NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]
ISSN: 2394-3696
VOLUME 2, ISSUE 5 MAY-2015

Development of Matlab Programme to study nonlinear vibration of a cantilever beam

Prof. Vyankatesh S. Kulkarni
Department of Mechanical Engineering, Solapur University /BIT/Barshi/India

Abstract

The finite element model for the nonlinear transverse vibration of the beam was implemented in the program *NLB* This program was used to calculate the steady state response of the beam. The numerical results from the program were compared to the experimental results obtained by Malatkar (2003). The *NLB* program was also used to calculate the transient response of the beam, which was compared with the response obtained with ANSYS.

The paper is a extension of the research papers which has been published earlier in this issue of journal and December 2014 issue.

NLB Matlab Program

```
%Nonlinear Vibration of a Cantilever Beam %Written by Ivan
Delgado-Velazquez
L= 0.662; W= 0.01271; tk= 5.5e-4; %beam dimensions rho= 7400; E=
165.5e9; %material properties
Alpha1= 0.064; Alpha2= 4.72e-5; %proportional damping coefficients N= 20; h= L/N;
numX=10; deltax= h/numX; %elements variables TOL=0.01; %convergence criterion for
nonlinear loop
deltat= 0.001; tf=3; k= tf/deltat; %time variables g=9.81; ab= 2.97*g;
OMEGA= 17.547; %force variables Area= W*tk; I= 1/12*W*tk^3; %section
properties %shape functions and derivatives
xs=0:deltax:h;
PSI(:,1)= 1 - (3/h^2)*xs.^2 + (2/h^3)*xs.^3; PSI(:,2)= h*((xs/h) -
(2/h^2)*xs.^2 + (1/h^3)*xs.^3); PSI(:,3)= (3/h^2)*xs.^2 - (2/h^3)*xs.^3;
PSI(:,4)= h*(-(1/h^2)*xs.^2 + (1/h^3)*xs.^3); PSIP(:,1)= -
6*(xs/h^2) + (6/h^3)*xs.^2; PSIP(:,2)= 1 - 4*(xs/h) +
(3/h^2)*xs.^2; PSIP(:,3)= 6*(xs/h^2) - (6/h^3)*xs.^2; PSIP(:,4)= -
2*(xs/h) + (3/h^2)*xs.^2; xs4=0:deltax/4:h;
PSIP_f2_4(:,1)= -6*(xs4/h^2) + (6/h^3)*xs4.^2; %FOR EVALUATION OF f2 ONLY PSIP_f2_4(:,2)= 1 -
4*(xs4/h) + (3/h^2)*xs4.^2;
PSIP_f2_4(:,3)= 6*(xs4/h^2) - (6/h^3)*xs4.^2; PSIP_f2_4(:,4)= -
2*(xs4/h) + (3/h^2)*xs4.^2; PSIPP(:,1)= -6/h^2 + (12/h^3)*xs;
PSIPP(:,2)= -4/h + (6/h^2)*xs;
PSIPP(:,3)= 6/h^2 - (12/h^3)*xs; PSIPP(:,4)= -2/h + (6/h^2)*xs;
PSIPPP(:,1)= (12/h^3)*ones(1,(h/deltax)+1); PSIPPP(:,2)=
(6/h^2)*ones(1,(h/deltax)+1);
```

ISSN: 2394-3696
VOLUME 2, ISSUE 5 MAY-2015

```
PSIPPP(:,3)= -(12/h^3)*ones(1,(h/deltax)+1);
PSIPPP(:,4)= (6/h^2)*ones(1,(h/deltax)+1);
Me= (rho*Area*h/420)*[156 22*h 54 -13*h; 22*h 4*h*h 13*h -3*h*h; 54 13*h 156 -22*h;
                  -13*h -3*h*h -22*h 4*h*h];
Ke= (E*I/h^3)*[12 6*h -12 6*h; 6*h 4*h*h -6*h 2*h*h; -12 -6*h 12 -6*h;
            6*h 2*h*h -6*h 4*h*h];
Ce= Alpha1*Me + Alpha2*Ke; %element matrices - [M], [KL], [C] Gamma= 0.5;
Beta= 0.25; %Newmark coefficients
epsilon= 0.5-2*Beta + Gamma;     PHI= rho*Area*ab;
delta= 0.5 + Beta - Gamma;       THETA= 2*pi*OMEGA;
%global matrices - [M], [KL], [C] z= 2*(N+1);

Mg= zeros(z); Kg= zeros(z); Cg= zeros(z); x1= 1;
for x=1:N a=1;
   for i=x1:x1+3 b=1;
      for j=x1:x1+3 Mg(i,j)=Mg(i,j)+ Me(a,b); Kg(i,j)=Kg(i,j)+ Ke(a,b);
         Cg(i,j)=Cg(i,j)+ Ce(a,b); b= b+1;
      end
      a= a+1; end
   x1= x1 +2; end
%reduced global matrices for i=1:z-2
   for j=1:z-2
      MgR(i,j)= Mg(i+2,j+2);
      KgR(i,j)= Kg(i+2,j+2);
      CgR(i,j)= Cg(i+2,j+2); end
end
%Newmark matrices - LINEAR
A1= MgR + Gamma*deltat*CgR + Beta*deltat^2*KgR;
A2= -2*MgR + (1-2*Gamma)*deltat*CgR + epsilon*deltat^2*KgR; A3= MgR - (1-Gamma)*deltat*CgR
+ delta*deltat^2*KgR; %Force discretization vector
FF= zeros(z,1); a=0;
for ll=1:N for aa=1:4
      S1=0; S2=0;
      for l=1:(h/deltax +1) alpha(l)= PSI(l,aa);
      end
for i1=2:2:h/deltax S1= S1 + alpha(i1);
      end
      for j1=3:2:h/deltax -1 S2= S2 + alpha(j1);
      end
      SS(aa)=(deltax/3)*(alpha(1)+ 4*S1+2*S2+ alpha(h/deltax+1)); end
   for i1=1:4
```

```
      FF(i1+a)= FF(i1+a) + SS(i1); end
    a= a+2;
    for i1=1:z-2 FD(i1,1)= FF(i1+2);
    end end
UL(:,1)= zeros(z,1); %from IC time= 0:deltat:tf; %time vector %Main
loop
for j=1:k %Linear loop for o=1:z-2
      for p=1:j
         ULr(o,p)= UL(o+2,p); end
    end
    if j==1
      U0= ULr(:,1); U1= ULr(:,1); F0= zeros(z-2,1);
      FLin(:,j)= PHI*FD*cos(THETA*time(j)); F1= FLin(:,j);
    else
      U0= ULr(:,j-1); U1= ULr(:,j);
      F0= FLin(:,j-1); F1= FLin(:,j); end
    FLin(:,j+1)= PHI*FD*cos(THETA*time(j+1)); F2= FLin(:,j+1);
    F= Beta*F2 + epsilon*F1 + delta*F0;
    U2= -inv(A1)*A2*U1 - inv(A1)*A3*U0 + deltat^2*inv(A1)*F; ULr(:,j+1)= U2;
    for i=1:z
 for p=1:j+1 if i<= 2
          UL(i,p)=0; else
          UL(i,p)= ULr(i-2,p); end
      end end
    %Nonlinear loop eps= 10^5; counter= 0; while eps > TOL
      %first nonlinear stiffness matrix a1= 1; b0= 0;
      for o=1:N

    for p= 1:(h/deltax)+1
      WWP(p)= PSIP(p,1)*UL(a1,j+1) + PSIP(p,2)*UL(a1+1,j+1)+...
         PSIP(p,3)*UL(a1+2,j+1) + PSIP(p,4)*UL(a1+3,j+1); WWPP(p)= PSIPP(p,1)*UL(a1,j+1)+
      PSIPP(p,2)*UL(a1+1,j+1)+...
         PSIPP(p,3)*UL(a1+2,j+1) + PSIPP(p,4)*UL(a1+3,j+1); WWPPP(p)=
      PSIPPP(p,1)*UL(a1,j+1)+ PSIPPP(p,2)*...
         UL(a1+1,j+1)+PSIPPP(p,3)*UL(a1+2,j+1) + ...
         PSIPPP(p,4)*UL(a1+3,j+1); WP(p+b0)= WWP(p);
      WPP(p+b0)= WWPP(p); WPPP(p+b0)= WWPPP(p);
    end
    a1= a1+2;
    b0= b0 + h/deltax; end
AA= N*(h/deltax + 1)- N +1; AA_f2_2= N*(2*h/deltax + 1)- N +1;
AA_f2_4= N*(4*h/deltax + 1)- N +1; f1= WP.*WPPP + WPP.^2;
f1(AA)= 0; %from BC
aaa=0; bbb=0; mmm=0; KNL1=zeros(z); for o=1:N
    for pp=1:4 for qq=1:4
```

```
          S1=0; S2=0;
          for l=1:(h/deltax)+1
             PsiPijF1(l)= PSIP(l,pp)*PSIP(l,qq)*f1(mmm+l); end
          for ii=2:2:(h/deltax) S1= S1 + PsiPijF1(ii);
          end
for ii=3:2:(h/deltax - 1) S2= S2 + PsiPijF1(ii);
          end
          KKNL(pp,qq)= (deltax/3)*(PsiPijF1(1)+ 4*S1+2*S2 + ...
             PsiPijF1(h/deltax+1));
          KNL1(pp+aaa,qq+bbb)= KNL1(pp+aaa,qq+bbb)+ KKNL(pp,qq); end
   end
   aaa= aaa+2; bbb= bbb+2; mmm= mmm + h/deltax; end
KNL1= E*I*KNL1; %first nonlinear stiffness matrix for ii=1:z-2
   for jj=1:z-2
     KNL1r(ii,jj)= KNL1(ii+2,jj+2); %reduced KNL1 end
end
%second nonlinear stiffness matrix if j == 1
   WPsq2Dot= Wpsq2Dot_approx_SS; %approx. with lin. disp. else
   a1= 1; b0= 0; for o=1:N
      for p= 1:(4*h/deltax)+1
         wWP(p,3)= PSIP_f2_4(p,1)*UL(a1,j+1)+PSIP_f2_4(p,2)*...
            UL(a1+1,j+1)+PSIP_f2_4(p,3)*UL(a1+2,j+1) +...
            PSIP_f2_4(p,4)*UL(a1+3,j+1);
         wWP(p,2)= PSIP_f2_4(p,1)*UL(a1,j) + PSIP_f2_4(p,2)*...
            UL(a1+1,j)+PSIP_f2_4(p,3)*UL(a1+2,j) +...
            PSIP_f2_4(p,4)*UL(a1+3,j);
         wWP(p,1)= PSIP_f2_4(p,1)*UL(a1,j-1)+PSIP_f2_4(p,2)*...
            UL(a1+1,j-1)+PSIP_f2_4(p,3)*UL(a1+2,j-1) +...
            PSIP_f2_4(p,4)*UL(a1+3,j-1);
         wP(p+b0,3)= wWP(p,3); wP(p+b0,2)= wWP(p,2); wP(p+b0,1)= wWP(p,1);
      end
      a1= a1+2;
      b0= b0 + 4*h/deltax; end
   for ii=1:AA_f2_4 for jj=1:3
         WPsq(ii,jj)= wP(ii,jj)*wP(ii,jj); end
   end
   for ii=1:AA_f2_4
      WPsqDot(ii,1)= (WPsq(ii,2)-WPsq(ii,1))/deltat;
      WPsqDot(ii,2)= (WPsq(ii,3)-WPsq(ii,2))/deltat; WPsq2Dot(ii)= (WPsqDot(ii,2)-
      WPsqDot(ii,1))/deltat;
   end end xx1=1;
for xx=1:2:AA_f2_4 S1=0; S2=0; i1=2; j1=3; while i1<=
   xx-1
      S1= S1 + WPsq2Dot(i1); i1= i1 + 2;
   end
```

ISSN: 2394-3696
VOLUME 2, ISSUE 5 MAY-2015

```
   while j1<= xx-2
      S2= S2 + WPsq2Dot(j1); j1= j1 + 2;
   end
   SS1(xx1)=(deltax/12)*(WPsq2Dot(1)+4*S1 + 2*S2 + WPsq2Dot(xx));


   xx1=xx1+1; end SS1(1)=0; xx2=1;

for xx=1:2:AA_f2_2
   S1=0; S2=0; i1=xx+1; j1=xx+2; while i1<= AA_f2_2 - 1
      S1= S1 + SS1(i1); i1= i1 + 2;
   end
   while j1<= AA_f2_2 - 2 S2= S2 + SS1(j1); j1= j1 + 2;
   end
   f2(xx2)=(-deltax/6)*(SS1(xx) + 4*S1 + 2*S2 + SS1(AA_f2_2)); xx2=xx2+1;
end f2(AA)= 0;
aaa=0; bbb=0; mmm=0; KNL2=zeros(z); for o=1:N
   for pp=1:4 for qq=1:4
         S1=0; S2=0;
         for l=1:(h/deltax)+1
PsiPijF2(l)= PSIP(l,pp)*PSIP(l,qq)*f2(mmm+l); end
         for ii=2:2:(h/deltax) S1= S1 + PsiPijF2(ii);
         end
         for ii=3:2:(h/deltax - 1) S2= S2 + PsiPijF2(ii);
         end
         KKnl(pp,qq)= (deltax/3)*(PsiPijF2(1)+ 4*S1+2*S2 + ...
            PsiPijF2(h/deltax+1));
         KNL2(pp+aaa,qq+bbb)= KNL2(pp+aaa,qq+bbb)+ KKnl(pp,qq); end
   end
   aaa= aaa+2; bbb= bbb+2; mmm= mmm + h/deltax; end
KNL2= 0.5*rho*Area*KNL2; %second nonlinear stiffness matrix for ii=1:z-2
   for jj=1:z-2
     KNL2r(ii,jj)= KNL2(ii+2,jj+2); %reduced KNL2 end
end
KTotal= KgR-KNL1r-KNL2r; %total stiffness matrix

CNL= Alpha1*MgR + Alpha2*KTotal; %nonlinear damping matrix %nonlinear displacement -
Newmark technique
for o=1:z-2 for p=1:j
      UNLr(o,p)= UL(o+2,p); end
end
if j==1
   U0= UNLr(:,1); U1= UNLr(:,1); F0= zeros(z-2,1);
   FNLin(:,j)= FLin(:,j); F1= FNLin(:,j);
else
```

```
   U0= UNLr(:,j-1); U1= UNLr(:,j);
   F0= FNLin(:,j-1); F1= FNLin(:,j); end
%Newmark matrices - NONLINEAR
A1NL= MgR + Gamma*deltat*CNL + Beta*deltat^2*KTotal;
A2NL= -2*MgR + (1-2*Gamma)*deltat*CNL + epsilon*deltat^2*KTotal; A3NL= MgR - (1-
Gamma)*deltat*CNL + delta*deltat^2*KTotal;
%total force vector
X= 0:deltax:L; %global X vector for ii=1:AA
   f3(ii)= (X(ii)*deltax - L)*WPP(ii) + WP(ii); end
G= zeros(z,1); a=0; m=0; for ll=1:N
   for aa=1:4 S1=0; S2=0;
for l=1:(h/deltax +1) alpha(l)= PSI(l,aa)*f3(m+l);
      end
      for i1=2:2:h/deltax S1= S1 + alpha(i1);
      end
      for j1=3:2:h/deltax -1 S2= S2 + alpha(j1);
      end
      SS(aa)=(deltax/3)*(alpha(1)+ 4*S1+2*S2 + ...
         alpha(h/deltax+1));
   end
   for i1=1:4
      G(i1+a)= G(i1+a) + SS(i1); end
   a= a+2; m= m+h/deltax;
      end
      G= rho*Area*g*G; %gravity vector for i1=1:z-2
         Gr(i1)= G(i1+2); end
      FNLin(:,j+1)= FLin(:,j+1)+ Gr'; %nonlinear force vector F2= FNLin(:,j+1);
      F= Beta*F2 + epsilon*F1 + delta*F0;
      U2= -inv(A1NL)*A2NL*U1 - inv(A1NL)*A3NL*U0 + deltat^2*inv(A1NL)*F; UNLr(:,j+1)= U2;
      for i=1:z
         for p=1:j+1 if i<= 2
               UNL(i,p)=0; else
               UNL(i,p)= UNLr(i-2,p); end
         end end
      for ii=1:z
         DELTA(ii)= abs(UL(ii,j+1) - UNL(ii,j+1)); end
      eps= sum(DELTA); UL(:,j+1)= UNL(:,j+1);
      counter=counter +1;
   end
   kounter(j)=counter; %number of iterations in NL loop EPS(j)= eps;
   %convergence variable for each time step time(j)
end
%time response plots
Response_base= UL(3,:); %response of base (2nd node) Response_tip=
UL(z-1,:); %response of tip (last node) figure
```

ISSN: 2394-3696
VOLUME 2, ISSUE 5 MAY-2015

```
plot(time,Response_base)
xlabel('Time (s)'), ylabel('Base Response (m)')
title('Time response of cantilever beam (x=33.1mm)') %figure 1 figure
plot(time,Response_tip)
xlabel('Time (s)'), ylabel('Tip Response (m)')
title('Time response of cantilever beam (x=66.2mm)') %figure 2 %Fast Fourier
Transform (FFT)
Trecord= 3; %length of time record
TI= tf- Trecord; %initial time of sampling TF= tf; %final time
of sampling
Ta= TI/deltat +1; %element number corresponding to TI
Tb= TF/deltat +1; %element number corresponding to TF SR= 6; %sampling rate
DELTAT= SR*deltat; %sampling interval fs= 1/(DELTAT); %sampling
frequency ii=1;
for i=Ta:SR:Tb
   Response_baseS(ii)= Response_base(i); %sampled points for FFT Response_tipS(ii)= Response_tip(i);
   %sampled points for FFT ii=ii+1;
end
Yb= fft(Response_baseS,512); FB= Yb.*conj(Yb);
Yt= fft(Response_tipS,512); FT= Yt.*conj(Yt);
figure
freq = fs*(0:256)/512; plot(freq,FT(1:257))
xlabel('frequency (Hz)'), ylabel('Tip FFT') title('Frequency content of UNL-tip')
%figure 3 figure
freq = fs*(0:256)/512; plot(freq,FB(1:257))
xlabel('frequency (Hz)'), ylabel('Base FFT') title('Frequency content of UNL-base')
%figure 4
```

ISSN: 2394-3696
VOLUME 2, ISSUE 5 MAY-2015

Conclusion

In summary, the numerical results obtained with *NLB* differ from the experimental results (Malatkar, 2003) due to the presence of numerical error in the former. The transient response calculated with *NLB* agrees with the response calculated

With ANSYS® for the most part.

Future Work

The program *NLB* needs to be streamlined to reduce computation time of the steady state response. Also, the numerical error in the calculation of the nonlinear inertia term should be improved by using alternate numerical methods for its calculation.

References

1. Reddy, J.N., *An Introduction to the Finite Element Method*, McGraw-Hill, New York, 1993
2. Zienkiewicz, O.C., *The Finite Element Method*, McGraw-Hill, London, 1977
3. Zill, D.G., et al., *Differential Equations with Boundary Value Problems,* Brooks/Cole Publishing Company, New York, 1997
4. *A review of nonlinear flexural-torsional vibration of a cantilever beam* ISSN: 2394-3696 VOLUME 1, ISSUE 2 DEC-2014
5. *Study of numerical algorithm used to solve the equation of motion for the planar flexural forced vibration of the cantilever beam* ISSN: 2394-3696 VOLUME 1, ISSUE 2 DEC-2014