MEEN 673: NONLINEAR FINITE ELEMENT ANALYSIS

Time-Dependent Problems

CONTENTS

Transient problems
- Semi-discretization
- Time approximations
- Mass lumping
- Stability and accuracy
- Computer implementation
- Numerical examples
INTRODUCTION

Equations of motion

Static problem
(set all derivatives with respect to time $t$ to zero)

Construct weak form
Construct FE model

Eigenvalue problem
(replace the solution with periodic or decay type solution with respect to time $t$)

Semidiscretization
(construct weak form in spatial coordinates and semi-discrete FEM)

Construct weak form
Construct FE model

Full discretization
(use time approximations)
INTRODUCTION (continued)

**Equilibrium (static) problem**

Equilibrium configuration

\[- \frac{d}{dx} \left( T \frac{du}{dx} \right) = f(x)\]
\[u(0) = 0, \quad u(L) = 0\]

**Transient problem**

Transient configurations

\[\frac{\partial}{\partial t} \left( \rho \frac{\partial u}{\partial t} \right) - \frac{\partial}{\partial x} \left( T \frac{\partial u}{\partial x} \right) = f(x, t)\]
\[B.C. : \quad u(0, t) = 0, \quad u(L, t) = 0\]
\[I.C. : \quad u(x, 0) = u_0(x), \quad \dot{u}(x, 0) = v_0(x)\]

Eigenvalue and Dynamics Problems: 3
TRANSIENT ANALYSIS
(steps involved)

Model Equation

\[
c_1 \frac{\partial u}{\partial t} + c_2 \frac{\partial^2 u}{\partial t^2} - \frac{\partial}{\partial x} \left( a(x,u) \frac{\partial u}{\partial x} \right) + \frac{\partial^2}{\partial x^2} \left( b(x,u) \frac{\partial^2 u}{\partial x^2} \right) = f(x,t)
\]

Approximate solution

\[
u(x,t) \approx u_h(x,t) = \sum_{j=1}^{n} \Delta_j(t) \varphi_j(x)
\]

1. Spatial approximation (semidiscretization)

\[
C \ddot{\Delta} + M \dot{\Delta} + K(\Delta)\Delta = F
\]

2. Time approximation (full discretization)

\[
\hat{K}(\Delta_s, \Delta_{s+1}) \Delta_{s+1} = F_{s,s+1}
\]
SPATIAL APPROXIMATION

Model Equation

\[ c_1 \frac{\partial u}{\partial t} + c_2 \frac{\partial^2 u}{\partial t^2} - \frac{\partial}{\partial x} \left( a \frac{\partial u}{\partial x} \right) + \frac{\partial^2}{\partial x^2} \left( b \frac{\partial^2 u}{\partial x^2} \right) = f(x,t) \]

Weak Form for semi-discretization

\[ 0 = \int_{x_a}^{x_b} w_i(x) \left[ c_1 \frac{\partial u_h}{\partial t} + c_2 \frac{\partial^2 u_h}{\partial t^2} - \frac{\partial}{\partial x} \left( a \frac{\partial u_h}{\partial x} \right) + \frac{\partial^2}{\partial x^2} \left( b \frac{\partial^2 u_h}{\partial x^2} \right) - f(x,t) \right] dx \]

\[ = \int_{x_a}^{x_b} \left[ c_1 w_i \frac{\partial u_h}{\partial t} + c_2 w_i \frac{\partial^2 u_h}{\partial t^2} + \frac{d w_i}{dx} \left( a \frac{\partial u_h}{\partial x} \right) + \frac{d^2 w_i}{dx^2} \left( b \frac{\partial^2 u_h}{\partial x^2} \right) - w_i f(x,t) \right] dx \]

\[ - w_i(x_a) Q_1 - w_i(x_b) Q_3 - \left( - \frac{d w_i}{dx} \right)_{x_a} Q_2 - \left( - \frac{d w_i}{dx} \right)_{x_b} Q_4 \]

Eigenvalue and Dynamics Problems: 5
SPATIAL DISCRETIZATION

Finite Element Model

**Approximation**

\[ u(x,t) \approx u_h(x,t) = \sum_{j=1}^{n} \Delta_j(t) \varphi_j(x) \]

**Finite element model**

\[
\begin{align*}
C \dot{\Delta} + M \ddot{\Delta} + K \Delta &= F \\
C_{ij}^e &= \int_{x_a}^{x_b} c_{ij} \varphi_i \varphi_j \, dx, \quad M_{ij}^e = \int_{x_a}^{x_b} c_{ij} \dot{\varphi}_i \dot{\varphi}_j \, dx \\
K_{ij}^e &= \int_{x_a}^{x_b} \left( b \frac{d^2 \varphi_i}{dx^2} \frac{d^2 \varphi_j}{dx^2} + a \frac{d \varphi_i}{dx} \frac{d \varphi_j}{dx} \right) \, dx \\
F_i^e &= \int_{x_a}^{x_b} f \varphi_i \, dx + \varphi_i(x_a)Q_1 + \varphi_i(x_b)Q_3 + \left(- \frac{d \varphi_i}{dx}\right)_{x_a}Q_2 + \left(- \frac{d \varphi_i}{dx}\right)_{x_b}Q_4
\end{align*}
\]
TIME APPROXIMATIONS

PARABOLIC EQUATION (heat transfer, fluid mechanics, and like problems)

\[ C \dot{u} + K(u)u = F \]

\[ C_{ij}^e = \int_{x_a}^{x_b} c_1 \psi_i \psi_j \, dx, \quad K_{ij}^e = \int_{x_a}^{x_b} \left( a(x,u,u_x) \frac{d\psi_i}{dx} \frac{d\psi_j}{dx} \right) \, dx \]

\[ F_i^e = \int_{x_a}^{x_b} f \psi_i \, dx + \psi_i(x_a)Q_1 + \psi_i(x_b)Q_2 \]

HYPERBOLIC EQUATION (structural mechanics problems)

\[ C \ddot{\Delta} + M \ddot{\Delta} + K(\Delta)\Delta = F \]

Eigenvalue and Dynamics Problems: 7
TIME APPROXIMATIONS OF PARABOLIC EQUATIONS

Approximation of the first derivative

\[ \dot{u}_j^s \approx \frac{u_j^{s+1} - u_j^s}{\Delta t_{s+1}} \], forward difference
\[ \dot{u}_j^{s+1} \approx \frac{u_j^{s+1} - u_j^s}{\Delta t_{s+1}} \], backward difference

Alfa (\(\alpha\))-family of approximation

\[ \alpha \dot{u}_j^{s+1} + (1 - \alpha)\dot{u}_j^s \approx \frac{u_j^{s+1} - u_j^s}{\Delta t_{s+1}}, \quad 0 \leq \alpha \leq 1 \]
\[ u_j^{s+1} = u_j^s + \Delta t_{s+1} \left[ \alpha \dot{u}_j^{s+1} + (1 - \alpha)\dot{u}_j^s \right] \]
TIME APPROXIMATIONS (Parabolic)

Alfa-family of approximation (Parabolic equation)

\[ C\dot{u} + K(u)u = F, \quad 0 < t < T \]

\[ \Rightarrow C\dot{u}^s + K(u)^s = F^s, \quad C\dot{u}^{s+1} + K(u)^{s+1} = F^{s+1} \]

\[ u^{s+1} = u^s + \Delta t^{s+1} \left[ \alpha \dot{u}^{s+1} + (1 - \alpha)\dot{u}^s \right] \]

\[ C\dot{u}^{s+1} = C\dot{u}^s + \Delta t^{s+1} \left[ \alpha C\dot{u}^{s+1} + (1 - \alpha)C\dot{u}^s \right] \]

\[ C\dot{u}^{s+1} = F^{s+1} - K^{s+1}u^{s+1} \quad (\text{where}) \]

\[ \dot{K}^{s+1} = \frac{\Delta t^{s+1}}{1 - \alpha} K^{s+1} + C \]

\[ \hat{F}^{s+1} = [(1 - \alpha)\Delta t^{s+1} K^{s+1} + C]u^s + \Delta t^{s+1} \left[ \alpha \hat{F}^{s+1} + (1 - \alpha)F^s \right] \]
TIME APPROXIMATIONS (Hyperbolic)

Semidiscrete FE model

\[ C^e \ddot{u}^e + M^e \dot{u}^e + K^e(u^e)u^e = F^e \]

Newmark scheme (hyperbolic equations)

\[ u^{s+1} = u^s + \Delta t \dot{u}^s + \frac{1}{2}(\Delta t)^2 \ddot{u}^{s,\gamma} \]

\[ \dot{u}^{s+1} = \dot{u}^s + \Delta t \ddot{u}^{s,\alpha}, \quad \dddot{u}^{s,\theta} \equiv (1 - \theta)\dot{u}^s + \theta \ddot{u}^{s+1} \]

Fully discretized model

\[ \hat{K}_{s+1} u^{s+1} = \hat{F}^{s+1}, \quad \hat{K}_{s+1} = K_{s+1} + a_3 M_{s+1} + a_5 C_{s+1} \]

\[ \hat{F}^{s+1} = F^{s+1} + M_{s+1} \left( a_3 u^s + a_4 \dot{u}^s + a_5 \ddot{u}^s \right) + C_{s+1} \left( a_3 u^s + a_6 \dot{u}^s + a_7 \dddot{u}^s \right) \]

Transient Problems: 10
General form of the time-marching scheme

\[ \hat{K}u^{s+1} = Bu^s + F^{s,s+1} \]

The scheme is called *explicit* if the coefficient matrix \( \hat{K} \) is diagonal (and hence, no inversion of equations is necessary); otherwise, the scheme is said to be implicit.

\[ \hat{K}^{s+1} u^{s+1} = F^{s+1} \]

\[ \hat{K}^{s+1} = \alpha \Delta t_{s+1} K_{s+1} + C, \quad \hat{F}^{s+1} = \left[ (1 - \alpha) \Delta t_{s+1} K_{s+1} + C \right] u^s + \Delta t_{s+1} \left[ \alpha F^{s+1} + (1 - \alpha) F^s \right] \]

The alfa-family scheme is *explicit* if and only if

(1) \( \alpha = 0 \) and (2) \( C \) is diagonal.
ROW-SUM MASS LUMPING

For the Lagrange linear and quadratic elements we have

\[
[M^e]_C = \frac{\rho A_e h_e}{6} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}, \quad [M^e]_C = \frac{\rho A_e h_e}{30} \begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix}
\]

\[
[M^e]_L = \frac{\rho A_e h_e}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad [M^e]_L = \frac{\rho A_e h_e}{6} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]
STABILITY OF APPROXIMATIONS

\[ \hat{K}^{s+1}u^{s+1} = \bar{K}u^s + F^{s,s+1} \]
\[ u^{s+1} = Au^s + B^{s,s+1} \]

The scheme is called **stable** if the repeated solution of the above equation does not result in unbounded solution \( u_{s+1} \). The necessary and sufficient condition for the above scheme to be stable is that the maximum eigenvalue of the coefficient matrix \( A \) is less than unity:

\[ \lambda_{\text{max}}^A \leq 1 \]
STABILITY OF APPROXIMATIONS
(continued)

Alfa-family of approximation scheme

\[ \alpha \geq \frac{1}{2}, \quad \text{the scheme is stable} \]
\[ \alpha < \frac{1}{2}, \quad \text{the scheme is conditionally stable} \]

\[
\begin{array}{c}
\alpha = 0.0, \text{ Forward difference (Euler) scheme (conditionally stable)} \\
\alpha = 0.5, \text{ Crank-Nicolson's scheme (stable)} \\
\alpha = \frac{2}{3}, \text{ Galerkin's scheme (stable)} \\
\alpha = 1.0, \text{ Backward difference scheme (stable)} \\
\end{array}
\]

Stability condition:

\[
\Delta t \leq (\Delta t)_{\text{crit}} = \frac{2}{(1 - 2\alpha)\lambda_{\text{max}}}.
\]
STABILITY OF APPROXIMATIONS
(continued)

Newmark’s scheme for Structural Dynamics

\[( -\lambda M + K ) u = 0 \]

Stability condition: \( \Delta t \leq (\Delta t)_{\text{crit}} = \frac{2}{(\alpha - \gamma)\lambda_{\text{max}}} \)

\( \alpha = 0.5, \gamma = 2\beta = 0.5, \) Constant-average acceleration scheme (stable)
\( \alpha = 0.5, \gamma = 2\beta = \frac{1}{3}, \) Linear acceleration scheme (conditionally stable)
\( \alpha = 1.5, \gamma = 2\beta = 1.6, \) Galerkin's scheme (stable)
\( \alpha = 1.5, \gamma = 2\beta = 2.0, \) Backward difference scheme (stable)

(\( \Delta t \)\)\textsubscript{crit} gets smaller as the mesh is refined.

Transient Problems: 15
**Load loop**
DO NL = 1, NLOAD
F = F + DF

**Time loop**
DO NT = 1, NTIME

ITER = 0

ITER = ITER + 1

**Initialize global** $K_{ij}$, $M_{ij}$, and $f_i$

DO $N = 1$ to $NEM$

Transfer global information (material properties, geometry and solution) to element

CALL ELKMF to calculate $K_{ij}^{(e)}$, $M_{ij}^{(e)}$, and $f_i^{(e)}$, and assemble to form global $K_{ij}$, $M_{ij}$, and $F_i$

Impose boundary conditions and solve the equations

error < $\varepsilon$

yes

no

yes

iter < itmax

no

Update velocities, accelerations, and print solution

Write a message

STOP

IF $NT > NTIME$

no

yes
Read the necessary data for time-dependent problems

IF(ITEM.NE.0) THEN
    READ(IN,*) C0,CX,CY
    WRITE(ITT,820)
    WRITE(ITT,540) C0,CX,CY
    READ(IN,*) NTIME
    READ(IN,*) DT,ALFA,GAMA,EPSLN
    A1=ALFA*DT
    A2=(1.0-ALFA)*DT
    DO 40 I=1,NEQ
        GLU(I)=0.0
    40

    IF(ITEM.EQ.1) THEN
        IF(NSSV.NE.0) THEN
            DO 50 I=1,NSSV
                VSSV(I)=VSSV(I)*DT
            50
        ENDIF
        ELSE
            DT2=DT*DT
            A3=2.0/GAMA/DT2
            A4=A3*DT
            A5=1.0/GAMA-1.0

            C ***It is assumed that the initial conditions are all zero***
            DO 70 I=1,NEQ
                GLV(I)=0.0
                GLA(I)=0.0
            70
        ENDIF
    ENDIF

    C ***Initialize the arrays***

    DO 250 N=1,NEM
        DO 200 I=1,NPE
            NI=NOD(N,I)
            IF(NONLIN.GT.0 .OR. ITEM.GT.0) THEN
                ELU(I)=GLU(NI) !Transfer of the current solution
                ELU0(I)=GLP(NI) !Transfer of the previous time step solution
            ENDIF
            IF(ITEM.EQ.2) THEN
                ELU1(I)=GLV(NI) !Transfer of the previous first time derivative
                ELU2(I)=GLA(NI) !Transfer of the previous second time derivative
            ENDIF
            ELXY(I,1)=GLXY(NI,1)
            ELXY(I,2)=GLXY(NI,2)
        200
    250

    Call Subroutine ELMATRCS2D to compute ELK, ELK-HAT, etc.
    and assemble them into global matrices GLK

    .............
SUBROUTINE ELMATRCS2D(MODEL,NPE,NN,NONLIN)
C     ________________________________________________________________
C     Element calculations based on linear and quadratic rectangular
elements with isoparametric formulation.
C     ________________________________________________________________
IMPLICIT REAL*8(A-H,O-Z)
COMMON/STF/ELF(9),ELK(9,9),ELXY(9,2),ELU(9)
COMMON/PST/A10,A1X,A1Y,A20,A2X,A2Y,A00,F0,FX,FY,
COMMON/SHP/SFL(9),GDSFL(2,9)
COMMON/PNT/IPDF,IPDR,NIPF,NIPR
DIMENSION GAUSPT(5,5),GAUSWT(5,5),TANG(9,9)
COMMON/IO/IN,IT
C
C DATA GAUSPT/5*0.0D0, −0.57735027D0, 0.57735027D0, 3*0.0D0,
C 2 −0.77459667D0, 0.0D0, 0.77459667D0, 2*0.0D0, −0.86113631D0,
C 3 −0.33998104D0, 0.33998104D0, 0.86113631D0, 0.0D0, −0.90617984D0,
C 4 −0.53846931D0,0.0D0,0.53846931D0,0.90617984D0/
C DATA GAUSWT/2.0D0, 4*0.0D0, 2*1.0D0, 3*0.0D0, 0.55555555D0,
C 2 0.88888888D0, 0.55555555D0, 2*0.0D0, 0.34785485D0,
C 3 2*0.65214515D0, 0.34785485D0, 0.0D0, 0.23692688D0,
C 4 0.47862867D0, 0.56888888D0, 0.47862867D0, 0.23692688D0/
C NDF=NN/NPE
C Initialize the arrays
C DO 100 I = 1,NPE
   ELF(I) = 0.0
DO 100 J = 1,NPE
   IF(NONLIN.GT.1)THEN
      TANG(I,J)=0.0
   ENDIF
100   ELK(I,J)= 0.0
C Do-loops on numerical (Gauss) integration begin here.
C DO 200 NI = 1,IPDF
DO 200 NJ = 1,IPDF
   XI = GAUSPT(NI,IPDF)
   ETA = GAUSPT(NJ,IPDF)
   CALL INTERPLN2D(NPE,XI,ETA,DET,ELXY)
   CNST = DET*GAUSWT(NI,IPDF)*GAUSWT(NJ,IPDF)
   X=0.0
   Y=0.0
DO 120 I=1,NPE
   X=X+ELXY(I,1)*SFL(I)
120      Y=Y+ELXY(I,2)*SFL(I)
IF(MODEL.EQ.1) THEN
C Define the coefficients of the differential equation
C
AXX=A10+A1X*X+A1Y*Y
AYY=A20+A2X*X+A2Y*Y
FXY=F0+FX*X+FY*Y
C
IF(NONLIN.GT.0) THEN
U=0.0
DO 140 I = 1, NPE
U = U + ELU(I) * SFL(I)
140 CONTINUE
AXX = AXX + A1U*U + A1UX*UX + A1UY*UY
AYY = AYY + A2U*U + A2UX*UX + A2UY*UY
ENDIF
C
DO 180 I = 1, NPE
ELF(I) = ELF(I) + FXY * SFL(I) * CNST
DO 160 J = 1, NPE
S00 = SFL(I) * SFL(J) * CNST
S11 = GDSFL(1, I) * GDSFL(1, J) * CNST
S22 = GDSFL(2, I) * GDSFL(2, J) * CNST
S12 = GDSFL(1, I) * GDSFL(2, J) * CNST
S21 = GDSFL(2, I) * GDSFL(1, J) * CNST
ELK(I, J) = ELK(I, J) + AXX * S11 + AYY * S22 + A00 * S00
C Write statements here for the part needed to be added to K to obtain T
C
160 CONTINUE
180 CONTINUE
ENDIF
200 CONTINUE
C
C Write statements here to compute the residual vector and tangent matrix
C
RETURN
END
DO 200 NI = 1, NGPF  ! full Gauss integration loop
DO 200 NJ = 1, NGPF

C  Define the linear and nonlinear coefficients of the differential equation
C

IF(ITEM.GT.0)THEN  
   CXY = C0 + CX*X + CY*Y  ! Define the coefficient of the time derivative
   ENDIF
C
C IF(ITEM.GT.0)THEN  ! Define the solution vector $\mathbf{u}_s$ and its derivatives
   UP  = 0.0
   UPX = 0.0
   UPY = 0.0
   DO 140 I=1,NPE
      UP = UP + ELU0(I)*SF(I)
      UPX = UPX + ELU0(I)*GDSF(1,I)
   140        UPY = UPY + ELU0(I)*GDSF(2,I)
   APXX = A11 + A1U*UP + A1UX*UPX + A1UY*UPY
   APYY = A22 + A2U*UP + A2UX*UPX + A2UY*UPY
   ENDIF
C
C Define the element coefficient matrices ELK, ELF, and ELM
C
C IF(ITEM.GT.0)THEN
   ELM(I,J) = ELM(I,J) + CXY*S00
   IF(NONLIN.GT.0)THEN  ! Define $\mathbf{K}$ ($a_{xx} = a_{yy} = 0$)
      ELK0(I,J) = ELK0(I,J) + APXX*SXX + APYY*SYY + A00*S00
   ENDIF
   ENDIF
C
200 CONTINUE
C
C Compute $\hat{\mathbf{K}}_{s+1}$ and $\hat{\mathbf{F}}$
C
IF(ITEM.EQ.1) THEN  ! for parabolic equations
   DO 220 I=1,NN
      SUM = 0.0
   DO 210 J=1,NN
      IF(NONLIN.GT.0)THEN
         SUM = SUM + ((ELM(I,J) - A2*ELK0(I,J)) * ELU0(J))
      ELSE
         SUM = SUM + (ELM(I,J) - A2*ELK(I,J)) * ELU0(J)
      ENDIF
   210       ELK(I,J) = ELM(I,J) + A2*ELK(I,J)
   220       ELF(I) = (A1 + A2)*ELF(I) + SUM
   ENDIF
IF(ITEM.GT.1) THEN  ! for hyperbolic equations
   DO 270 I = 1,NN
      SUM = 0.0
   DO 260 J = 1,NN
      SUM = SUM + ELM(I,J) * (A3*ELU0(J) + A4*ELU1(J) + A5*ELU2(J))
   260       ELK(I,J) = ELK(I,J) + A3*ELM(I,J)
   270       ELF(I) = ELF(I) + SUM
   ENDIF
NUMERICAL EXAMPLES

Heat transfer in a rod

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = 0, \quad 0 < x < 1$$

$$u(0, t) = 0, \quad \frac{\partial u}{\partial x}(1, t) = 0$$

$$u(x, 0) = 1$$

one-element model:

$$\Delta t_{\text{crit}} = \frac{2}{3} = 0.66667$$

two-element model:

$$\Delta t_{\text{crit}} = \frac{2}{31.689} = 0.063$$

Eigenvalue and Dynamics Problems: 21
NUMERICAL EXAMPLES
(continued)

Bending of a clamped beam

\[ \frac{\partial^2 w}{\partial t^2} + \frac{\partial^4 w}{\partial x^4} = 0, \quad 0 < x < 1 \]

\[ w(0, t) = 0, \quad \frac{\partial w}{\partial x}(0, t) = 0, \]

\[ w(1, t) = 0, \quad \frac{\partial w}{\partial x}(1, t) = 0 \]

\[ w(x, 0) = \sin \pi x - \pi x(1 - x) \]

one-element model:

\[ \Delta t_{\text{crit}} = \frac{12}{516.93} = 0.023214 \]

two-element model:

\[ \Delta t_{\text{crit}} = 0.00897 \]

![Graph showing deflection over time for different time steps and gamma values](image)
NUMERICAL EXAMPLES
(continued)

Eigenvalue and Dynamics Problems:
23
**Figure 5.7.3:** Actual and computational domains of the transient heat transfer problem.

**Figure 5.7.4:** Transient response predicted by various schemes.