEN(J)                                                             15 LCOORD
      DO 10 I=1,NSD                                                        16 LCOORD
   10 XL(I,J)=X(I,K)                                                       17 LCOORD
C                                                                          18 LCOORD
      RETURN                                                               19 LCOORD
      END                                                                  20 LCOORD
                                                                           21 LCOORD


C
      SUBROUTINE LDISBC(DL,F,IEN,LM,NDOF,NEE,NED,NEN)                       1 LDISBC
C                                                                           2 LDISBC
C.... PROGRAM TO LOCALIZE PRESCRIBED DISPLACEMENT BOUNDARY CONDITIONS       3 LDISBC
C                                                                           4 LDISBC
C        DL(NED,NEN)=ELEMENT PRESCRIBED DISPLACEMENT BOUNDARY CONDITION     5 LDISBC
C        ARRAY                                                              6 LDISBC
C                                                                           7 LDISBC
      IMPLICIT REAL *8 (A-H,O-Z)                                            8 LDISBC
C                                                                           9 LDISBC
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                     10 LDISBC
C                                                                          11 LDISBC
      DIMENSION DL(NED,1),F(NDOF,1),IEN(1),LM(NED,1)                       12 LDISBC
C                                                                          13 LDISBC
      CALL CLEAR(DL,NEE)                                                   14 LDISBC
C                                                                          15 LDISBC
      DO 10 I=1,NED                                                        16 LDISBC
      DO 10 J=1,NEN                                                        17 LDISBC
      IF(LM(I,J).GT.0) GOTO 10      if global eqn no. ≠ 0  the node is free & value  18 LDISBC
      K=IEN(J)                                              is that of free  19 LDISBC
      DL(I,J)=F(I,K)                if global eqn no. = 0  the node is fixed & value is  20 LDISBC
   10 CONTINUE                                  find global node no. K = IEN(J)  21 LDISBC
C                                               that of nodal                   22 LDISBC
      RETURN                                    transfer to local displ vector DL(J) ← F(K)  23 LDISBC
      END                                                                  24 LDISBC
                                                                           25 LDISBC


C
      SUBROUTINE LDISP(DL,D,IEN,NDOF,NED,NEN)                               1 LDISP
C                                                                           2 LDISP
C.... PROGRAM TO LOCALIZE DISPLACEMENTS                                     3 LDISP
C                                                                           4 LDISP
C        DL(NED,NEN) = ELEMENT DISPLACEMENT ARRAY                           5 LDISP
C                                                                           6 LDISP
      IMPLICIT REAL*8(A-H,O-Z)                                              7 LDISP
C                                                                           8 LDISP
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                      9 LDISP
                                                                           10 LDISP
```

```
C
      DIMENSION IEN(1),DL(NED,1),D(NDOF,1)                              11 LDISP
C                                                                       12 LDISP
      DO 10 J=1,NEN                                                     13 LDISP
      K=IEN(J)                                                          14 LDISP
      DO 10 I=1,NED                                                     15 LDISP
   10 DL(I,J)=D(I,K)                                                    16 LDISP
C                                                                       17 LDISP
      RETURN                                                            18 LDISP
      END                                                               19 LDISP
                                                                        20 LDISP


C
      SUBROUTINE LOAD(ID,F,B,NDOF,NUMNP)                                 1 LOAD
C                                                                        2 LOAD
C.... PROGRAM TO TRANSFER NODAL APPLIED FORCES INTO RIGHT HAND SIDE VECTOR 3 LOAD
C                                                                        4 LOAD
C        B(NEQ) = RIGHT-HAND SIDE VECTOR                                 5 LOAD
C                                                                        6 LOAD
      IMPLICIT REAL*8 (A-H,O-Z)                                          7 LOAD
C                                                                        8 LOAD
C.... REMOVE ABOVE CARD FOR SINGLE-PRECISION OPERATION                   9 LOAD
C                                                                       10 LOAD
      DIMENSION ID(NDOF,1),F(NDOF,1),B(1)                               11 LOAD
C                                                                       12 LOAD
      DO 10 I=1,NDOF                                                    13 LOAD
      DO 10 J=1,NUMNP                                                   14 LOAD
      K=ID(I,J)                                                         15 LOAD
   10 IF(K.GT.0) B(K)=F(I,J)                                            16 LOAD
C                                                                       17 LOAD
      RETURN                                                            18 LOAD
      END                                                               19 LOAD
                                                                        20 LOAD


C
      SUBROUTINE LOCAL(ID,IEN,LM,NDOF,NED,NEN,NUMEL)                     1 LOCAL
C                                                                        2 LOCAL
C.... SUBROUTINE TO LOCALIZE ID ARRAY                                    3 LOCAL
C                                                                        4 LOCAL
      DIMENSION ID(NDOF,1),IEN(NEN,1),LM(NED,NEN,1)                      5 LOCAL
C                                                                        6 LOCAL
      DO 10 K=1,NUMEL                                                    7 LOCAL
      DO 10 J=1,NEN                                                      8 LOCAL
      NN=IEN(J,K)                                                        9 LOCAL
      DO 10 I=1,NED                                                     10 LOCAL
   10 LM(I,J,K)=ID(I,NN)                                                11 LOCAL
C                                                                       12 LOCAL
      RETURN                                                            13 LOCAL
      END                                                               14 LOCAL
                                                                        15 LOCAL

NN global node no ←
IEN(J,K) has global node no. of node J, element K
ID(I,NN) has global eqn no. of degree of freedom I of global node NN
LM(I,J,K) has global eqn no. I th degree of freedom of node J of element K


C
      SUBROUTINE PACK(S,NP,NU)                                           1 PACK
C                                                                        2 PACK
C.... PROGRAM TO PACK AN NU*NU MATRIX INTO AN NP*NP MATRIX               3 PACK
C                                                                        4 PACK
      IMPLICIT REAL*8(A-H,O-Z)                                           5 PACK
C                                                                        6 PACK
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                   7 PACK
C                                                                        8 PACK
      DIMENSION S(1)                                                     9 PACK
      DO 10 J=1,NP                                                      10 PACK
      K=(J-1)*NP                                                        11 PACK
      L=(J-1)*NU                                                        12 PACK
      DO 10 I=1,NP                                                      13 PACK
      K=K+1                                                             14 PACK
      L=L+1                                                             15 PACK
   10 S(K)=S(L)                                                         16 PACK
C                                                                       17 PACK
      RETURN                                                            18 PACK
      END                                                               19 PACK
                                                                        20 PACK
```

```
C
      SUBROUTINE PREAD(IA,JA,A,M)                                          1 PREAD
C                                                                          2 PREAD
C.... PROGRAM TO READ ELEMENT CONSISTENT LOAD DATA                         3 PREAD
C                                                                          4 PREAD
      IMPLICIT REAL*8(A-H,O-Z)                                             5 PREAD
C                                                                          6 PREAD
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                     7 PREAD
C                                                                          8 PREAD
C        A =INPUT ARRAY                                                    9 PREAD
C        IA= ELEMENT NUMBER                                               10 PREAD
C        JA= ELEMENT FACE/SIDE NUMBER                                     11 PREAD
C        M = NUMBER OF LOADS PER FACE/SIDE                                12 PREAD
C        N = LOAD NUMBER                                                  13 PREAD
C                                                                         14 PREAD
      COMMON/DEVICE/ IEL,IIN,IOUT                                         15 PREAD
      DIMENSION A(M,1),IA(1),JA(1),B(6)                                   16 PREAD
C                                                                         17 PREAD
      N=0                                                                 18 PREAD
   10 N=N+1                                                               19 PREAD
      READ (IIN,1000) II,JJ,(B(I),I=1,M)                                 20 PREAD
      IF (II.EQ.0) RETURN                                                 21 PREAD
      IA(N)=II                                                            22 PREAD
      JA(N)=JJ                                                            23 PREAD
      DO 20 I=1,M                                                         24 PREAD
   20 A(I,N)=B(I)                                                         25 PREAD
      GO TO 10                                                            26 PREAD
C                                                                         27 PREAD
 1000 FORMAT(2I5,10X,6F10.0)                                              28 PREAD
      END                                                                 29 PREAD
                                                                          30 PREAD


C                                                                          1 PRINC
      SUBROUTINE PRINC(N,S,P)                                              2 PRINC
C                                                                          3 PRINC
C.... PROGRAM TO COMPUTE PRINCIPAL VALUES OF SYMMETRIC SECOND RANK TENSOR  4 PRINC
C                                                                          5 PRINC
C        S = SYMMETRIC SECOND-RANK TENSOR STORED AS A VECTOR               6 PRINC
C        N = NUMBER OF DIMENSIONS (2 OR 3)                                 7 PRINC
C        P = PRINCIPAL VALUES                                              8 PRINC
C                                                                          9 PRINC
C.... THE COMPONENTS OF S MUST BE STORED IN THE FOLLOWING ORDERS           10 PRINC
C                                                                         11 PRINC
C        2-D PROBLEMS,  S11,S12,S22                                       12 PRINC
C        3-D PROBLEMS,  S11,S12,S13,S22,S23,S33                           13 PRINC
C                                                                         14 PRINC
      IMPLICIT REAL*8(A-H,O-Z)                                            15 PRINC
C                                                                         16 PRINC
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                    17 PRINC
C                                                                         18 PRINC
      DIMENSION S(1),P(1)                                                 19 PRINC
      DATA RT2/1.4142135623730D0/,PI23/2.0943951023932D0/                 20 PRINC
C                                                                         21 PRINC
      IF(N.EQ.3) GOTO 10                                                  22 PRINC
C                                                                         23 PRINC
C.... 2-D PROGRAM                                                         24 PRINC
C                                                                         25 PRINC
      A=22.5D0/DATAN(1.D0)                                                26 PRINC
      X=0.5D0*(S(1)+S(3))                                                 27 PRINC
      Y=0.5D0*(S(1)-S(3))                                                 28 PRINC
      R=DSQRT(Y*Y+S(2)*S(2))                                              29 PRINC
      P(1)=X+R                                                            30 PRINC
      P(2)=X-R                                                            31 PRINC
      P(3)=45.D0                                                          32 PRINC
      IF(Y.NE.0.D0.OR.S(2).NE.0.D0) P(3)=A*DATAN2(S(2),Y)                 33 PRINC
      IF (S(2).EQ.0.D0)  P(3)=0.D0                                        34 PRINC
      P(4)=R                                                              35 PRINC
      RETURN                                                              36 PRINC
C                                                                         37 PRINC
C.... 3-D PROBLEM                                                         38 PRINC
C                                                                         39 PRINC
```

```
   10 R=0.D0                                                                 40 PRINC
      X=(S(1)+S(4)+S(6))/3.D0                                                41 PRINC
      Y=S(1)*(S(4)+S(6))+S(4)*S(6)-S(2)*S(2)-S(3)*S(3)-S(5)*S(5)             42 PRINC
      Z=S(1)*S(4)*S(6)+2.D0*S(2)*S(3)*S(5)-S(1)*S(5)*S(5)                    43 PRINC
     X      -S(4)*S(3)*S(3)-S(6)*S(2)*S(2)                                   44 PRINC
      T=3.D0*X*X-Y                                                           45 PRINC
      U=0.D0                                                                 46 PRINC
      IF(T.EQ.0.D0) GO TO 20                                                 47 PRINC
      U=DSQRT(2.D0*T/3.D0)                                                   48 PRINC
      A=(Z+(T-X*X)*X)*RT2/U**3                                               49 PRINC
      R=DSQRT(DABS(1.D0-A*A))                                                50 PRINC
      R=DATAN2(R,A)/3.D0                                                     51 PRINC
   20 P(1)=X+U*RT2*DCOS(R)                                                   52 PRINC
      P(2)=X+U*RT2*DCOS(R-PI23)                                              53 PRINC
      P(3)=X+U*RT2*DCOS(R+PI23)                                              54 PRINC
C                                                                            55 PRINC
      RETURN                                                                 56 PRINC
      END                                                                    57 PRINC


C
      SUBROUTINE PRINTD(D,NDOF,NUMNP)                                         1 PRINTD
C                                                                             2 PRINTD
C.... PROGRAM TO PRINT DISPLACEMENTS                                          3 PRINTD
C                                                                             4 PRINTD
      IMPLICIT REAL*8(A-H,O-Z)                                                5 PRINTD
C                                                                             6 PRINTD
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                        7 PRINTD
C                                                                             8 PRINTD
      DIMENSION D(NDOF,1)                                                     9 PRINTD
      COMMON/DEVICE/IEL,IIN,IOUT                                             10 PRINTD
      COMMON/LABELS/LABELC(3),LABELD(6)                                      11 PRINTD
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                                  12 PRINTD
      NN=0                                                                   13 PRINTD
      DO 10 N=1,NUMNP                                                        14 PRINTD
      NN=NN+1                                                                15 PRINTD
      IF(NN.LE.50) GOTO 10                                                   16 PRINTD
      NN=1                                                                   17 PRINTD
      WRITE(IOUT,2000) (LABELD(I),I=1,NDOF)                                  18 PRINTD
   10 WRITE(IOUT,2010)  N,(D(I,N),I=1,NDOF)                                  19 PRINTD
C                                                                            20 PRINTD
      RETURN                                                                 21 PRINTD
 2000 FORMAT(1H1,' D I S P L A C E M E N T S '///                            22 PRINTD
     X    4X,'NODE NUMBER',10X,6(A4,13X)/)                                  23 PRINTD
 2010 FORMAT(6X,I5,10X,6(1PE15.8,2X))                                       24 PRINTD
      END                                                                    25 PRINTD
                                                                             26 PRINTD


C
      SUBROUTINE PRINTP(A,IDIAG,NEQ)                                          1 PRINTP
C                                                                             2 PRINTP
C.... PROGRAM TO PRINT ARRAY D AFTER FACTORIZATION A=(U)T*D*U                 3 PRINTP
C                                                                             4 PRINTP
      IMPLICIT REAL*8(A-H,O-Z)                                                5 PRINTP
C                                                                             6 PRINTP
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                        7 PRINTP
C                                                                             8 PRINTP
      COMMON/DEVICE/IEL,IIN,IOUT                                              9 PRINTP
      DIMENSION A(1),IDIAG(1)                                                10 PRINTP
C                                                                            11 PRINTP
      WRITE(IOUT,2000)                                                       12 PRINTP
      NN=0                                                                   13 PRINTP
      DO 10 N=1,NEQ                                                          14 PRINTP
      NN=NN+1                                                                15 PRINTP
      IF(NN.LE.50) GOTO 20                                                   16 PRINTP
      WRITE(IOUT,2000)                                                       17 PRINTP
   20 I=IDIAG(N)                                                             18 PRINTP
      NN=0                                                                   19 PRINTP
      D=A(I)                                                                 20 PRINTP
   10 WRITE(IOUT,2010) N,D                                                   21 PRINTP
      RETURN                                                                 22 PRINTP
                                                                             23 PRINTP
```

```
C
 2000 FORMAT(1H1,' ARRAY D(NEQ) OF  FACTORIZATION  A=(U)T*D*U'///)                  24 PRINTP
 2010 FORMAT(1X,I5,4X,1PE20.8)                                                     25 PRINTP
C                                                                                  26 PRINTP
      END                                                                          27 PRINTP
                                                                                   28 PRINTP


C
      SUBROUTINE SECOND(T)                                                         1 SECOND
C                                                                                  2 SECOND
C.... PROGRAM TO DETERMINE TIME                                                    3 SECOND
C                                                                                  4 SECOND
C     SUBROUTINE RTIME IS USED ON CAL TECH IBM 370 SYSTEM                          5 SECOND
C                                                                                  6 SECOND
      INTEGER RTIME                                                                7 SECOND
      IX=RTIME(ZDUM)                                                               8 SECOND
      T=IX/1000.D0                                                                 9 SECOND
C                                                                                 10 SECOND
      RETURN                                                                      11 SECOND
      END                                                                         12 SECOND
                                                                                  13 SECOND


C                                  inp   curr      inp
      SUBROUTINE SHAP20(S,T,X,DET,SH,NEN,INC,IEN,QUAD)                             1 SHAP20
C                                                                                  2 SHAP20
C.... PROGRAM TO COMPUTE SHAPE FUNCTIONS FOR QUADRILATERAL                         3 SHAP20
C                                                                                  4 SHAP20
C        S,T         = NATURAL COORDINATES                                         5 SHAP20
C        SH(NSD,I)   = DERIVATIVES OF SHAPE FUNCTIONS                              6 SHAP20
C        SH(3,I)     = SHAPE FUNCTIONS                                             7 SHAP20
C        XS(NSD,NSD)= JACOBIAN MATRIX                  X_ξ                         8 SHAP20
C        DET         = JACOBIAN DETERMINANT                                        9 SHAP20
C        X(NSD,NEN) = GLOBAL COORDINATES                                          10 SHAP20
C                                                                                 11 SHAP20
      IMPLICIT REAL*8(A-H,O-Z)                                                    12 SHAP20
C                                                                                 13 SHAP20
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                            14 SHAP20
C                                                                                 15 SHAP20
      LOGICAL QUAD                                                                16 SHAP20
      COMMON /DEVICE/ IEL,IIN,IOUT                                                17 SHAP20
      DIMENSION SA(4),TA(4),SH(3,1),X(2,1),XS(2,2)                                18 SHAP20
      DIMENSION IEN(1)                                                            19 SHAP20
      DATA SA/-0.500,0.500,0.500,-0.500/,TA/-0.500,-0.500,0.500,0.500/            20 SHAP20
      DO 10 I=1,4                                                                 21 SHAP20
      SH(3,I)=(0.500+SA(I)*S)*(0.500+TA(I)*T)   Na(ξ_ℓ, η_ℓ)                      22 SHAP20
      SH(1,I)=SA(I)*(0.500+TA(I)*T)             Na,ξ (ξ_ℓ, η_ℓ)                   23 SHAP20
   10 SH(2,I)=TA(I)*(0.500+SA(I)*S)                                               24 SHAP20
      IF(QUAD) GOTO 30          FOR TRIANGLE    Na,η (ξ_ℓ, η_ℓ)                  25 SHAP20
      DO 20 I=1,3                                                                 26 SHAP20
      SH(I,3)=SH(I,3)+SH(I,4)                                                     27 SHAP20
   20 SH(I,4)=0.D0          ← IF MORE THAN 4 NODE ELEM.                         28 SHAP20
   30 IF(NEN.GT.4) CALL SHAP21(S,T,SH,NEN,IEN)                                    29 SHAP20
      DO 40 I=1,2                                                                 30 SHAP20
      DO 40 J=1,2                                                                 31 SHAP20
      XS(I,J)=0.D0   — zero out jacob.   X(ξ_ℓ,η_ℓ) = Σ_a Na(ξ_ℓ,η_ℓ) X_a^e      32 SHAP20
      DO 40 K=1,NEN                                                               33 SHAP20
   40 XS(I,J)=XS(I,J)+X(I,K)*SH(J,K)   - nodal do loop   Σ Na,ξ(ξ_ℓ,η_ℓ) X_a^e = X_,ξ(ξ_ℓ,η_ℓ) 34 SHAP20
      DET=XS(1,1)*XS(2,2)-XS(1,2)*XS(2,1) - compute jacob at (ξ_ℓ,η_ℓ) is        35 SHAP20
      DO 50 I=1,2                                                                 36 SHAP20
      DO 50 J=1,2                                                                 37 SHAP20
   50 XS(I,J)=XS(I,J)/DET   - compute inverse   ξ,x at (ξ_ℓ, η_ℓ)                38 SHAP20
      DO 60 I=1,NEN                                                               39 SHAP20
      TEMP=XS(2,2)*SH(1,I)-XS(2,1)*SH(2,I)   Na,x(ξ_ℓ,η_ℓ)                        40 SHAP20
      SH(2,I)=-XS(1,2)*SH(1,I)+XS(1,1)*SH(2,I)  Na,y                              41 SHAP20
   60 SH(1,I)=TEMP                                                                42 SHAP20
      IF(INC.EQ.0) RETURN                                                         43 SHAP20
C                                                                                 44 SHAP20
C.... INCOMPATIBLE MODES     add incomp modes.                                    45 SHAP20
C                                                                                 46 SHAP20
                                                                                  47 SHAP20
```

```
      IF(QUAD) GOTO 80                                                  SHAP20 48
      DO 70 I=1,3                                                       SHAP20 49
      DO 70 J=5,6                                                       SHAP20 50
   70 SH(I,J)=0.D0                                                      SHAP20 51
      RETURN                                                            SHAP20 52
   80 SH(1,5)=-S-S          N5,ξ                                        SHAP20 53
      SH(2,5)=0.D0                                                      SHAP20 54
      SH(3,5)=1.D0-S*S      N5                                          SHAP20 55
      SH(1,6)=0.D0                                                      SHAP20 56
      SH(2,6)=-T-T          N6,η                                        SHAP20 57
      SH(3,6)=1.D0-T*T      N6                                          SHAP20 58
      XS(1,1)=0.25D0*(-X(1,1)+X(1,2)+X(1,3)-X(1,4))                     SHAP20 59
      XS(1,2)=0.25D0*(-X(1,1)-X(1,2)+X(1,3)+X(1,4))                     SHAP20 60
      XS(2,1)=0.25D0*(-X(2,1)+X(2,2)+X(2,3)-X(2,4))   } x_{a,ξ}         SHAP20 61
      XS(2,2)=0.25D0*(-X(2,1)-X(2,2)+X(2,3)+X(2,4))                     SHAP20 62
      DO 90 I=5,6                                                       SHAP20 63
      TEMP=(XS(2,2)*SH(1,I)-XS(2,1)*SH(2,I))/DET      Na,x              SHAP20 64
      SH(2,I)=(-XS(1,2)*SH(1,I)+XS(1,1)*SH(2,I))/DET    Na,y            SHAP20 65
   90 SH(1,I)=TEMP                                                      SHAP20 66
      RETURN                                                            SHAP20 67
      END                                                               SHAP20 68
```

```
C                                                                       SHAP21  1
      SUBROUTINE SHAP21 (S,T,SH,NEN,IEN)                                SHAP21  2
C                                                                       SHAP21  3
C.... PROGRAM TO COMPUTE SHAPE FUNCTIONS FOR QUADRILATERAL              SHAP21  4
C        SHAPE FUNCTIONS  5  TO  8                                      SHAP21  5
C                                                                       SHAP21  6
      IMPLICIT REAL*8(A-H,O-Z)                                          SHAP21  7
C                                                                       SHAP21  8
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                  SHAP21  9
C                                                                       SHAP21 10
      DIMENSION SH(3,1),IEN(1)                                          SHAP21 11
      SS=(1.D0-S*S)/2.D0                                                SHAP21 12
      TT=(1.D0-T*T)/2.D0                                                SHAP21 13
      DO 10 I=5,8                                                       SHAP21 14
      DO 10 J=1,3                                                       SHAP21 15
   10 SH(J,I)=0.D0                                                      SHAP21 16
      IF(IEN(5).EQ.0) GOTO 20                                           SHAP21 17
      SH(1,5)=-S*(1.D0-T)                                               SHAP21 18
      SH(2,5)=-SS                                                       SHAP21 19
      SH(3,5)=SS*(1.D0-T)                                               SHAP21 20
   20 IF(NEN.LT.6) GOTO 50                                              SHAP21 21
      IF(IEN(6).EQ.0) GOTO 30                                           SHAP21 22
      SH(1,6)=TT                                                        SHAP21 23
      SH(2,6)=-T*(1.D0+S)                                               SHAP21 24
      SH(3,6)=TT*(1.D0+S)                                               SHAP21 25
   30 IF(NEN.LT.7) GOTO 50                                              SHAP21 26
      IF(IEN(7).EQ.0) GOTO 40                                           SHAP21 27
      SH(1,7)=-S*(1.D0+T)                                               SHAP21 28
      SH(2,7)=SS                                                        SHAP21 29
      SH(3,7)=SS*(1.D0+T)                                               SHAP21 30
   40 IF(NEN.LT.8) GOTO 50                                              SHAP21 31
      IF(IEN(8).EQ.0) GOTO 50                                           SHAP21 32
      SH(1,8)=-TT                                                       SHAP21 33
      SH(2,8)=-T*(1.D0-S)                                               SHAP21 34
      SH(3,8)=TT*(1.D0-S)                                               SHAP21 35
   50 K=8                                                               SHAP21 36
      DO 70 I=1,4                                                       SHAP21 37
      L=I+4                                                             SHAP21 38
      DO 60 J=1,3                                                       SHAP21 39
   60 SH(J,I)=SH(J,I)-0.500*(SH(J,K)+SH(J,L))                           SHAP21 40
   70 K=L                                                               SHAP21 41
C                                                                       SHAP21 42
      RETURN                                                            SHAP21 43
      END                                                               SHAP21 44
```

```
C
      SUBROUTINE SOLVE (A,B,IDIAG,NEQ,FACT,BACK)                          1 SOLVE
C                                                                         2 SOLVE
C.... PROGRAM TO PERFORM  A =(U)T*D*U FACTORIZATION AND/OR FORWARD         3 SOLVE
C        REDUCTION/BACKSUBSTITUTION OF SYMMETRIC POSITIVE DEFINITE        4 SOLVE
C        SYSTEM OF EQUATIONS                                              5 SOLVE
C                                                                         6 SOLVE
C        A(NA)       = COEFFICIENT MATRIX STORED IN COMPACTED COLUMN FORM  7 SOLVE
C                      (AFTER FACTORIZATION CONTAINS D AND U)             8 SOLVE
C        B(NEQ)      = RIGHT SIDE VECTOR (AFTER BACKSUBSTITUTION SOLUTION  9 SOLVE
C                      VECTOR)                                           10 SOLVE
C        IDIAG(NEQ) = ADDRESSES OF DIAGONAL TERMS IN A(NA)               11 SOLVE
C        NEQ         = NUMBER OF EQUATIONS                               12 SOLVE
C        FACT        = .TRUE.  , FACTOR A(NA)                            13 SOLVE
C                      .FALSE. , DO NOT FACTOR A(NA)                     14 SOLVE
C        BACK        = .TRUE.  , FORWARD REDUCE B(NEQ) AND BACKSUBSTITUTE 15 SOLVE
C                      .FALSE. , DO NOT FORWARD REDUCE B(NEQ) OR         16 SOLVE
C                      BACKSUBSTITUTE                                    17 SOLVE
C                                                                        18 SOLVE
      IMPLICIT REAL*8(A-H,O-Z)                                           19 SOLVE
C                                                                        20 SOLVE
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                   21 SOLVE
C                                                                        22 SOLVE
      LOGICAL FACT,BACK                                                  23 SOLVE
      DIMENSION A(1),B(1),IDIAG(1)                                       24 SOLVE
C                                                                        25 SOLVE
C.... FACTOR A , REDUCE B                                                26 SOLVE
      JR = 0                                                             27 SOLVE
      DO 60 J=1,NEQ                                                      28 SOLVE
      JD = IDIAG(J)                                                      29 SOLVE
      JH = JD - JR                                                       30 SOLVE
      IS = J - JH + 2                                                    31 SOLVE
      IF(JH-2) 60,30,10                                                  32 SOLVE
   10 IF(.NOT.FACT) GO TO 50                                             33 SOLVE
      IE = J - 1                                                         34 SOLVE
      K = JR + 2                                                         35 SOLVE
      ID = IDIAG(IS - 1)                                                 36 SOLVE
C                                                                        37 SOLVE
C.... REDUCE ALL EQUATIONS EXCEPT DIAGONAL                               38 SOLVE
C                                                                        39 SOLVE
      DO 20 I=IS,IE                                                      40 SOLVE
      IR = ID                                                            41 SOLVE
      ID = IDIAG(I)                                                      42 SOLVE
      IH = MIN0(ID-IR-1,I-IS+1)                                          43 SOLVE
      IF(IH.GT.0) A(K) = A(K) - DOT(A(K-IH),A(ID-IH),IH)                 44 SOLVE
   20 K = K + 1                                                          45 SOLVE
C                                                                        46 SOLVE
C.... REDUCE DIAGONAL TERM                                               47 SOLVE
C                                                                        48 SOLVE
   30 IF(.NOT.FACT) GO TO 50                                             49 SOLVE
      IR = JR + 1                                                        50 SOLVE
      IE = JD - 1                                                        51 SOLVE
      K = J - JD                                                         52 SOLVE
      DO 40 I=IR,IE                                                      53 SOLVE
      ID = IDIAG(K+I)                                                    54 SOLVE
      IF(A(ID).EQ.0.0D0) GO TO 40                                        55 SOLVE
      D = A(I)                                                           56 SOLVE
      A(I) = A(I)/A(ID)                                                  57 SOLVE
      A(JD) = A(JD) - D*A(I)                                             58 SOLVE
   40 CONTINUE                                                           59 SOLVE
C                                                                        60 SOLVE
C.... REDUCE RHS                                                         61 SOLVE
C                                                                        62 SOLVE
   50 IF(BACK) B(J) = B(J) - DOT(A(JR+1),B(IS-1),JH-1)                   63 SOLVE
   60 JR = JD                                                            64 SOLVE
      IF(.NOT.BACK) RETURN                                               65 SOLVE
C                                                                        66 SOLVE
C.... DIVIDE BY DIAGONAL PIVOTS                                          67 SOLVE
C                                                                        68 SOLVE
      DO 70 I=1,NEQ                                                      69 SOLVE
      ID = IDIAG(I)                                                      70 SOLVE
      IF(A(ID).NE.0.0D0) B(I) = B(I)/A(ID)                               71 SOLVE
   70 CONTINUE                                                           72 SOLVE
                                                                         73 SOLVE
```

```
C                                                                       74 SOLVE
C.... BACKSUBSTITUTE                                                    75 SOLVE
C                                                                       76 SOLVE
      J = NEQ                                                           77 SOLVE
      JD = IDIAG(J)                                                     78 SOLVE
   80 D = B(J)                                                          79 SOLVE
      J = J - 1                                                         80 SOLVE
      IF(J.LE.0) RETURN                                                 81 SOLVE
      JR = IDIAG(J)                                                     82 SOLVE
      IF(JD-JR.LE.1) GO TO 100                                          83 SOLVE
      IS = J - JD + JR + 2                                              84 SOLVE
      K = JR - IS + 1                                                   85 SOLVE
      DO 90 I=IS,J                                                      86 SOLVE
   90 B(I) = B(I) - A(I+K)*D                                            87 SOLVE
  100 JD = JR                                                           88 SOLVE
      GO TO 80                                                          89 SOLVE
C                                                                       90 SOLVE
      END                                                               91 SOLVE


C                                                                        1 STACON
      SUBROUTINE STACON (S,P,N,M)                                        2 STACON
C                                                                        3 STACON
C.... PROGRAM TO STATICALLY CONDENSE SYMMETRIC POSITIVE DEFINITE MATRIX  4 STACON
C        S AND VECTOR P                                                  5 STACON
C                                                                        6 STACON
C        N= ORIGINAL NUMBER ROWS/COLUMNS IN A  AND VECTOR P              7 STACON
C        M= NUMBER OF ROWS/COLUMNS TO BE CONDENSED                       8 STACON
C                                                                        9 STACON
      IMPLICIT REAL*8(A-H,O-Z)                                          10 STACON
C                                                                       11 STACON
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                  12 STACON
C                                                                       13 STACON
      DIMENSION S(N,1),P(1)                                             14 STACON
C                                                                       15 STACON
      DO 10 I=1,M                                                       16 STACON
      II=N-I                                                            17 STACON
      JJ=II+1                                                           18 STACON
      SS=S(JJ,JJ)                                                       19 STACON
      DO 10 J=1,II                                                      20 STACON
      TT=S(J,JJ)/SS                                                     21 STACON
      P(J)=P(J)-TT*P(JJ)                                                22 STACON
      DO 10 K=1,J                                                       23 STACON
      S(K,J)=S(K,J)-S(K,JJ)*TT                                          24 STACON
   10 S(J,K)=S(K,J)                                                     25 STACON
C                                                                       26 STACON
      RETURN                                                            27 STACON
      END                                                               28 STACON


C                                                                        1 TRUSS
      SUBROUTINE TRUSS                                                   2 TRUSS
C                                                                        3 TRUSS
C.... PROGRAM TO SET UP STORAGE FOR TRUSS ELEMENTS                       4 TRUSS
C                                                                        5 TRUSS
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEQ,NF,NL,NPAR(16),NUMEQ   6 TRUSS
      COMMON/POINTS/N1,N2,N3,N4,N5,N6,N7                                 7 TRUSS
      COMMON A(1)                                                        8 TRUSS
C                                                                        9 TRUSS
      EQUIVALENCE (NPAR(2),NUMEL),(NPAR(3),NUMAT)                       10 TRUSS
C                                                                       11 TRUSS
      IF(NUMAT.EQ.0) NUMAT=1                                            12 TRUSS
      NF=N5                                                             13 TRUSS
      IF(IE.GT.1) NF=N7                                                 14 TRUSS
      N101=NF                                                           15 TRUSS
      N=NUMAT*IPREC                                                     16 TRUSS
      N102=N101+N                                                       17 TRUSS
      N103=N102+N                                                       18 TRUSS
      N104=N103+N                                                       19 TRUSS
      N105=N104+3*IPREC                                                 20 TRUSS
```

```
      N106=N105+NUMEL                                                    21 TRUSS
      N107=N106+2*NUMEL                                                  22 TRUSS
      N108=N107+6*NUMEL                                                  23 TRUSS
      NL=N108                                                            24 TRUSS
C                                                                        25 TRUSS
      IF(IE.EQ.1.AND.NL.GT.MTOT) CALL ERROR(NL-MTOT,4)                   26 TRUSS
      CALL TRUSS1(A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),                   27 TRUSS
     XA(N101),A(N102),A(N103),A(N104),A(N105),A(N106),A(N107))           28 TRUSS
C                                                                        29 TRUSS
      RETURN                                                             30 TRUSS
      END                                                                31 TRUSS


C                                                                         1 TRUSS1
      SUBROUTINE TRUSS1 (ID,X,D,IDIAG,B,A,E,AREA,WT,GRAV,                 2 TRUSS1
     X MAT,IEN,LM)                                                        3 TRUSS1
C                                                                         4 TRUSS1
C.... TRUSS ELEMENT PROGRAM                                               5 TRUSS1
C                                                                         6 TRUSS1
      IMPLICIT REAL*8 (A-H,O-Z)                                           7 TRUSS1
C                                                                         8 TRUSS1
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                    9 TRUSS1
C                                                                        10 TRUSS1
      COMMON/INFO/ICODE,MODE,NDOF,NSD,NUMNP                              11 TRUSS1
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEQ,NF,NL,NPAR(16),NUMEG   12 TRUSS1
      COMMON/DEVICE/IEL,IIN,IOUT                                         13 TRUSS1
      DIMENSION ID(NDOF,1),X(NSD,1),D(NDOF,1),IDIAG(1),B(1),A(1)         14 TRUSS1
      DIMENSION E(1),AREA(1),WT(1),GRAV(3),MAT(1),IEN(2,1),LM(3,2,1)     15 TRUSS1
      DIMENSION ST(6),S(6,6),P(6),DL(3,2),XL(3,2)                        16 TRUSS1
      EQUIVALENCE (NPAR(1),NTYPE),(NPAR(2),NUMEL),(NPAR(3),NUMAT)        17 TRUSS1
C                                                                        18 TRUSS1
      NED=3                                                              19 TRUSS1
      NEN=2                                                              20 TRUSS1
      NEE=6                                                              21 TRUSS1
      NEC=6                                                              22 TRUSS1
C                                                                        23 TRUSS1
      GOTO (1,2,3                        ),IE                            24 TRUSS1
C                                                                        25 TRUSS1
C.... READ AND PRINT ELEMENT DATA                                        26 TRUSS1
C                                                                        27 TRUSS1
    1 WRITE(IOUT,2000) NTYPE,NUMEL                                       28 TRUSS1
      WRITE(IOUT,2010) NUMAT                                             29 TRUSS1
      WRITE(IOUT,2020)                                                   30 TRUSS1
      DO 10 N=1,NUMAT                                                    31 TRUSS1
      READ(IIN,1000) N,E(N),AREA(N),WT(N)                                32 TRUSS1
   10 WRITE(IOUT,2030) N,E(N),AREA(N),WT(N)                              33 TRUSS1
      READ(IIN,1010) GRAV                                                34 TRUSS1
      WRITE(IOUT,2040) GRAV                                              35 TRUSS1
C                                                                        36 TRUSS1
      CALL ELGEN(IEN,MAT,NEN)                                            37 TRUSS1
      WRITE(IOUT,2050)                                                   38 TRUSS1
      NN=0                                                               39 TRUSS1
      DO 20 N=1,NUMEL                                                    40 TRUSS1
      NN=NN+1                                                            41 TRUSS1
      IF(NN.LE.50) GOTO 20                                               42 TRUSS1
      NN=1                                                               43 TRUSS1
      WRITE(IOUT,2050)                                                   44 TRUSS1
   20 WRITE(IOUT,2060) N,MAT(N),(IEN(I,N),I=1,NEN)                       45 TRUSS1
C                                                                        46 TRUSS1
      CALL LOCAL(ID,IEN,LM,NDOF,NED,NEN,NUMEL)                           47 TRUSS1
      CALL COLHT(IDIAG,LM,NED,NEN,NUMEL)                                 48 TRUSS1
C                                                                        49 TRUSS1
      RETURN                                                             50 TRUSS1
C                                                                        51 TRUSS1
C.... FORM STIFFNESS MATRIX AND CONSISTENT LOAD VECTOR                   52 TRUSS1
C                                                                        53 TRUSS1
    2 DO 100 N=1,NUMEL                                                   54 TRUSS1
      M=MAT(N)                                                           55 TRUSS1
      CALL LOCORD(IEN(1,N),X,XL,NEC,NEN,NSD)                             56 TRUSS1
```

```
      Y=0.00                                                            57 TRUSS1
      DO 110 I=1,3                                                      58 TRUSS1
      ST(I)=XL(I,1)-XL(I,2)                                             59 TRUSS1
  110 Y=Y+ST(I)**2                                                      60 TRUSS1
      Y=DSQRT(Y)                                                        61 TRUSS1
      DO 120 I=1,3                                                      62 TRUSS1
      ST(I)=ST(I)/Y                                                     63 TRUSS1
  120 ST(I+3)=-ST(I)                                                    64 TRUSS1
      FOR=WT(M)*Y/2.00                                                  65 TRUSS1
      DO 130 I=1,3                                                      66 TRUSS1
      P(I)=FOR*GRAV(I)                                                  67 TRUSS1
  130 P(I+3)=P(I)                                                       68 TRUSS1
      Y=E(M)*AREA(M)/Y                                                  69 TRUSS1
      DO 140 I=1,6                                                      70 TRUSS1
      Z=ST(I)*Y                                                         71 TRUSS1
      DO 140 J=1,I                                                      72 TRUSS1
      S(I,J)=ST(J)*Z                                                    73 TRUSS1
  140 S(J,I)=S(I,J)                                                     74 TRUSS1
      CALL LDISBC(DL,D,IEN(1,N),LM(1,1,N),NDOF,NEE,NED,NEN)             75 TRUSS1
      CALL DISBC(DL,P,S,NEE)                                            76 TRUSS1
  100 CALL ADDSTF(A,B,S,P,IDIAG,LM(1,1,N),NEE)                          77 TRUSS1
C                                                                       78 TRUSS1
      RETURN                                                            79 TRUSS1
C                                                                       80 TRUSS1
C.... STRESS COMPUTATION                                                81 TRUSS1
C                                                                       82 TRUSS1
    3 NN=0                                                              83 TRUSS1
      WRITE(IOUT,2070) NEG                                              84 TRUSS1
      DO 200 N=1,NUMEL                                                  85 TRUSS1
      M=MAT(N)                                                          86 TRUSS1
      CALL LCOORD(IEN(1,N),X,XL,NEC,NEN,NSD)                            87 TRUSS1
      Y=0.00                                                            88 TRUSS1
      DO 210 I=1,3                                                      89 TRUSS1
      ST(I)=XL(I,1)-XL(I,2)                                             90 TRUSS1
C                                                                       91 TRUSS1
  210 Y=Y+ST(I)**2                                                      92 TRUSS1
      Y=DSQRT(Y)                                                        93 TRUSS1
      CALL LDISP(DL,D,IEN(1,N),NDOF,NED,NEN)                            94 TRUSS1
      Z=0.00                                                            95 TRUSS1
      DO 220 I=1,3                                                      96 TRUSS1
      ST(I)=ST(I)+DL(I,1)-DL(I,2)                                       97 TRUSS1
  220 Z=Z+ST(I)**2                                                      98 TRUSS1
      Z=DSQRT(Z)                                                        99 TRUSS1
      STR=E(M)*(Z-Y)/Y                                                 100 TRUSS1
      FOR=STR*AREA(M)                                                  101 TRUSS1
      NN=NN+1                                                          102 TRUSS1
      IF(NN.LE.50) GOTO 200                                            103 TRUSS1
      NN=1                                                             104 TRUSS1
      WRITE(IOUT,2070) NEG                                             105 TRUSS1
  200 WRITE(IOUT,2080) N,STR,FOR                                       106 TRUSS1
      RETURN                                                           107 TRUSS1
C                                                                      108 TRUSS1
 1000 FORMAT(I5,5X,3F10.0)                                             109 TRUSS1
 1010 FORMAT(3F10.0)                                                   110 TRUSS1
 2000 FORMAT(1H0,                                                      111 TRUSS1
     X' T R U S S   E L E M E N T S          ',//5X,                    112 TRUSS1
     X' ELEMENT TYPE NUMBER  . . . . . (NTYPE) =',I5//5X,              113 TRUSS1
     X' NUMBER OF ELEMENTS . . . . . (NUMEL) =',I5//)                  114 TRUSS1
 2010 FORMAT(1H0,                                                      115 TRUSS1
     X' M A T E R I A L   S E T   D A T A            ',  //5X,        116 TRUSS1
     X' NUMBER OF MATERIAL SETS  . . . . . (NUMAT) =',I5//)            117 TRUSS1
 2020 FORMAT(1H0,                                                      118 TRUSS1
     X 5X,'SET',10X,'YOUNGS',10X,'CROSS-SECTION',8X,'WEIGHT/UNIT',/,   119 TRUSS1
     X 4X,'NUMBER',8X,'MODULUS',14X,'AREA',16X,'LENGTH'/)              120 TRUSS1
 2030 FORMAT(3X,I5,2X,3(1PE20.8))                                      121 TRUSS1
 2040 FORMAT(1H0,                                                      122 TRUSS1
     X' G R A V I T Y   L O A D   M U L T I P L I E R S     ',//5X,    123 TRUSS1
     X' X1-DIRECTION . . . . . =',                                     124 TRUSS1
     X1PE20.8//5X,                                                     125 TRUSS1
     X' X2-DIRECTION . . . . . =',                                     126 TRUSS1
     X1PE20.8//5X,                                                     127 TRUSS1
     X' X3-DIRECTION . . . . . =',                                     128 TRUSS1
     X1PE20.8//)                                                       129 TRUSS1
```

```
 2050 FORMAT(1H1,                                                        130 TRUSS1
     X' E L E M E N T   D A T A',//5X,                                   131 TRUSS1
     X'  ELEMENT  MATERIAL   NODE 1    NODE 2'/5X,                       132 TRUSS1
     X'   NUMBER   NUMBER'/)                                             133 TRUSS1
 2060 FORMAT(6X,I5,3(5X,I5))                                             134 TRUSS1
 2070 FORMAT(1H1,                                                        135 TRUSS1
     X' T R U S S   E L E M E N T   S T R E S S E S          ',  //5X,    136 TRUSS1
     X' ELEMENT GROUP NUMBER . . . . . . . . . . . . (NEG) =',I5//5X,    137 TRUSS1
     X'  ELEMENT',7X,'STRESS',9X,'FORCE',/5X,                            138 TRUSS1
     X'   NUMBER'//)                                                     139 TRUSS1
 2080 FORMAT(6X,I5,2(1PE20.8))                                           140 TRUSS1
      END                                                                141 TRUSS1


C                                                                         1 BEAM
      SUBROUTINE BEAM                                                     2 BEAM
C                                                                         3 BEAM
      IMPLICIT REAL*8 (A-H,O-Z)                                           4 BEAM
C                                                                         5 BEAM
      RETURN                                                              6 BEAM
      END                                                                 7 BEAM


C                                                                         1 PLANE
      SUBROUTINE PLANE                                                    2 PLANE
C                                                                         3 PLANE
C.... PROGRAM TO SET UP STORAGE FOR 4-NODE QUADRILATERAL ISOTROPIC        4 PLANE
C        PLANE STRESS/STRAIN ELEMENTS                                     5 PLANE
C                                                                         6 PLANE
      COMMON/POINTS/N1,N2,N3,N4,N5,N6,N7                                  7 PLANE
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG    8 PLANE
      COMMON A(1)                                                         9 PLANE
C                                                                        10 PLANE
      EQUIVALENCE  (NPAR(2),NUMEL),(NPAR(3),NUMAT),(NPAR(9),NUMPR)       11 PLANE
C                                                                        12 PLANE
      IF (NUMAT.EQ.0) NUMAT=1                                            13 PLANE
      NF=N5                                                              14 PLANE
      IF(IE.GT.1) NF=N7                                                  15 PLANE
      N101=NF                                                            16 PLANE
      N=NUMAT*IPREC                                                      17 PLANE
      N102=N101+N                                                        18 PLANE
      N103=N102+N                                                        19 PLANE
      N104=N103+N                                                        20 PLANE
      N105=N104+N                                                        21 PLANE
      N106=N105+2*IPREC                                                  22 PLANE
      N107=N106+NUMEL                                                    23 PLANE
      N108=N107+4*NUMEL                                                  24 PLANE
      N109=N108+8*NUMEL                                                  25 PLANE
      N110=N109+NUMPR                                                    26 PLANE
      N111=N110+NUMPR                                                    27 PLANE
      NL=N111+NUMPR*IPREC                                                28 PLANE
C                                                                        29 PLANE
      IF(IE.EQ.1 .AND.NL.GT.MTOT) CALL ERROR(NL-MTOT,4)                  30 PLANE
      CALL PLANE1(A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N101),           31 PLANE
     XA(N102),A(N103),A(N104),A(N105),A(N106) ,A(N107),A(N108),          32 PLANE
     XA(N109),A(N110),A(N111))                                           33 PLANE
C                                                                        34 PLANE
      RETURN                                                             35 PLANE
      END                                                                36 PLANE


C                                                                         1 PLANE1
      SUBROUTINE PLANE1(ID,X,D,IDIAG,B,A,E,POIS,WT,TH,GRAV,               2 PLANE1
     XMAT,IEN,LM,IELNO,ISIDE,PRES)                                        3 PLANE1
C                                                                         4 PLANE1
C.... 4-NODE QUADRILATERAL ISOTROPIC PLANE STRESS/STRAIN ELEMENT          5 PLANE1
C                                                                         6 PLANE1
C.... OPTIONS                                                             7 PLANE1
```

```
C
C       IOPT        = 0 PLANE STRAIN                                          8 PLANE1
C                   = 1 PLANE STRESS                                          9 PLANE1
C                                                                            10 PLANE1
C       IL          = 0 TWO BY TWO QUADRATURE ON LAMBDA TERM                 11 PLANE1
C                   = 1 ONE-POINT  QUADRATURE ON LAMBDA TERM                 12 PLANE1
C                                                                            13 PLANE1
C       IM          = 0 TWO BY TWO QUADRATURE ON MHU TERM                    14 PLANE1
C                   = 1 ONE-POINT  QUADRATURE ON MHU TERM                    15 PLANE1
C                                                                            16 PLANE1
C       INC         = 0 INCOMPATIBLE MODES NEGLECTED                         17 PLANE1
C                   = 1 INCOMPATIBLE MODES ADDED                             18 PLANE1
C                                                                            19 PLANE1
C       IHOUR       = 0 HOURGLASS STIFFNESS NEGLECTED                        20 PLANE1
C                   = 1 HOURGLASS STIFFNESS ADDED                            21 PLANE1
C                                                                            22 PLANE1
C.... REPITION OF LAST TWO NODES PRODUCES CONSTANT STRESS/STRAIN TRIANGLE     23 PLANE1
C                                                                            24 PLANE1
      IMPLICIT REAL*8(A-H,O-Z)                                               25 PLANE1
C                                                                            26 PLANE1
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                       27 PLANE1
C                                                                            28 PLANE1
      LOGICAL QUAD                                                           29 PLANE1
      COMMON/INFO/ICODE,MODE,NDOF,NSD,NUMNP                                  30 PLANE1
      COMMON/ELDATA/IE ,IPREC,LENGTH,MAX,MTOT,NEG,NF,NL,NPAR(16),NUMEG       31 PLANE1
      COMMON/DEVICE/IEL,IIN,IOUT                                             32 PLANE1
      DIMENSION ID(NDOF,1),X(NSD,1),D(NDOF,1),IDIAG(1),B(1),A(1)             33 PLANE1
      DIMENSION E(1),POIS(1),WT(1),TH(1),GRAV(2),MAT(1),                     34 PLANE1
     XIEN(4,1),LM(2,4,1),IELNO(1),ISIDE(1),PRES(1)                           35 PLANE1
      DIMENSION S(12,12),P(12),DL(2,4),XL(2,4),SG(4),TG(4),                  36 PLANE1
     XSH(3,6),EE(3),ST(3),PS(4),PE(4)                                        37 PLANE1
      DIMENSION S1(4,4),S2(2,2)                                              38 PLANE1
      DATA SG/-1.D0,1.D0,1.D0,-1.D0/                                         39 PLANE1
      DATA TG/-1.D0,-1.D0,1.D0,1.D0/                                         40 PLANE1
      EQUIVALENCE (NPAR(1),NTYPE),(NPAR(2),NUMEL),(NPAR(3),NUMAT),           41 PLANE1
     X(NPAR(4),IOPT),(NPAR(5),IL),(NPAR(6),IM),(NPAR(7),INC),                42 PLANE1
     X(NPAR(9),NUMPR),(NPAR(8),IHOUR)                                        43 PLANE1
C                                                                            44 PLANE1
      NED=2                                                                  45 PLANE1
      NEN=4                                                                  46 PLANE1
      NEE=8                                                                  47 PLANE1
      NEC=8                                                                  48 PLANE1
      NEN1=NEN                                                               49 PLANE1
      NEE1=NEE                                                               50 PLANE1
      IF(INC.EQ.0) GOTO 5                                                    51 PLANE1
Changes NEN1=6                                                               52 PLANE1
      NEE1=12                                                                53 PLANE1
    5 CONTINUE                                                               54 PLANE1
C                                                                            55 PLANE1
                                                                             56 PLANE1

      GOTO (1,2,3                                          ),IE              57 PLANE1
C                                                                            58 PLANE1
C.... READ AND PRINT ELEMENT DATA                                            59 PLANE1
C                                                                            60 PLANE1
    1 WRITE(IOUT,2000) NTYPE,NUMEL,IOPT,IL,IM,INC,IHOUR,NUMPR                61 PLANE1
      WRITE(IOUT,2010) NUMAT                                                 62 PLANE1
      WRITE(IOUT,2020)                                                       63 PLANE1
      DO 10 N=1,NUMAT                                                        64 PLANE1
      READ(IIN,1000) N,E(N),POIS(N),WT(N),TH(N)                              65 PLANE1
      IF(TH(N).EQ.0.D0) TH(N)=1.D0                                           66 PLANE1
   10 WRITE(IOUT,2030) N,E(N),POIS(N),WT(N),TH(N)                            67 PLANE1
      READ(IIN,1010) GRAV                                                    68 PLANE1
      WRITE(IOUT,2040) GRAV                                                  69 PLANE1
C                                                                            70 PLANE1
      CALL ELGEN(IIN,MAT,NEN) ~ should material no. still check              71 PLANE1
      WRITE(IOUT,2050)                                                       72 PLANE1
      NN=0                                                                   73 PLANE1
      DO 20 N=1,NUMEL                                                        74 PLANE1
      NN=NN+1                                                                75 PLANE1
      IF(NN.LE.50) GOTO 20                                                   76 PLANE1
```

```
      NN=1                                                              77 PLANE1
      WRITE(IOUT,2050)                                                  78 PLANE1
   20 WRITE(IOUT,2060) N,MAT(N),(IEN(I,N),I=1,NEN)                      79 PLANE1
C                                                                       80 PLANE1
      CALL LOCAL(ID,IEN,LM,NDOF,NED,NEN,NUMEL)   localise the ID array in LM   81 PLANE1
      CALL COLHT(IDIAG,LM,NED,NEN ,NUMEL)        ie put global eqn no. of each element   82 PLANE1
C                                                  node into LM           83 PLANE1
      IF(NUMPR.EQ.0) RETURN                                             84 PLANE1
      CALL PREAD(IELNO,ISIDE,PRES,1)                                    85 PLANE1
      WRITE(IOUT,2070)                                                  86 PLANE1
      NN=0                                                              87 PLANE1
      DO 30 N=1,NUMPR                                                   88 PLANE1
      NN=NN+1                                                           89 PLANE1
      IF(NN.LE.50) GOTO 30                                              90 PLANE1
      NN=1                                                              91 PLANE1
      WRITE(IOUT,2070)                                                  92 PLANE1
   30 WRITE(IOUT,2080) IELNO(N),ISIDE(N),PRES(N)                        93 PLANE1
C                                                                       94 PLANE1
      RETURN                                                            95 PLANE1
C                                                                       96 PLANE1
C.... FORM STIFFNESS MATRIX AND GRAVITY LOAD VECTOR                     97 PLANE1
C                                                                       98 PLANE1
    2 DO 100 N=1,NUMEL                                                  99 PLANE1
      M=MAT(N)                                                         100 PLANE1
      QUAD=.TRUE.                                                      101 PLANE1
      IF(IEN(3,N).EQ.IEN(4,N)) QUAD=.FALSE.   Check for △              102 PLANE1
      CALL LOCORD(IEN(1,N),X,XL,NEC,NEN,NSD)  - local/x coord of element N. in XL (NSD, NEN)  103 PLANE1
      CALL CLEAR(S,144)                                                104 PLANE1
      CALL CLEAR(P,12)                                                 105 PLANE1
C                                                                      106 PLANE1
C.... SET MATERIAL CONSTANTS                                           107 PLANE1
C                                                                      108 PLANE1
      AM=E(M)/(1.00+POIS(M))                                           109 PLANE1
      AL=AM*POIS(M)/(1.00-2.00*POIS(M))                                110 PLANE1
      AM=0.500*AM                                                      111 PLANE1
      IF(IOPT.EQ.1) AL=2.00*AL*AM/(AL+2.00*AM)                         112 PLANE1
C
C.... LAMBDA TERM CONTRIBUTION TO STIFFNESS (GRAVITY LOAD COMPUTED IN  113 PLANE1
C        THIS SEGMENT)                                                 114 PLANE1
C                                                                      115 PLANE1
      NINT=4                                                           116 PLANE1
      W=1.00                                                           117 PLANE1
      G=1.00/DSQRT(3.00)        } 2x2 quad                             118 PLANE1
      IF(IL.EQ.0) GOTO 110                                             119 PLANE1
      NINT=1                                                           120 PLANE1
      W=4.00                    } 1x1 quad.                            121 PLANE1
      G=0.00                                                           122 PLANE1
  110 DO 120 L=1,NINT                                                  123 PLANE1
      CALL SHAP2D(SG(L)*G,TG(L)*G,XL,DET,SH,NEN,INC,IEN(1,N),QUAD)     124 PLANE1
      C =DET*W*AL*TH(M)                                                125 PLANE1
      GRAV1=DET*W*AM*TH(M)                                             126 PLANE1
      GRAV2=GRAV1*GRAV(2)                                              127 PLANE1
      GRAV1=GRAV1*GRAV(1)                                              128 PLANE1
      DO 120 J=1,NEN1                                                  129 PLANE1
      DB11=C *SH(1,J)                                                  130 PLANE1
      DB12=C *SH(2,J)                                                  131 PLANE1
      IF(J.LE.4) P(2*J-1)=P(2*J-1)+GRAV1*SH(3,J)                       132 PLANE1
      IF(J.LE.4) P(2*J  )=P(2*J  )+GRAV2*SH(3,J)                       133 PLANE1
      DO 120 I=1,J                                                     134 PLANE1
      S(2*I-1,2*J-1)=S(2*I-1,2*J-1)+SH(1,I)*DB11                       135 PLANE1
      S(2*I-1,2*J  )=S(2*I-1,2*J  )+SH(1,I)*DB12                       136 PLANE1
      S(2*I  ,2*J-1)=S(2*I  ,2*J-1)+SH(2,I)*DB11                       137 PLANE1
  120 S(2*I  ,2*J  )=S(2*I  ,2*J  )+SH(2,I)*DB12                       138 PLANE1
C                                                                      139 PLANE1
C.... MAU CONTRIBUTION TO STIFFNESS                                    140 PLANE1
C                                                                      141 PLANE1
      NINT=4                                                           142 PLANE1
      W=1.00                                                           143 PLANE1
      G=1.00/DSQRT(3.00)        } 2x2 quad                             144 PLANE1
                                                                       145 PLANE1
```

```
      IF(IM.EQ.0) GOTO 130
      NINT=1
      W=4.00                    ] 1x1 quad
      G=0.00
  130 DO 140 L=1,NINT
      CALL SHAP2D (SG(L)*G,TG(L)*G,XL,DET,SH,NEN,INC,IEN(1,N),QUAD)
      C =DET*W*AM*TH(M)
      C2=2.00*C
      DO 140 J=1,NEN1
      DB11= C2*SH(1,J)
      DB22= C2*SH(2,J)
      DB31= C *SH(2,J)
      DB32= C *SH(1,J)
      DO 140 I=1,J
      S(2*I-1,2*J-1)=S(2*I-1,2*J-1)+SH(1,I)*DB11+SH(2,I)*DB31
      S(2*I-1,2*J  )=S(2*I-1,2*J  )+SH(2,I)*DB32
      S(2*I  ,2*J-1)=S(2*I  ,2*J-1)+SH(1,I)*DB31
  140 S(2*I  ,2*J  )=S(2*I  ,2*J  )+SH(2,I)*DB22+SH(1,I)*DB32
C
      IF(IHGOR.EQ.0.OR..NOT.QUAD) GO TO 149   no hourglass / no triangle go to 149
      CALL FORMS1(XL,S1)
      CALL FORMS2(XL,DET,S2)                  hourglass or triangle
      IF(IOPT .EQ. 0) C=4.000*AM*(AM+AL)/(AL+2.000*AM)
      IF(IOPT .EQ. 1) C=E(M)
      C=C/(12.00*DET)
      S2(1,1)=C*S2(1,1)
      S2(1,2)=C*S2(1,2)
      S2(2,2)=C*S2(2,2)
      DO 145 J=1,NEN
      DO 145 I=1,J
      C=S1(I,J)*S2(1,2)
      S(2*I-1,2*J-1)=S(2*I-1,2*J-1)+S1(I,J)*S2(1,1)
      S(2*I-1,2*J  )=S(2*I-1,2*J  )+C
      S(2*I  ,2*J-1)=S(2*I  ,2*J-1)+C
  145 S(2*I  ,2*J  )=S(2*I  ,2*J  )+S1(I,J)*S2(2,2)
C
C.... SYMMETRIZE THE STIFFNESS
C
  149 DO 150 I=2,NEE1
      DO 150 J=1,I
  150 S(I,J)=S(J,1)
      IF(INC.EQ.1.AND.QUAD) CALL STACON(S,P,12,4)
      CALL PACK(S,8,12)
      CALL LOISBC(DL,D,IEN(1,N),LM(1,1,N),NDOF,NEE,NED,NEN)
      CALL DISBC(DL,P,S,NEE)
  100 CALL ADDSTF(A,B,S,P,IDIAG,LM(1,1,N),NEE)
C
      IF(NUMPR.EQ.0) RETURN
C
C.... PRESSURE LOADS
C
      DO 160 K=1,NUMPR
      N=IELNO(K)  -- element no.
      M=MAT(N)    - get material
      CALL LCOORD(IEN(1,N),X,XL,NEC,NEN,NSD)   localize coord of element N
      I=ISIDE(K)                               IEN holds global node no of node I of element N
      J=I+1                                    I= side # of element
      IF(J.EQ.5) J=1
      EE(1)=(XL(2,I)-XL(2,J))*PRES(K)*TH(M)/2.00   1/2 force/side in x dir
      EE(2)=(XL(1,J)-XL(1,I))*PRES(K)*TH(M)/2.00   1/2 force/side in y dir
      DO 170 L=1,2
      II=LM(L,I,N)  - get eqn no. of def L, node I of element N
      JJ=LM(L,J,N)  -  "   "   "  of def L, node I+1 of element N
  IF(II.NE.0) B(II)=B(II)+EE(L)   } assign 1/2 force to each node
  IF(JJ.NE.0) B(JJ)=B(JJ)+EE(L)   }
  170 CONTINUE
  160 CONTINUE
      RETURN
```

```
C
C.... STRESS COMPUTATIONS (AT CENTROID)
C
    3 NN=0
      WRITE(IOUT,2090) NEG
      DO 200 N=1,NUMEL
      M=MAT(N)
      CALL LCOORD(IEN(1,N),X,XL,NEC,NEN,NSD)
      CALL LOISP(DL,D,IEN(1,N),NDOF,NED,NEN)
      QUAD=.TRUE.
      IF(IEN(3,N).EQ.IEN(4,N)) QUAD=.FALSE.
      CALL SHAP2D(0.D0,0.D0,XL,DET,SH,NEN,0,IEN(1,N),QUAD)
      EE(1)=0.D0
      EE(2)=0.D0
      EE(3)=0.D0
      XX=0.D0
      YY=0.D0
      AM=E(M)/(1.D0+POIS(M))
      AL=AM*POIS(M)/(1.D0-2.D0*POIS(M))
      AM=0.5D0*AM
      IF(IOPT.EQ.1) AL=2.D0*AL*AM/(AL+2.D0*AM)
      AL2=AL+2.D0*AM
      DO 210 I=1,NEN
      EE(1)=EE(1)+SH(1,I)*DL(1,I)
      EE(2)=EE(2)+SH(2,I)*DL(1,I)+SH(1,I)*DL(2,I)
      EE(3)=EE(3)+SH(2,I)*DL(2,I)
      XX=XX+SH(3,I)*XL(1,I)
  210 YY=YY+SH(3,I)*XL(2,I)
      ST(1)=AL2*EE(1)+AL*EE(3)
      ST(2)=AM*EE(2)
      ST(3)=AL*EE(1)+AL2*EE(3)
      EE(2)=0.5D0*EE(2)
      CALL PRINC(NSD,ST,PS)
      CALL PRINC(NSD,EE,PE)
      EE(2)=2.D0*EE(2)
      NN=NN+1
      IF(NN.LE.5) GOTO 200
      NN=1
      WRITE(IOUT,2090) NEG
  200 WRITE(IOUT,2100) N,M,XX,YY,ST,EE,PS,PE
C
      RETURN
 1000 FORMAT(I5,5X,4F10.0)
 1010 FORMAT(2F10.0)
 2000 FORMAT(1H0,
     X' P L A N E   S T R E S S / S T R A I N   E L E M E N T S'//5X,
     X' ELEMENT TYPE NUMBER . . . . . (NTYPE) =',I5//5X,
     X' NUMBER OF ELEMENTS  . . . . . (NUMEL) =',I5//5X,
     X' ANALYSIS OPTION . . . . . (IOPT ) =',I5//5X,
     X'     EQ.0, PLANE STRAIN                ',/5X ,
     X'     EQ.1, PLANE STRESS                ',//5X,
     X' LAMBDA TERM INTEGRATION CODE . . . . . (IL   ) =',I5/ 5X,
     X'     EQ.0, 2X2 GAUSSIAN QUADRATURE     ',  / 5X,
     X'     EQ.1, 1-POINT GAUSSIAN QUADRATURE ',  //5X,
     X' MHU TERM INTEGRATION CODE . . . . . (IM   ) =',I5/ 5X,
     X'     EQ.0, 2X2 GUASSIAN QUADRATURE     ',  / 5X,
     X'     EQ.1, 1-POINT GAUSSIAN QUADRATURE ',  //5X,
     X' INCOMPATIBLE MODES CODE . . . . . (INC  ) =',I5/ 5X
     X'     EQ.0, INCOMPATIBLE MODES NEGLECTED',  / 5X,
     X'     EQ.1, INCOMPATIBLE MODES ADDED    ',  //5X,
     X' HOURGLASS STIFFNESS CODE. . . . . (IHOUR) =',I5/ 5X,
     X'     EQ.0, HOURGLASS STIFFNESS NEGLECTED',  / 5X,
     X'     EQ.1, HOURGLASS STIFFNESS ADDED    ',  //5X,
     X' NUMBER OF ELEMENT PRESSURE LOAD CARDS . . . (NUMPR) =',I5//)
2010 FORMAT(1H1,
     X' M A T E R I A L   S E T   D A T A                    ',  //5X,
     X' NUMBER OF MATERIAL SETS . . . . . (NUMAT) =',I5//)
2020 FORMAT(1HO,
     X 5X,'SET',11X,'YOUNGS',13X,'POISSONS',10X,'WEIGHT/UNIT',10X,
     X'THICKNESS',11X,/
     X  4X,'NUMBER',9X,'MODULUS',14X,'RATIO',14X,'VOLUME',33X,/)
2030 FORMAT(3X,I5,2X,4(1PE20.8))
```

$u_1 = \Sigma d_{a1} N_a \quad u_2 = \Sigma d_{a2} N_a \quad \varepsilon_{xy} = \Sigma d_{a1} N_{a,y} + d_{a2} N_{a,x}$

$\varepsilon_{xx} = \sum_{a=1}^{4} d_{a1} N_{a,y} + d_{a2} N_{a,x}$

```
 2040 FORMAT(1HO,                                                        286 PLANE1
     X' G R A V I T Y   L O A D   M U L T I P L I E R S       ',//5X,   287 PLANE1
     X' X1-DIRECTION  . . . . . . . . . . . . . . . . . . =',1PE20.8//5X,  288 PLANE1
     X' X2-DIRECTION  . . . . . . . . . . . . . . . . . . =',1PE20.8//)   289 PLANE1
 2050 FORMAT(1H1,                                                        290 PLANE1
     X' E L E M E N T   D A T A',//5X,                                   291 PLANE1
     X' ELEMENT  MATERIAL   NODE 1    NODE 2    NODE 3    NODE 4'/5X,   292 PLANE1
     X'  NUMBER   NUMBER'/)                                              293 PLANE1
 2060 FORMAT(6X,I5,5(5X,I5))                                             294 PLANE1
 2070 FORMAT(1H1,                                                        295 PLANE1
     X' P R E S S U R E   L O A D   D A T A',//5X,                       296 PLANE1
     X' ELEMENT    SIDE',10X,'PRESSURE',//5X,                            297 PLANE1
     X'  NUMBER   NUMBER',/)                                             298 PLANE1
 2080 FORMAT(6X,I5,4X,I5,1PE20.8)                                        299 PLANE1
 2090 FORMAT(1H1,                                                        300 PLANE1
     X' P L A N E   E L E M E N T   S T R E S S E S         ',  //5X,    301 PLANE1
     X' ELEMENT GROUP NUMBER                                  ',I5//)    302 PLANE1
 2100 FORMAT(1HO,                                                        303 PLANE1
     X' ELEMENT    MATERIAL   COORDINATE  COORDINATE    STRESS      STRE 304 PLANE1
     XSS      STRESS',3(6X,'STRAIN'),/' ',2X,'NUMBER',5X,                305 PLANE1
     X'NUMBER',8X,'X1', 10X,'X2',10X,'11',10X,'12',10X,'22',10X, '11',  306 PLANE1
     X 10X,'12',10X,'22'//2X,I5,6X,I5,4X,1P8E12.3,//,24X,                307 PLANE1
     X' STRESS 1    STRESS 2      ANGLE       SHEAR     STRAIN 1',3X,    308 PLANE1
     X' STRAIN 2      ANGLE       SHEAR',//,22X,1P8E12.3,/)              309 PLANE1
C                                                                        310 PLANE1
      END                                                                311 PLANE1


C                                                                         1 FORMS1
      SUBROUTINE FORMS1(X,S1)                                             2 FORMS1
C                                                                         3 FORMS1
C.... PROGRAM TO COMPUTE ARRAY S1 FOR HOURGLASS STIFFNESS FORMULATION     4 FORMS1
C                                                                         5 FORMS1
      IMPLICIT REAL*8 (A-H,O-Z)                                           6 FORMS1
C                                                                         7 FORMS1
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                    8 FORMS1
C                                                                         9 FORMS1
      DIMENSION X(2,1),S1(4,1),A(4)                                      10 FORMS1
C                                                                        11 FORMS1
      T23=X(1,2)*X(2,3)-X(1,3)*X(2,2)                                    12 FORMS1
      T13=X(1,1)*X(2,3)-X(1,3)*X(2,1)                                    13 FORMS1
      T12=X(1,1)*X(2,2)-X(1,2)*X(2,1)                                    14 FORMS1
      T43=X(1,4)*X(2,3)-X(1,3)*X(2,4)                                    15 FORMS1
      T42=X(1,4)*X(2,2)-X(1,2)*X(2,4)                                    16 FORMS1
      T14=X(1,1)*X(2,4)-X(1,4)*X(2,1)                                    17 FORMS1
C                                                                        18 FORMS1
      TEMP=T23-T13+T12                                                   19 FORMS1
C                                                                        20 FORMS1
      A(1)=(-T23+T43-T42)/TEMP                                           21 FORMS1
      A(2)=(-T43+T13-T14)/TEMP                                           22 FORMS1
      A(3)=-(1.000+A(1)+A(2))                                            23 FORMS1
      A(4)=1.000                                                         24 FORMS1
C                                                                        25 FORMS1
      TEMP=((A(1)**2+A(2)**2+A(3)**2+1.000)**0.500)/2.000                26 FORMS1
C                                                                        27 FORMS1
      DO 10 I=1,4                                                        28 FORMS1
   10 A(I)=A(I)/TEMP                                                     29 FORMS1
C                                                                        30 FORMS1
      DO 20 J=1,4                                                        31 FORMS1
      DO 20 I=1,J                                                        32 FORMS1
      S1(I,J)=A(I)*A(J)                                                  33 FORMS1
   20 S1(J,I)=S1(I,J)                                                    34 FORMS1
C                                                                        35 FORMS1
      RETURN                                                             36 FORMS1
      END                                                                37 FORMS1
```

```
C
      SUBROUTINE FORMS2(X,DET,S2)                                          1 FORMS2
C                                                                          2 FORMS2
C.... PROGRAM TO COMPUTE ARRAY S2 FOR HOURGLASS STIFFNESS FORMULATION      3 FORMS2
C                                                                          4 FORMS2
      IMPLICIT REAL*8 (A-H,O-Z)                                            5 FORMS2
C                                                                          6 FORMS2
C.... REMOVE ABOVE CARD FOR SINGLE PRECISION OPERATION                     7 FORMS2
C                                                                          8 FORMS2
      DIMENSION X(2,1),S2(2,1),XS(2,2),A(2,4)                              9 FORMS2
      DATA A/-0.500,-0.500,0.500,-0.500,0.500,0.500,-0.500,0.500/         10 FORMS2
C                                                                         11 FORMS2
      DO 10 I=1,2                                                         12 FORMS2
      DO 10 J=1,2                                                         13 FORMS2
      XS(I,J)=0.000                                                       14 FORMS2
      DO 10 K=1,4                                                         15 FORMS2
   10 XS(I,J)=XS(I,J)+X(I,K)*A(J,K)                                       16 FORMS2
C                                                                         17 FORMS2
      DET=XS(1,1)*XS(2,2)-XS(1,2)*XS(2,1)                                 18 FORMS2
C                                                                         19 FORMS2
      S2(1,1)=XS(2,1)**2+XS(2,2)**2                                       20 FORMS2
      S2(1,2)=-XS(1,1)*XS(2,1)-XS(1,2)*XS(2,2)                            21 FORMS2
      S2(2,1)=S2(1,2)                                                     22 FORMS2
      S2(2,2)=XS(1,1)**2+XS(1,2)**2                                       23 FORMS2
C                                                                         24 FORMS2
      RETURN                                                              25 FORMS2
      END                                                                 26 FORMS2
                                                                          27 FORMS2
```

Solved Problems

TRUSS PROBLEM

C O N T R O L I N F O R M A T I O N

NUMBER OF NODAL POINTS . . . . . (NUMNP) = 27

NUMBER OF SPACE DIMENSIONS . . . . . (NSD) = 3

NUMBER OF DEGREES OF FREEDOM PER NODE . . . (NDOF) = 3

NUMBER OF ELEMENT GROUPS . . . . . (NUMEG) = 1

ANALYSIS CODE . . . . . (ICODE) = 0
EQ.0, STATIC ANALYSIS
EQ.1, RANK CHECK

SOLUTION MODE . . . . . (MODE) = 1
EQ.0, DATA CHECK
EQ.1, EXECUTION

N O D A L C O O R D I N A T E D A T A

NODE NUMBER	X1	X2	X3
1	0.0	0.0	0.0
2	6.000000000D 01	0.0	0.0
3	1.200000000D 02	0.0	0.0
4	1.800000000D 02	0.0	0.0
5	2.400000000D 02	0.0	0.0
6	3.000000000D 02	0.0	0.0
7	3.600000000D 02	0.0	0.0
8	4.200000000D 02	0.0	0.0
9	4.800000000D 02	0.0	0.0
10	5.400000000D 02	0.0	0.0
11	6.000000000D 02	0.0	0.0
12	6.600000000D 02	0.0	0.0
13	7.200000000D 02	0.0	0.0
14	7.800000000D 02	0.0	0.0
15	3.000000000D 01	6.000000000D 01	0.0
16	9.000000000D 01	6.000000000D 01	0.0
17	1.500000000D 02	6.000000000D 01	0.0
18	2.100000000D 02	6.000000000D 01	0.0
19	2.700000000D 02	6.000000000D 01	0.0
20	3.300000000D 02	6.000000000D 01	0.0
21	3.900000000D 02	6.000000000D 01	0.0
22	4.500000000D 02	6.000000000D 01	0.0
23	5.100000000D 02	6.000000000D 01	0.0
24	5.700000000D 02	6.000000000D 01	0.0
25	6.300000000D 02	6.000000000D 01	0.0
26	6.900000000D 02	6.000000000D 01	0.0
27	7.500000000D 02	6.000000000D 01	0.0

N O D A L B O U N D A R Y C O N D I T I O N C O D E S

NODE NUMBER	DOF1	DOF2	DOF3
1	1	1	1
2	0	0	1
3	0	0	1
4	0	0	1
5	0	0	1
6	0	0	1
7	0	0	1
8	0	0	1
9	0	0	1
10	0	0	1
11	0	0	1
12	0	0	1
13	0	0	1
14	1	1	1
15	0	0	1
16	0	0	1
17	0	0	1
18	0	0	1
19	0	0	1
20	0	0	1
21	0	0	1
22	0	0	1
23	0	0	1
24	0	0	1
25	0	0	1
26	0	0	1
27	0	0	1

N O D A L F O R C E S A N D D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2	DOF3
21	0.0	-1.000000000 03	0.0

E L E M E N T G R O U P D A T A

ELEMENT GROUP NUMBER . . . (NEG) = 1

T R U S S E L E M E N T S

ELEMENT TYPE NUMBER . . . . . (NTYPE) = 1

NUMBER OF ELEMENTS . . . . . (NUMEL) = 51

M A T E R I A L S E T D A T A

NUMBER OF MATERIAL SETS . . . . . (NUMAT) = 1

SET NUMBER	YOUNGS MODULUS	CROSS-SECTION AREA	WEIGHT/UNIT LENGTH
1	3.000000000 07	1.000000000 01	0.0

G R A V I T Y L O A D M U L T I P L I E R S

X1-DIRECTION . . . . . = 0.0

X2-DIRECTION . . . . . = 0.0

X3-DIRECTION . . . . . = 0.0

E L E M E N T D A T A

ELEMENT NUMBER	MATERIAL NUMBER	NODE 1	NODE 2
1	1	1	2
2	1	2	3
3	1	3	4
4	1	4	5
5	1	5	6
6	1	6	7
7	1	7	8
8	1	8	9
9	1	9	10
10	1	10	11
11	1	11	12
12	1	12	13
13	1	13	14
14	1	15	16
15	1	16	17
16	1	17	18
17	1	18	19
18	1	19	20
19	1	20	21
20	1	21	22
21	1	22	23
22	1	23	24
23	1	24	25
24	1	25	26
25	1	26	27
26	1	1	15
27	1	2	16
28	1	3	17
29	1	4	18
30	1	5	19

31	1	6	20
32	1	7	21
33	1	8	22
34	1	9	23
35	1	10	24
36	1	11	25
37	1	12	26
38	1	13	27
39	1	2	15
40	1	3	16
41	1	4	17
42	1	5	18
43	1	6	19
44	1	7	20
45	1	8	21
46	1	9	22
47	1	10	23
48	1	11	24
49	1	12	25
50	1	13	26

E L E M E N T D A T A

ELEMENT NUMBER	MATERIAL NUMBER	NODE 1	NODE 2
51	1	14	27

TRUSS PROBLEM

E Q U A T I O N S Y S T E M D A T A

NUMBER OF EQUATIONS . . . . . (NEQ) = 50

NUMBER OF TERMS IN STIFFNESS . . . . . (NA) = 791

MEAN HALF BANDWIDTH . . . . . (MB) = 15

TOTAL LENGTH OF BLANK COMMON REQUIRED . . . . (NB) = 2609

D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2	DOF3
1	0.0	0.0	0.0
2	-2.76923077D-04	-2.51797004D-03	0.0
3	-4.53846154D-04	-5.16286315D-03	0.0
4	-5.30769231D-04	-7.73467934D-03	0.0
5	-5.07692308D-04	-1.00334186D-02	0.0

6	-3.846153850-04	-1.185908090-02	0.0
7	-1.615384620-04	-1.301166640-02	0.0
8	1.615384620-04	-1.301166640-02	0.0
9	3.846153850-04	-1.185908090-02	0.0
10	5.076923080-04	-1.003341860-02	0.0
11	5.307692310-04	-7.734679340-03	0.0
12	4.538461540-04	-5.162863150-03	0.0
13	2.769230770-04	-2.517970040-03	0.0
14	0.0	0.0	0.0
15	2.100000000-03	-1.189754250-03	0.0
16	2.000000000-03	-3.796185820-03	0.0
17	1.800000000-03	-6.429540470-03	0.0
18	1.500000000-03	-8.889818200-03	0.0
19	1.100000000-03	-1.097701900-02	0.0
20	6.000000000-04	-1.249114290-02	0.0
21	-2.005774020-18	-1.323218980-02	0.0
22	-6.000000000-04	-1.249114290-02	0.0
23	-1.100000000-03	-1.097701900-02	0.0
24	-1.500000000-03	-8.889818200-03	0.0
25	-1.800000000-03	-6.429540470-03	0.0
26	-2.000000000-03	-3.796185820-03	0.0
27	-2.100000000-03	-1.189754250-03	0.0

T R U S S E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER . . . . . (NEG) = 1

ELEMENT NUMBER	STRESS	FORCE
1	-1.384351210 02	-1.384351210 03
2	-8.843239060 01	-8.843239060 02
3	-3.843397910 01	-3.843397910 02
4	1.156047900 01	1.156047900 02
5	6.155234920 01	6.155234920 02
6	1.115439970 02	1.115439970 03
7	1.615384620 02	1.615384620 03
8	1.115439970 02	1.115439970 03
9	6.155234920 01	6.155234920 02
10	1.156047900 01	1.156047900 02
11	-3.843397910 01	-3.843397910 02
12	-8.843239060 01	-8.843239060 02
13	-1.384351210 02	-1.384351210 03
14	-4.997169380 01	-4.997169380 02
15	-9.997110590 01	-9.997110590 02
16	-1.499747790 02	-1.499747790 03
17	-1.999818480 02	-1.999818480 03
18	-2.499904480 02	-2.499904480 03
19	-2.999977120 02	-2.999977120 03
20	-2.999977120 02	-2.999977120 03
21	-2.499904480 02	-2.499904480 03
22	-1.999818480 02	-1.999818480 03
23	-1.499747790 02	-1.499747790 03

24	-9.997110590 01	-9.997110590 02
25	-4.997169380 01	-4.997169380 02
26	-5.588233310 01	-5.588233310 02
27	-5.587902410 01	-5.587902410 02
28	-5.587947050 01	-5.587947050 02
29	-5.588355690 01	-5.588355690 02
30	-5.589016800 01	-5.589016800 02
31	-5.589718830 01	-5.589718830 02
32	-5.590150240 01	-5.590150240 02
33	5.590448360 01	5.590448360 02
34	5.591158770 01	5.591158770 02
35	5.591944280 01	5.591944280 02
36	5.592543350 01	5.592543350 02
37	5.592794450 01	5.592794450 02
38	5.592636040 01	5.592636040 02
39	5.592636040 01	5.592636040 02
40	5.592794450 01	5.592794450 02
41	5.592543350 01	5.592543350 02
42	5.591944280 01	5.591944280 02
43	5.591158770 01	5.591158770 02
44	5.590448360 01	5.590448360 02
45	-5.590150240 01	-5.590150240 02
46	-5.589718830 01	-5.589718830 02
47	-5.589016800 01	-5.589016800 02
48	-5.588355690 01	-5.588355690 02
49	-5.587947050 01	-5.587947050 02
50	-5.587902410 01	-5.587902410 02

T R U S S E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER . . . . . (NEG) = 1

ELEMENT NUMBER	STRESS	FORCE
51	-5.588233310 01	-5.588233310 02

TRUSS PROBLEM

S O L U T I O N T I M E L O G I N S E C O N D S

INPUT PHASE . . . . . =	0.43
FORMATION OF STIFFNESS AND LOAD . . . . . =	0.28
FACTOR STIFFNESS . . . . . =	0.23
DISPLACEMENT AND STRESS ANALYSIS . . . . . =	0.47
T O T A L S O L U T I O N T I M E . . . . . =	1.42

PLANE STRAIN P.R.=0.300 1X1 MHU TERM , 1X1 LAMBDA TERM , HOURGLASS ADDED

C O N T R O L I N F O R M A T I O N

NUMBER OF NODAL POINTS . . . . . (NUMNP) = 45

NUMBER OF SPACE DIMENSIONS . . . . . (NSD) = 2

NUMBER OF DEGREES OF FREEDOM PER NODE . . . (NDOF) = 2

NUMBER OF ELEMENT GROUPS . . . . . (NUMEG) = 1

ANALYSIS CODE . . . . . (ICODE) = 0
EQ.0, STATIC ANALYSIS
EQ.1, RANK CHECK

SOLUTION MODE . . . . . (MODE) = 1
EQ.0, DATA CHECK
EQ.1, EXECUTION

N O D A L C O O R D I N A T E D A T A

NODE NUMBER	X1	X2
1	0.0	0.0
2	2.000000000D 00	0.0
3	4.000000000D 00	0.0
4	6.000000000D 00	0.0
5	8.000000000D 00	0.0
6	1.000000000D 01	0.0
7	1.200000000D 01	0.0
8	1.400000000D 01	0.0
9	1.600000000D 01	0.0
10	0.0	5.000000000D-01
11	2.000000000D 00	5.000000000D-01
12	4.000000000D 00	5.000000000D-01
13	6.000000000D 00	5.000000000D-01
14	8.000000000D 00	5.000000000D-01
15	1.000000000D 01	5.000000000D-01
16	1.200000000D 01	5.000000000D-01
17	1.400000000D 01	5.000000000D-01
18	1.600000000D 01	5.000000000D-01
19	0.0	1.000000000D 00
20	2.000000000D 00	1.000000000D 00
21	4.000000000D 00	1.000000000D 00
22	6.000000000D 00	1.000000000D 00
23	8.000000000D 00	1.000000000D 00
24	1.000000000D 01	1.000000000D 00
25	1.200000000D 01	1.000000000D 00
26	1.400000000D 01	1.000000000D 00
27	1.600000000D 01	1.000000000D 00
28	0.0	1.500000000D 00
29	2.000000000D 00	1.500000000D 00
30	4.000000000D 00	1.500000000D 00
31	6.000000000D 00	1.500000000D 00
32	8.000000000D 00	1.500000000D 00
33	1.000000000D 01	1.500000000D 00
34	1.200000000D 01	1.500000000D 00
35	1.400000000D 01	1.500000000D 00
36	1.600000000D 01	1.500000000D 00
37	0.0	2.000000000D 00
38	2.000000000D 00	2.000000000D 00
39	4.000000000D 00	2.000000000D 00
40	6.000000000D 00	2.000000000D 00
41	8.000000000D 00	2.000000000D 00
42	1.000000000D 01	2.000000000D 00
43	1.200000000D 01	2.000000000D 00
44	1.400000000D 01	2.000000000D 00
45	1.600000000D 01	2.000000000D 00

N O D A L B O U N D A R Y C O N D I T I O N C O D E S

NODE NUMBER	DOF1	DOF2
1	1	1
2	1	0
3	1	0
4	1	0
5	1	0
6	1	0
7	1	0
8	1	0
9	1	0
37	1	0

N O D A L F O R C E S A N D D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2
9	0.0	-9.280000000D-02
10	-7.500000000D-01	1.738300000D-01
18	0.0	-1.738300000D-01
19	-1.500000000D 00	1.386700000D-01
27	0.0	-1.386700000D-01
28	-2.250000000D 00	8.008000000D-02
36	0.0	-8.008000000D-02
37	0.0	1.465000000D-02
45	0.0	-1.465000000D-02

E L E M E N T G R O U P D A T A

ELEMENT GROUP NUMBER . . . (NEG) = 1

P L A N E S T R E S S / S T R A I N E L E M E N T S

ELEMENT TYPE NUMBER . . . . . (NTYPE) = 3

NUMBER OF ELEMENTS . . . . . (NUMEL) = 32

ANALYSIS OPTION . . . . . (IOPT) = 0

EQ.0, PLANE STRAIN
EQ.1, PLANE STRESS

LAMBDA TERM INTEGRATION CODE . . . . . (IL) = 1
EQ.0, 2X2 GAUSSIAN QUADRATURE
EQ.1, 1-POINT GAUSSIAN QUADRATURE

MHU TERM INTEGRATION CODE . . . . . (IM) = 1
EQ.0, 2X2 GUASSIAN QUADRATURE
EQ.1, 1-POINT GAUSSIAN QUADRATURE

INCOMPATIBLE MODES CODE . . . . . (INC) = 0
EQ.0, INCOMPATIBLE MODES NEGLECTED
EQ.1, INCOMPATIBLE MODES ADDED

HOURGLASS STIFFNESS CODE. . . . . (IHOUR) = 1
EQ.0, HOURGLASS STIFFNESS NEGLECTED
EQ.1, HOURGLASS STIFFNESS ADDED

NUMBER OF ELEMENT PRESSURE LOAD CARDS . . . (NUMPR) = 0

M A T E R I A L S E T D A T A

NUMBER OF MATERIAL SETS . . . . . (NUMAT) = 1

SET NUMBER	YOUNGS MODULUS	POISSONS RATIO	WEIGHT/UNIT VOLUME	THICKNESS
1	1.000000000 00	3.000000000-01	0.0	1.000000000 00

G R A V I T Y L O A D M U L T I P L I E R S

X1-DIRECTION . . . . . = 0.0

X2-DIRECTION . . . . . = 0.0

E L E M E N T D A T A

ELEMENT NUMBER	MATERIAL NUMBER	NODE 1	NODE 2	NODE 3	NODE 4
1	1	1	2	11	10
2	1	2	3	12	11
3	1	3	4	13	12
4	1	4	5	14	13
5	1	5	6	15	14
6	1	6	7	16	15
7	1	7	8	17	16
8	1	8	9	18	17
9	1	10	11	20	19
10	1	11	12	21	20
11	1	12	13	22	21
12	1	13	14	23	22
13	1	14	15	24	23
14	1	15	16	25	24
15	1	16	17	26	25
16	1	17	18	27	26
17	1	19	20	29	28
18	1	20	21	30	29
19	1	21	22	31	30
20	1	22	23	32	31
21	1	23	24	33	32
22	1	24	25	34	33
23	1	25	26	35	34
24	1	26	27	36	35
25	1	28	29	38	37
26	1	29	30	39	38
27	1	30	31	40	39
28	1	31	32	41	40
29	1	32	33	42	41
30	1	33	34	43	42
31	1	34	35	44	43
32	1	35	36	45	44

PLANE STRAIN P.R.=0.300 1X1 MHU TERM , 1X1 LAMBDA TERM , HOURGLASS ADDED

E Q U A T I O N S Y S T E M D A T A

NUMBER OF EQUATIONS . . . . . (NEQ) = 79

NUMBER OF TERMS IN STIFFNESS . . . . . (NA) = 1401

MEAN HALF BANDWIDTH . . . . . (MB) = 17

TOTAL LENGTH OF BLANK COMMON REQUIRED . . . . (NB) = 3918

D I S P L A C E M E N T S

NODE NUMBER	DOF1	DOF2
1	0.0	0.0
2	0.0	-6.47465382D 00
3	0.0	-2.252313030 01
4	0.0	-4.675671700 01
5	0.0	-7.781646640 01
6	0.0	-1.143365390 02
7	0.0	-1.549511050 02
8	0.0	-1.983012320 02
9	0.0	-2.429982530 02
10	-1.369540190-01	-1.423141900-01
11	2.419415360 00	-6.603561650 00
12	4.641170700 00	-2.263285260 01
13	6.517505300 00	-4.684804140 01
14	8.052915360 00	-7.738959550 01
15	9.247664650 00	-1.143914730 02
16	1.010151150 01	-1.545876450 02
17	1.060989880 01	-1.983185700 02
18	1.078355480 01	-2.430022060 02
19	-2.203007990-01	-5.685737110-01
20	4.891882760 00	-6.990463230 00
21	9.334055470 00	-2.296181790 01
22	1.308656920 01	-4.712209060 01
23	1.615785270 01	-7.810698260 01
24	1.854688330 01	-1.145562010 02
25	2.025473610 01	-1.550974670 02
26	2.127285910 01	-1.983704190 02
27	2.162085190 01	-2.430143460 02
28	-1.947076100-01	-1.279149380 00
29	7.470012620 00	-7.635726700 00
30	1.413018260 01	-2.350942850 01
31	1.975846000 01	-4.757915320 01
32	2.436740780 01	-7.847463370 01
33	2.794897140 01	-1.148304200 02
34	3.051111730 01	-1.552811740 02
35	3.204158010 01	-1.984563950 02
36	3.256708700 01	-2.430354850 02
37	0.0	-2.271419700 00
38	1.020510420 01	-8.539198150 00
39	1.907307640 01	-2.427535330 01
40	2.653879540 01	-4.821952140 01
41	3.272970020 01	-7.898654130 01
42	3.750943270 01	-1.152138590 02
43	4.091945080 01	-1.555390670 02
44	4.296688690 01	-1.985767190 02
45	4.368313060 01	-2.430670870 02

P L A N E E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
1	1	1.0000 00	2.5000-01	7.0390-01	-3.6600-01	3.6060-03	6.3910-01	-9.5160-01	-2.7120-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		8.6020-01	-1.5280-01	-2.3130 01	5.0650-01	8.4240-01	-4.7450-01	-2.3130 01	6.5840-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
2	1	3.0000 00	2.5000-01	6.1000-01	-3.6880-01	-7.8730-04	5.5540-01	-9.5890-01	-2.3860-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		7.8350-01	-1.7420-01	-2.5190 01	4.7880-01	7.8090-01	-4.6410-01	-2.5190 01	6.2250-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
3	1	5.0000 00	2.5000-01	5.1550-01	-3.6670-01	-1.4500-05	4.6910-01	-9.5350-01	-2.0100-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		7.0600-01	-1.9050-01	-2.7450 01	4.4820-01	7.1670-01	-4.4870-01	-2.7450 01	5.8270-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
4	1	7.0000 00	2.5000-01	4.2190-01	-3.6730-01	7.3690-05	3.8390-01	-9.5490-01	-1.6450-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		6.3450-01	-2.1260-01	-3.0070 01	4.2350-01	6.6030-01	-4.4090-01	-3.0070 01	5.3060-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
5	1	9.0000 00	2.5000-01	3.2820-01	-3.6730-01	-7.3720-05	2.9870-01	-9.5490-01	-1.2810-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		5.6630-01	-2.3820-01	-3.2960 01	4.0230-01	6.0830-01	-4.3770-01	-3.2960 01	5.2300-01

P L A N E E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
6	1	1.1000 01	2.5000-01	2.3460-01	-3.6670-01	1.2480-05	2.1350-01	-9.5350-01	-9.1470-02
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		5.0230-01	-2.6770-01	-3.6130 01	3.8500-01	5.6150-01	-4.3950-01	-3.6130 01	5.0050-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
7	1	1.3000 01	2.5000-01	1.4000-01	-3.6880-01	7.9690-04	1.2710-01	-9.5890-01	-5.3880-02
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		4.4570-01	-3.0490-01	-3.9660 01	3.7530-01	5.2450-01	-4.5130-01	-3.9660 01	4.8790-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
8	1	1.5000 01	2.5000-01	4.6160-02	-3.6600-01	-3.6140-03	4.3410-02	-9.5170-01	-2.1290-02
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.8820-01	-3.4560-01	-4.3060 01	3.6690-01	4.8800-01	-4.6590-01	-4.3060 01	4.7700-01

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
9	1	1.6000 00	7.5000-01	2.1110 00	-3.1980-01	1.0880-02	1.9170 00	-8.3150-01	-8.1360-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		2.1590 00	-3.6730-02	-8.4680 00	1.0980 00	1.9790 00	-8.7540-01	-8.4680 00	1.4270 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
10	1	3.0000 00	7.5000-01	1.8300 00	-3.2110-01	-2.5100-03	1.6660 00	-8.3480-01	-7.1590-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.8840 00	-5.7150-02	-9.6580 00	9.7070-01	1.7370 00	-7.8690-01	-9.6580 00	1.2620 00

PLANE ELEMENT STRESSES

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
11	1	5.0000 00	7.5000-01	1.5460 00	-3.2000-01	1.0270-04	1.4070 00	-8.3190-01	-6.0300-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.6100 00	-6.3490-02	-1.1240 01	8.3680-01	1.4900 00	-6.8570-01	-1.1240 01	1.0880 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
12	1	7.0000 00	7.5000-01	1.2660 00	-3.2040-01	1.8590-04	1.1520 00	-8.3310-01	-4.9340-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.3420 00	-7.6320-02	-1.3430 01	7.0920-01	1.2510 00	-5.9290-01	-1.3430 01	9.2200-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
13	1	9.0000 00	7.5000-01	9.8450-01	-3.2040-01	-1.8560-04	8.9590-01	-8.3310-01	-3.8410-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.0800 00	-9.5280-02	-1.6530 01	5.8740-01	1.0200 00	-5.0770-01	-1.6530 01	7.6360-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
14	1	1.1000 01	7.5000-01	7.0370-01	-3.2000-01	-1.0900-04	6.4040-01	-8.3190-01	-2.7450-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		8.2740-01	-1.2380-01	-2.1140 01	4.7560-01	8.0130-01	-4.3540-01	-2.1140 01	6.1830-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
15	1	1.3000 01	7.5000-01	4.2050-01	-3.2110-01	2.5360-03	3.8160-01	-8.3480-01	-1.6170-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		5.9460-01	-1.7160-01	-2.8470 01	3.8310-01	6.0800-01	-3.8800-01	-2.8470 01	4.9800-01

PLANE ELEMENT STRESSES

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
16	1	1.5000 01	7.5000-01	1.3860-01	-3.1990-01	-1.0900-02	1.3040-01	-8.3160-01	-6.3990-02
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.9230-01	-2.6460-01	-3.8420 01	3.2850-01	4.6020-01	-3.9360-01	-3.8420 01	4.2700-01

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
17	1	1.0000 00	1.2500 00	3.5180 00	-2.272D-01	1.818D-02	3.194D 00	-5.908D-01	-1.355D 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.533D 00	3.438D-03	-3.699D 00	1.765D 00	3.213D 00	-1.375D 00	-3.699D 00	2.294D 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
18	1	3.000D 00	1.250D 00	3.049D 00	-2.258D-01	-4.506D-03	2.776D 00	-5.870D-01	-1.193D 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.069D 00	-2.111D-02	-4.207D 00	1.543D 00	2.797D 00	-1.214D 00	-4.207D 00	2.006D 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
19	1	5.000D 00	1.250D 00	2.577D 00	-2.267D-01	5.541D-04	2.345D 00	-5.895D-01	-1.005D 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		2.597D 00	-1.924D-02	-4.990D 00	1.308D 00	2.371D 00	-1.030D 00	-4.990D 00	1.701D 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
20	1	7.000D 00	1.250D 00	2.110D 00	-2.266D-01	2.264D-04	1.926D 00	-5.891D-01	-8.227D-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		2.134D 00	-2.384D-02	-6.061D 00	1.079D 00	1.951D 00	-8.540D-01	-6.061D 00	1.403D 00

PLANE ELEMENT STRESSES

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
21	1	9.0000 00	1.250D 00	1.640D 00	-2.266D-01	-2.209D-04	1.493D 00	-5.891D-01	-6.399D-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.671D 00	-3.094D-02	-7.721D 00	8.509D-01	1.533D 00	-6.798D-01	-7.721D 00	1.106D 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
22	1	1.100D 01	1.250D 00	1.173D 00	-2.267D-01	-5.750D-04	1.067D 00	-5.895D-01	-4.579D-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.215D 00	-4.287D-02	-1.057D 01	6.290D-01	1.122D 00	-5.129D-01	-1.057D 01	8.177D-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
23	1	1.300D 01	1.250D 00	7.021D-01	-2.257D-01	4.549D-03	6.371D-01	-5.869D-01	-2.697D-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		7.688D-01	-6.213D-02	-1.646D 01	4.155D-01	7.238D-01	-3.564D-01	-1.646D 01	5.401D-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
24	1	1.500D 01	1.250D 00	2.322D-01	-2.272D-01	-1.821D-02	2.184D-01	-5.908D-01	-1.071D-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.664D-01	-1.525D-01	-3.057D 01	2.594D-01	3.929D-01	-2.816D-01	-3.057D 01	3.373D-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
25	1	1.000D 00	1.750D 00	4.920D 00	-8.703D-02	2.542D-02	4.467D 00	-2.263D-01	-1.896D 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		4.922D 00	2.387D-02	-1.018D 00	2.449D 00	4.469D 00	-1.898D 00	-1.018D 00	3.184D 00

P L A N E E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
26	1	5.0000 00	1.7500 00	4.2640 00	-8.4410-02	-6.9070-03	3.8830 00	-2.1950-01	-1.6690 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		4.2660 00	-8.5750-03	-1.1320 00	2.1370 00	3.8850 00	-1.6720 00	-1.1320 00	2.7790 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
27	1	5.5000 00	1.7500 00	3.6100 00	-8.6630-02	1.9610-03	3.2850 00	-2.2520-01	-1.4060 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.6130 00	-1.1720-04	-1.3740 00	1.8060 00	3.2870 00	-1.4090 00	-1.3740 00	2.3480 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
28	1	7.0000 00	1.7500 00	2.9530 00	-8.5770-02	-6.8160-04	2.6870 00	-2.2300-01	-1.1520 00
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		2.9550 00	-3.1700-03	-1.5620 00	1.4790 00	2.6910 00	-1.1560 00	-1.5620 00	1.9230 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
29	1	9.0000 00	1.7500 00	2.2970 00	-8.5780-02	6.8180-04	2.0900 00	-2.2300-01	-8.9530-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		2.3010 00	-2.5170-03	-2.1360 00	1.1520 00	2.0940 00	-8.9950-01	-2.1360 00	1.4970 00
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
30	1	1.1000 01	1.7500 00	1.6400 00	-8.6610-02	-1.9610-03	1.4930 00	-2.2520-01	-6.4130-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		1.6440 00	-6.5180-03	-3.0110 00	8.2550-01	1.4990 00	-6.4730-01	-3.0110 00	1.0730 00

P L A N E E L E M E N T S T R E S S E S

ELEMENT GROUP NUMBER 1

ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
31	1	1.3000 01	1.7500 00	9.8590-01	-8.4450-02	6.9050-03	8.9450-01	-2.1960-01	-3.7820-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		9.9310-01	-3.2600-04	-4.8940 00	4.9670-01	9.0390-01	-3.8760-01	-4.8940 00	6.4570-01
ELEMENT NUMBER	MATERIAL NUMBER	COORDINATE X1	COORDINATE X2	STRESS 11	STRESS 12	STRESS 22	STRAIN 11	STRAIN 12	STRAIN 22
32	1	1.5000 01	1.7500 00	3.3020-01	-8.6930-02	-2.5420-02	3.1040-01	-2.2600-01	-1.5190-01
		STRESS 1	STRESS 2	ANGLE	SHEAR	STRAIN 1	STRAIN 2	ANGLE	SHEAR
		3.5040-01	-4.5530-02	-1.3030 01	1.9790-01	3.3660-01	-1.7810-01	-1.3030 01	2.5730-01

PLANE STRAIN P.R.=0.300 1X1 MHU TERM , 1X1 LAMBDA TERM , HOURGLASS ADDED

S O L U T I O N T I M E L O G I N S E C O N D S

INPUT PHASE . . . . . =	0.55
FORMATION OF STIFFNESS AND LOAD . . . . . =	0.50
FACTOR STIFFNESS . . . . . =	0.40
DISPLACEMENT AND STRESS ANALYSIS . . . . . =	0.82
T O T A L S O L U T I O N T I M E . . . . . =	2.26