```
% The 4th Nonliear CAE Seminar
% Exercise 2 using Newton's Method
%
% Minako Sekiguchi 11/18/03
%
%-------------------------------------------------------------------------
% Material Properties
E=5*10^9; v=0.3;
%
% Geometry of the string
L=0.5;
d=0.002;
A=pi*d^2/4;
%
% Applied forces
b=10*9.81;
P=3.1*9.81;
%
% Order of polynomials and d.o.f.
p=2;
dof=p+1;
%
%-------------------------------------------------------------------------
% Pre-processing: Create an FEM model
%-------------------------------------------------------------------------
% # of elements, # of nodes, # of total d.o.f.
nelx=10;
nx=nelx*p+1;
dx=L/(nx-1);
neq=nx*2;
%
% Element connectivity
ijk=zeros(nelx,dof);
for i=1:nelx
   for j=1:dof
      ijk(i,j)=(i-1)*p+j;
   end
end
%
% Nodal coordinates
xG=zeros(nx,2);
for i=1:nx
   xG(i,1)=(i-1)*dx;
end
%
% Plot model
%
xo=zeros(1,2); yo=zeros(1,2); icont=[1,p+1];
for nel=1:nelx
   if nel==2, hold, end
   %
   for jj=1:2
      j=ijk(nel,icont(jj));
      xo(jj)=xG(j,1);
      yo(jj)=xG(j,2);
   end
   plot(xo,yo)
   %
end
hold off
pause
```

```
%
%-------------------------------------------------------------------------
% Define boundary and loading conditions
%-------------------------------------------------------------------------
% Boundary conditions (Simply supported)
ndc=2;
dcnode=[1,1,1,0,0;
   nx,0,1,0,0];
%
% Loading conditions
fg1=zeros(nx,1);
fg2=zeros(nx,1);
%
% Tensile point load at the roller
fg1(nx,1)=P;
%
% Distributed load in the transverse direction
if p==1
   imax=2; a=1/sqrt(3);
   gwi=[1,1]; gxi=[-a,a];
elseif p==2
   imax=3; a=sqrt(3/5);
   gwi=[5,8,5]/9;gxi=[-a,0,a];
end
%
for nel=1:nelx
   %
   xge=zeros(1,2);
   ideg=zeros(1,dof);
   %
   xge(1)=xG(ijk(nel,1),1);
   xge(2)=xG(ijk(nel,1+p),1);
   %
   for i=1:dof
      ideg(i)=ijk(nel,i);
   end
   %
   fge=zeros(dof,1);
   for ig=1:imax
      gw1=gwi(ig); r=gxi(ig);
      %
      Le=xge(2)-xge(1);
      J=Le/2;
      %
      if p==1
         NF=[1-r,1+r]/2;
      elseif p==2
         NF=[r*(r-1)/2,1-r^2,r*(r+1)/2];
      end
      fge=fge+NF'*b*J*gw1;
   end
   for i=1:dof
      fg2(ideg(i))=fg2(ideg(i))+fge(i);
   end
end
%
%-------------------------------------------------------------------------
% Construct tangent stiffness matrix
%-------------------------------------------------------------------------
%
Ug=zeros(2*nx,1);
```

```
%
It=0;
%for It=1:5
%
enorm=1;
while enorm > 10^-3;
It=It+1
%
K11=zeros(nx);
K12=zeros(nx);
K22=zeros(nx);
Qg1=zeros(nx,1);
Qg2=zeros(nx,1);
%
% Element tangent stiffness matrix
for nel=1:nelx
   %
   xge=zeros(1,2);
   ideg=zeros(1,dof);
   %
   xge(1)=xG(ijk(nel,1),1);
   xge(2)=xG(ijk(nel,1+p),1);
   %
   for i=1:dof
      ideg(i)=ijk(nel,i);
   end
   %
   ue=zeros(dof,1);
   ve=zeros(dof,1);
   for i=1:dof
      ue(i)=Ug(ideg(i));
      ve(i)=Ug(nx+ideg(i));
   end
   %
   ske11=zeros(dof);
   ske12=zeros(dof);
   ske22=zeros(dof);
   qe1=zeros(dof,1);
   qe2=zeros(dof,1);
   %
   for ig=1:imax
      gw1=gwi(ig); r=gxi(ig);
      %
      Le=xge(2)-xge(1);
      J=Le/2;
      %
      if p==1
         NF=[1-r,1+r]/2;
         B=[-1,+1]/2/J;
      elseif p==2
         NF=[r*(r-1)/2,1-r^2,r*(r+1)/2];
         B=[r-1/2,-2*r,r+1/2]/J;
      end
      %
      Nxp=E*A*((B*ue)+(B*ve)^2/2);
      %
      ske11=ske11+B'*(Nxp+E*A*(1+B*ue)^2)*B*J*gw1;
      ske12=ske12+B'*E*A*(1+B*ue)*(B*ve)*B*J*gw1;
      ske22=ske22+B'*(Nxp+E*A*(B*ve)^2)*B*J*gw1;
      %
      qe1=qe1+B'*Nxp*(1+B*ue)*J*gw1;
```

```
      qe2=qe2+B'*Nxp*(B*ve)*J*gw1;
   end
   %
   % Assembly
   for i=1:dof
      Qg1(ideg(i))=Qg1(ideg(i))+qe1(i);
      Qg2(ideg(i))=Qg2(ideg(i))+qe2(i);
      for j=1:dof
         K11(ideg(i),ideg(j))=K11(ideg(i),ideg(j))+ske11(i,j);
         K12(ideg(i),ideg(j))=K12(ideg(i),ideg(j))+ske12(i,j);
         K22(ideg(i),ideg(j))=K22(ideg(i),ideg(j))+ske22(i,j);
      end
   end
end
%
if It==1
   % Initial guess (linear tensile solution)
   %
   % Apply boundary condition
   kdeg=linspace(1,nx,nx);
   bdeg=[];
   k=0;
   for i=1:ndc
      if dcnode(i,2) ~= 0
         k=k+1;
         bdeg(k)=dcnode(i,1);
         kdeg(bdeg(k))=0;
      end
   end
   %
   kdeg(bdeg)=[];
   K11(bdeg,:)=[];
   K11(:,bdeg)=[];
   fg=fg1;
   fg(bdeg)=[];
   %
   % Solve linear equations
   ug=inv(K11)*fg;
   for i=1:nx-k
      Ug(kdeg(i))=ug(i);
   end
   % Error norm (energy)
   enorm=fg'*ug
   %
   %
else
   %
   skg=[K11,K12;K12',K22];
   fg=[fg1-Qg1;fg2-Qg2];
   %
   % Apply boundary conditions
   kdeg=linspace(1,neq,neq);
   bdeg=[];
   k=0;
   for i=1:ndc
      for j=1:2
         if dcnode(i,j+1) ~= 0
            k=k+1;
            bdeg(k)=mod(j+1,2)*nx+dcnode(i,1);
            kdeg(bdeg(k))=0;
         end
```

```
      end
   end
   %
   kdeg(bdeg)=[];
   skg(bdeg,:)=[];
   skg(:,bdeg)=[];
   fg(bdeg)=[];
   %
   % Solve linearized equations for increments
   ug=inv(skg)*fg;
   for i=1:neq-k
      Ug(kdeg(i))=Ug(kdeg(i))+ug(i);
   end
   % Error norm
   %norm(ug)
   enorm=fg'*ug
   %
   %-------------------------------------------------------------------------
   % Plot model
   %
   xo=zeros(1,dof); yo=zeros(1,dof);
   xp=zeros(1,dof); yp=zeros(1,dof); icont=linspace(1,dof,dof);
   %
   for nel=1:nelx
      if nel==2, hold, end
      %
      for jj=1:dof
         j=ijk(nel,icont(jj));
         xo(jj)=xG(j,1);
         yo(jj)=xG(j,2);
         xp(jj)=xG(j,1)+Ug(j);
         yp(jj)=xG(j,2)+Ug(nx+j);
      end
      plot(xo,yo,':',xp,yp,'o',xp,yp)
      %
   end
   axis equal
   %axis off
   hold off
   pause
   %
   if It > 20, enorm=0; 'Non-convergent after 20 iterations!', end
   %
end
%
end
%
%-------------------------------------------------------------------------
% Green-Lagrange Strain in each element
Ex=zeros(nelx,2);
for nel=1:nelx
   xge=zeros(2,1);
   ideg=zeros(1,dof);
   %
   xge(1)=xG(ijk(nel,1),1);
   xge(2)=xG(ijk(nel,1+p),1);
   %
   for i=1:dof
      ideg(i)=ijk(nel,i);
   end
   %
```

```
    ue=zeros(dof,1);
    ve=zeros(dof,1);
    for i=1:dof
       ue(i)=Ug(ideg(i));
       ve(i)=Ug(nx+ideg(i));
    end
    %
    r=0; %Strain evaluated at element center
    %
    Le=xge(2)-xge(1);
    J=Le/2;
    %
    if p==1
       NF=[1-r,1+r]/2;
       B=[-1,+1]/2/J;
    elseif p==2
       NF=[r*(r-1)/2,1-r^2,r*(r+1)/2];
       B=[r-1/2,-2*r,r+1/2]/J;
    end
    % Green-Lagrange Strain Ex
    Ex(nel,1)=B*ue+((B*ue)^2+(B*ve)^2)/2;
    %
    % Strain calculated from change in element length: Ex=(ds^2-dS^2)/dS^2/2
    ds=norm([Le+ue(dof)-ue(1),ve(dof)-ve(1)]);
    dS=Le;
    Ex(nel,2)=(ds^2-dS^2)/dS^2/2;
end
```