CHAPTER SEVEN

Analysis of Elastic Solids

Computer Implementation 7.1 *(Matlab)*

The element equations for a triangular element for a plane stress and plane strain problems can be generated conveniently by writing three functions in *Matlab*. The following PlaneTriElement, PlaneTriLoadTerm and PlaneTriResults functions are similar to those presented in Chapter 1 except that they are little more general and can handle both plane stress and plane strain problems as well as thermal effects and body forces.

MatlabFiles\Chap7\PlaneTriElement.m

```
function [k, r] = PlaneTriElement(type, e, nu, h, alpha, deltaT, bx, by, coord)
% [k, r] = PlaneTriElement(e, nu, h, alpha, deltaT, bx, by, coord)
% Generates for a triangular element for plane stress or plane strain problem
% e = Modulus of elasticity
% nu = Poisson's ratio
% h = Thickness
% alpha = coefficient of thermal expansion
```

```
% deltaT = temperature change
% bx, by = components of the body force
% coord = coordinates at the element ends

x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
b1 = y2 - y3; b2 = y3 - y1; b3 = y1 - y2;
c1 = x3 - x2; c2 = x1 - x3; c3 = x2 - x1;
f1 = x2*y3 - x3*y2; f2 = x3*y1 - x1*y3; f3 = x1*y2 - x2*y1;
A = (f1 + f2 + f3)/2;
switch (type)
case 1
   e0 = alpha*deltaT*[1; 1; 0];
   C = e/(1 - nu^2)*[1, nu, 0; nu, 1, 0; 0, 0, (1 - nu)/2];
case 2
   e0 = (1 + nu)*alpha*deltaT*[1; 1; 0];
   C = e/((1 + nu)*(1 - 2*nu))*[1 - nu, nu, 0; nu, 1 - nu, 0;
      0, 0, (1 - 2*nu)/2];
end
B = [b1, 0, c1; 0, c1, b1; b2, 0, c2; 0, c2, b2;
   b3, 0, c3; 0, c3, b3]/(2*A);
k = h*A*(B*C*B');
r = h*A*(B*C*e0 + [bx; by; bx; by; bx; by]/3);
```

MatlabFiles\Chap7\PlaneTriLoad.m

```
function rq = PlaneTriLoad(side, qn, qt, h, coord)
% PlaneTriLoad(side, qn, qt, h, coord)
% Generates equivalent load vector for a triangular element
% side = side over which the load is specified
% qn, qt = load components in the normal and the tangential direction
% h = thickness
% coord = coordinates at the element ends

x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
switch (side)
case 1
   L=sqrt((x2-x1)^2+(y2-y1)^2);
   nx=(y2-y1)/L; ny=-(x2-x1)/L;
   qx = nx*qn - ny*qt;
   qy = ny*qn + nx*qt;
   rq = h*L/2 * [qx; qy; qx; qy; 0; 0];
case 2
   L=sqrt((x2-x3)^2+(y2-y3)^2);
```

```
    nx=(y3-y2)/L; ny=-(x3-x2)/L;
    qx = nx*qn - ny*qt;
    qy = ny*qn + nx*qt;
    rq = h*L/2 * [0; 0; qx; qy; qx; qy];
case 3
    L=sqrt((x3-x1)^2+(y3-y1)^2);
    nx=(y1-y3)/L; ny=-(x1-x3)/L;
    qx = nx*qn - ny*qt;
    qy = ny*qn + nx*qt;
    rq = h*L/2 * [qx; qy; 0; 0; qx; qy];
end
```

MatlabFiles\Chap7\PlaneTriResults.m

```
function se = PlaneTriResults(type, e, nu, alpha, deltaT, coord, dn)
% se = PlaneTriResults(typ, e, nu, alpha, deltaT, coord, dn)
% Computes element solution for a plane stress/strain triangular element
% e = modulus of elasticity
% nu = Poisson's ratio
% alpha = coefficient of thermal expansion
% deltaT = temperature change
% coord = nodal coordinates
% dn = nodal displacements
% Following are the output variables are at element center
% {strains, stresses, principal stresses, effective stress}
x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
x=(x1+x2+x3)/3; y=(y1+y2+y3)/3;
switch (type)
case 1
    e0 = alpha*deltaT*[1; 1; 0];
    C = e/(1 - nu^2)*[1, nu, 0; nu, 1, 0; 0, 0, (1 - nu)/2];
case 2
    e0 = (1 + nu)*alpha*deltaT*[1; 1; 0];
    C = e/((1 + nu)*(1 - 2*nu))*[1 - nu, nu, 0; nu, 1 - nu, 0;
        0, 0, (1 - 2*nu)/2];
end

b1 = y2 - y3; b2 = y3 - y1; b3 = y1 - y2;
c1 = x3 - x2; c2 = x1 - x3; c3 = x2 - x1;
f1 = x2*y3 - x3*y2; f2 = x3*y1 - x1*y3; f3 = x1*y2 - x2*y1;
A = (f1 + f2 + f3)/2;
B = [b1, 0, c1; 0, c1, b1; b2, 0, c2; 0, c2, b2;
    b3, 0, c3; 0,c3, b3]/(2*A);
eps = B'*dn;
sig = C*(eps-e0)
```

```
sx = sig(1); sy= sig(2); sxy=sig(3);
PrincipalStresses = eig([sx,sxy; sxy,sy])
se = sqrt((sx - sy)^2 + sy^2 + sx^2 + 6*sxy^2)/sqrt(2);
```

MatlabFiles\Chap7\ThermalStressEx.m

```
% Plane stress model for thermal stresses example
e1 = 70000; nu1 = .33; alpha1 = 23*10^(-6);
e2 = 200000; nu2 = .3; alpha2 = 12*10^(-6); h = 5;
bx=0; by=0; deltaT = 70;
a = 150/2; b = 80/2; c = 100/2; d = 30/2;
nodes = [0, 0; c, 0; a,0; 0, d; c, d; a, d; 0, b; c, b; a, b];
conn = [1, 5, 4; 1, 2, 5; 2, 6, 5;
   2, 3, 6; 4, 8, 7; 4, 5, 8; 5, 9, 8; 5, 6, 9];
nel=size(conn,1); dof=2*size(nodes,1);
lmm=[];
for i=1:nel
   lm=[];
   for j=1:3
      lm=[lm, [2*conn(i,j)-1,2*conn(i,j)]];
   end
   lmm=[lmm; lm];
end
K=zeros(dof); R = zeros(dof,1);
% Generate equations for each element and assemble them.
for i=1:2
   con = conn(i,:);
   lm = lmm(i,:);
   [k, r] = PlaneTriElement(1, e1, nu1, h, alpha1, deltaT, bx, by, nodes(con,:));
   K(lm, lm) = K(lm, lm) + k;
   R(lm) = R(lm) + r;
end
for i=3:nel
   con = conn(i,:);
   lm = lmm(i,:);
   [k, r] = PlaneTriElement(1, e2, nu2, h, alpha2, deltaT, bx, by, nodes(con,:));
   K(lm, lm) = K(lm, lm) + k;
   R(lm) = R(lm) + r;
end

% Nodal solution and reactions
debc = [1,2,4,6,7,13]; ebcVals=zeros(length(debc),1);
[d, reactions] = NodalSoln(K, R, debc, ebcVals)
for i=1:2
   fprintf(1,'Results for element %3.0g \n',i)
   EffectiveStress=PlaneTriResults(1, e1, nu1, alpha1, deltaT, ...
      nodes(conn(i,:),:), d(lmm(i,:)))
```

```
end
for i=3:nel
    fprintf(1,'Results for element %3.0g \n',i)
    EffectiveStress=PlaneTriResults(1, e2, nu2, alpha2, deltaT, ...
        nodes(conn(i,:),:), d(lmm(i,:)))
end
```

```
>> ThermalStressEx

d =

         0
         0
    0.0513
         0
    0.0703
         0
         0
    0.0253
    0.0496
    0.0186
    0.0693
    0.0146
         0
    0.0446
    0.0498
    0.0389
    0.0716
    0.0367


reactions =

  1.0e+003 *

    2.5150
    1.3891
    0.3129
   -1.7020
    0.4562
   -2.9712

Results for element   1

sig =

  -46.6793
```

```
  -10.1709
   -3.5059


PrincipalStresses =

  -47.0129
   -9.8373


EffectiveStress =

   42.9477

Results for element   2

sig =

  -55.3796
  -44.1277
   -3.1397


PrincipalStresses =

  -56.1964
  -43.3109


EffectiveStress =

   50.9897

Results for element   3

sig =

   14.9716
   84.6275
  -21.5116


PrincipalStresses =

    8.8638
   90.7353
```

EffectiveStress =

86.6441

Results for element 4

sig =

-9.0613
23.9696
-5.4368

PrincipalStresses =

-9.9332
24.8415

EffectiveStress =

31.0245

Results for element 5

sig =

29.9479
-4.3978
-8.7956

PrincipalStresses =

-6.5192
32.0694

EffectiveStress =

35.7772

Results for element 6

```
sig =

   31.2874
    3.5569
   -9.4427


PrincipalStresses =

    0.6469
   34.1974


EffectiveStress =

   33.8785

Results for element   7

sig =

    5.0739
   -4.3071
   -5.9888


PrincipalStresses =

   -7.2236
    7.9904


EffectiveStress =

   13.1812

Results for element   8

sig =

   -8.6200
    5.9888
   -5.0739


PrincipalStresses =
```

```
-10.2094
  7.5781


EffectiveStress =

  15.4605
```

Computer Implementation 7.2 *(Matlab)*

In *Matlab*, the element equations for a quadrilateral element for a plane stress and plane strain problems can be generated in a manner similar to those presented for 2D BVP in Chapter 6. The following PlaneQuad4Element, PlaneQuad4LoadTerm and PlaneQuad4Results functions are developed for four node quadrilateral elements using 2×2 integration. Similar functions for 8 node quadrilateral element can easily be written.

MatlabFiles\Chap7\PlaneQuad4Element.m

```
function [k, r] = PlaneQuad4Element(type, e, nu, h, alpha, deltaT, bx, by, coord)
% [k, r] = PlaneQuad4Element(e, nu, h, alpha, deltaT, bx, by, coord)
% Generates for a triangular element for plane stress or plane strain problem
% e = Modulus of elasticity
% nu = Poisson's ratio
% h = Thickness
% alpha = coefficient of thermal expansion
% deltaT = temperature change
% bx, by = components of the body force
% coord = coordinates at the element ends

switch (type)
case 1
   e0 = alpha*deltaT*[1; 1; 0];
   c = e/(1 - nu^2)*[1, nu, 0; nu, 1, 0; 0, 0, (1 - nu)/2];
case 2
   e0 = (1 + nu)*alpha*deltaT*[1; 1; 0];
   c = e/((1 + nu)*(1 - 2*nu))*[1 - nu, nu, 0; nu, 1 - nu, 0;
      0, 0, (1 - 2*nu)/2];
end

% Use 2x2 integration. Gauss point locations and weights
pt=1/sqrt(3);
gpLocs = [-pt,-pt; -pt,pt; pt,-pt; pt,pt];
gpWts = [1,1,1,1];
k=zeros(8); r=zeros(8,1);
```

```
for i=1:length(gpWts)
    s = gpLocs(i, 1); t = gpLocs(i, 2); w = gpWts(i);
    n = [(1/4)*(1 - s)*(1 - t), (1/4)*(s + 1)*(1 -t), ...
            (1/4)*(s + 1)*(t + 1), (1/4)*(1 - s)*(t + 1)];
    dns=[(-1 + t)/4, (1 - t)/4, (1 + t)/4, (-1 - t)/4];
    dnt=[(-1 + s)/4, (-1 - s)/4, (1 + s)/4, (1 - s)/4];
    x = n*coord(:,1); y = n*coord(:,2);
    dxs = dns*coord(:,1); dxt = dnt*coord(:,1);
    dys = dns*coord(:,2); dyt = dnt*coord(:,2);
    J = [dxs, dxt; dys, dyt]; detJ = det(J);
    dnx = (J(2, 2)*dns - J(2, 1)*dnt)/detJ;
    dny = (-J(1, 2)*dns + J(1, 1)*dnt)/detJ;
    b = [dnx(1), 0, dnx(2), 0, dnx(3), 0, dnx(4), 0;
        0, dny(1), 0, dny(2), 0, dny(3), 0, dny(4);
        dny(1), dnx(1), dny(2), dnx(2), dny(3), dnx(3), dny(4), dnx(4)];
    n = [n(1),0,n(2),0,n(3),0,n(4),0;
        0,n(1),0,n(2),0,n(3),0,n(4)];
    k = k + h*detJ*w* b'*c*b;
    r = r + h*detJ*w*n'*[bx;by]+ h*detJ*w*b'*c*e0;
end
```

MatlabFiles\Chap7\PlaneQuad4Load.m

```
function rq = PlaneQuad4Load(side, qn, qt, h, coord)
% rq = PlaneQuad4Load(side, qn, qt, h, coord)
% Generates equivalent load vector for a triangular element
% side = side over which the load is specified
% qn, qt = load components in the normal and the tangential direction
% h = thickness
% coord = coordinates at the element ends

% Use 2 point integration. Gauss point locations and weights
pt=-1/sqrt(3);
gpLocs = [-pt, pt];
gpWts = [1,1];
rq=zeros(8,1);
for i=1:length(gpWts)
    a = gpLocs(i); w = gpWts(i);
    switch (side)
    case 1
        n = [(1 - a)/2, (1 + a)/2, 0, 0];
        dna = [-1/2, 1/2, 0, 0];
    case 2
        n = [0, (1 - a)/2, (1 + a)/2, 0];
        dna = [0, -1/2, 1/2, 0];
    case 3
        n = [0, 0, (1 - a)/2, (1 + a)/2];
```

```
        dna = [0, 0, -1/2, 1/2];
    case 4
        n = [(1 + a)/2, 0, 0, (1 - a)/2];
        dna = [1/2, 0, 0, -1/2];
    end
    dxa = dna*coord(:,1); dya = dna*coord(:,2);
    Jc=sqrt(dxa^2 + dya^2);
    nx = dya/Jc; ny = -dxa/Jc;
    qx = nx*qn - ny*qt;
    qy = ny*qn + nx*qt;
    n = [n(1),0,n(2),0,n(3),0,n(4),0;
        0,n(1),0,n(2),0,n(3),0,n(4)];
    rq = rq + h*Jc*w*n'*[qx; qy];
end
```

MatlabFiles\Chap7\PlaneQuad4Results.m

```
function se = PlaneQuad4Results(type, e, nu, alpha, deltaT, coord, dn)
% se = PlaneQuad4Results(type, e, nu, alpha, deltaT, coord, dn)
% Computes element solution for a plane stress/strain quad element
% e = modulus of elasticity
% nu = Poisson's ratio
% alpha = coefficient of thermal expansion
% deltaT = temperature change
% coord = nodal coordinates
% dn = nodal displacements
% Following are the output variables are at element center
% {strains, stresses, principal stresses, effective stress}
switch (type)
case 1
    e0 = alpha*deltaT*[1; 1; 0];
    c = e/(1 - nu^2)*[1, nu, 0; nu, 1, 0; 0, 0, (1 - nu)/2];
case 2
    e0 = (1 + nu)*alpha*deltaT*[1; 1; 0];
    c = e/((1 + nu)*(1 - 2*nu))*[1 - nu, nu, 0; nu, 1 - nu, 0;
        0, 0, (1 - 2*nu)/2];
end
s = 0; t = 0;
n = [(1/4)*(1 - s)*(1 - t), (1/4)*(s + 1)*(1 -t), ...
        (1/4)*(s + 1)*(t + 1), (1/4)*(1 - s)*(t + 1)];
dns=[(-1 + t)/4, (1 - t)/4, (1 + t)/4, (-1 - t)/4];
dnt=[(-1 + s)/4, (-1 - s)/4, (1 + s)/4, (1 - s)/4];
x = n*coord(:,1); y = n*coord(:,2);
dxs = dns*coord(:,1); dxt = dnt*coord(:,1);
dys = dns*coord(:,2); dyt = dnt*coord(:,2);
J = [dxs, dxt; dys, dyt]; detJ = det(J);
dnx = (J(2, 2)*dns - J(2, 1)*dnt)/detJ;
```

```
dny = (-J(1, 2)*dns + J(1, 1)*dnt)/detJ;
b = [dnx(1), 0, dnx(2), 0, dnx(3), 0, dnx(4), 0;
    0, dny(1), 0, dny(2), 0, dny(3), 0, dny(4);
    dny(1), dnx(1), dny(2), dnx(2), dny(3), dnx(3), dny(4), dnx(4)];
eps = b*dn;
sig = c*(eps-e0)
sx = sig(1); sy= sig(2); sxy=sig(3);
PrincipalStresses = eig([sx,sxy; sxy,sy])
se = sqrt((sx - sy)^2 + sy^2 + sx^2 + 6*sxy^2)/sqrt(2);
```

Using these functions finite element equations for any four node quadrilateral element for a plane stress or plane strain problem can easily be written. As an example we use these functions to solve the notched beam problem with three elements.

MatlabFiles\Chap7\PlaneQuad4Results.m

```
% Plane stress analysis of a notched beam
e = 3000*10^3; nu = 0.2; h = 4; q = 50;
nodes = [0, 5; 0, 12; 6, 0; 6, 5; 20, 0; 20, 12; 54, 0; 54, 12];
conn = [1, 4, 6, 2; 3, 5, 6, 4; 5, 7, 8, 6];
bx=0; by=0; alpha=0; deltaT = 0;
nel=size(conn,1); dof=2*size(nodes,1);
lmm=[];
for i=1:nel
    lm=[];
    for j=1:4
        lm=[lm, [2*conn(i,j)-1,2*conn(i,j)]];
    end
    lmm=[lmm; lm];
end
K=zeros(dof); R = zeros(dof,1);
% Generate equations for each element and assemble them.
for i=1:3
    con = conn(i,:);
    lm = lmm(i,:);
    [k, r] = PlaneQuad4Element(1, e, nu, h, alpha, deltaT, bx, by, nodes(con,:));
    K(lm, lm) = K(lm, lm) + k;
    R(lm) = R(lm) + r;
end
% Add the distributed load contributions
for i=1:2:3
    con = conn(i,:);
    lm = lmm(i,:);
    r = PlaneQuad4Load(3, -q, 0, h, nodes(con,:));
    R(lm) = R(lm) + r;
end
```

```
% Nodal solution and reactions
debc = [1,3,13,14,15,16]; ebcVals=zeros(length(debc),1);
[d, reactions] = NodalSoln(K, R, debc, ebcVals)
for i=1:3
    fprintf(1,'Results for element %3.0g \n',i)
    EffectiveStress=PlaneQuad4Results(1, e, nu, alpha, deltaT, ...
        nodes(conn(i,:),:), d(lmm(i,:)))
end
```

```
>> NotchedBeamEx

d =

            0
    -0.018316
            0
     -0.01832
    0.0027592
    -0.016649
    0.0011455
    -0.016463
      0.00305
    -0.011357
   -0.0021013
    -0.011625
            0
            0
            0
            0


reactions =

        -7932
        10361
       -19673
         5840
        17244
         4960

Results for element   1

sig =

      -104.57
       28.544
       166.98
```

```
PrincipalStresses =

  -217.77
   141.74


EffectiveStress =

   313.66

Results for element   2

sig =

  -22.085
  -19.167
  -43.656


PrincipalStresses =

  -64.306
   23.054


EffectiveStress =

   78.417

Results for element   3

sig =

    -50.6
  -43.717
   154.17


PrincipalStresses =

  -201.36
   107.05
```

EffectiveStress =

    271.22