

```

%-----
% Copyright (c) 2016, Anna Liakou
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
% met:
%
% * Redistributions of source code must retain the above copyright
% notice, this list of conditions and the following disclaimer.
% * Redistributions in binary form must reproduce the above copyright
% notice, this list of conditions and the following disclaimer in
% the documentation and/or other materials provided with the distribution
%
% THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
% AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
% IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
% ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
% LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
% CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
% SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
% INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
% CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
% ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
% POSSIBILITY OF SUCH DAMAGE.
% -----
% HeMo evaluates the dynamic response of a cantilever beam against two
% symmetrical stops ((end)-point and/or horizontal wall)
% The impact motion is simulated by two methods:
% 1. Moreau's midpoint method
% 2. Moreau-Jean's theta method
%-----
%%
clear all
close all
%% Index_method & Index_contact
%-----
% Index_method=1 : Moreau's midpoint method
% Index_method=2 : Moreau-Jean's theta method
Index_method=1;
% Index_contact=1 : Point contact
% Index_contact=2 : Horizontal wall
Index_contact=1;
%% theta (if Moreau-Jean's theta method is used)
theta=1;
%% Choice of coefficient of restitution and penalty parameter
%-----
e=0.5; % Coefficient of restitution
r=0.001; % penalty parameter
error_tol=1e-7; % error tolerance at contact constraint iteration
%% Number of segments
%-----
num_seg=50;
num_var=num_seg; % Number of nodes = number of elements for
cantilever beam
% ( The first node is removed enforcing the BC)
%% geometric properties:

```

```

%-----
tot_length=10;           % Total length
tot_len=1.0;            % Scaled length::
len=tot_len/num_seg;    % Dx of element
density=8000;           % Density of the beam
radius=0.005;           % Radius of the beam
thickness=0.0005;       % Thickness
I=pi*(radius^4)/4-pi*((2*radius-2*thickness)^4)/64; % Moment of inertia
Area=pi*(radius^2-(radius-thickness)^2); % Area of the beam
E=2e11;                 % Modulus of Elasticity
%% Mass and stiffness (scaled)
%-----
mass=(len/tot_len)/6;   % Translational mass
massr=(tot_len/len)*(I/(Area*(tot_length^2)))/6; % Rotational mass
stiffness=(tot_len/len)^3; % stiffness n^3
%% Scaled force, displacement and time::
%-----
f_star=1;force=1;      % scaled force
U_star=(tot_length^4)*f_star/(E*I); % U*=l^4*q/EI
Period=(tot_length^2)/sqrt(E*I/(density*Area)); % t*=L^2*sqrt(EI/pA)
omega=10*Period;       % Scaled frequency of applied force
%% Time step
hstep=0.01*len;        % Constant ratio Dt/Dx=0.01
%% Contact constraint
g_fin=0.1/U_star;      % gap function
%% Initialize variables::
%-----
M=zeros(num_var,num_var); % total mass matrix
M_t=zeros(num_var,num_var); % translational mass matrix
M_r=zeros(num_var,num_var); % rotational mass matrix
K=zeros(num_var,num_var); % stiffness matrix
% Initialize displacement - velocity and force
x=zeros(num_var,1);xold=zeros(num_var,1);fold=zeros(num_var,1);
v=zeros(num_var,1);vold=zeros(num_var,1);v_smooth=zeros(num_var,1);
f=zeros(num_var,1);
%% Mass matrices and Stiffness matrices::
%-----
for i=1:num_var
    if i==1
        M_t(i,i)=4;M_t(i,i+1)=1;
        M_r(i,i)=6;M_r(i,i+1)=-2;M_r(i,i+2)=-1;
        K(i,i)=6;K(i,i+1)=-4;K(i,i+2)=1;
    elseif i==2
        M_t(i,i)=4;M_t(i,i+1)=1; M_t(i,i-1)=1;
        M_r(i,i)=6;M_r(i,i+1)=-2;M_r(i,i-1)=-2; M_r(i,i+2)=-1;
        K(i,i)=6;K(i,i+1)=-4;K(i,i-1)=-4;K(i,i+2)=1;
    elseif i==num_var
        M_t(i,i)=2; M_t(i,i-1)=1;
        M_r(i,i)=2;M_r(i,i-1)=-1;M_r(i,i-2)=-1;
        K(i,i)=1;K(i,i-1)=-2;K(i,i-2)=1;
    elseif i==num_var-1
        M_t(i,i)=4;M_t(i,i+1)=1; M_t(i,i-1)=1;
        M_r(i,i)=4;M_r(i,i+1)=-1;M_r(i,i-1)=-2;M_r(i,i-2)=-1;
        K(i,i)=5;K(i,i+1)=-2;K(i,i-1)=-4;K(i,i-2)=1;
    else
        M_t(i,i)=4;M_t(i,i+1)=1; M_t(i,i-1)=1;
        M_r(i,i)=6;M_r(i,i+1)=-2;M_r(i,i-1)=-2;M_r(i,i-2)=-1;M_r(i,i+2)=-1;

```

```

        K(i,i)=6;K(i,i+1)=-4;K(i,i-1)=-4;K(i,i-2)=1;K(i,i+2)=1;
    end

end

if Index_method==1          % Moreau's midpoint method
%% Total mass and inverse::
%-----
M=mass*M_t+massr*M_r;
Minv=inv(M);
M_K=stiffness*Minv*K;
%% Constant matrices used at the numerical scheme::
%-----
A_1=hstep^2*M_K+eye(num_var);
A_1=inv(A_1);
A_2=A_1*Minv;
A_3=A_2*stiffness*K;
%% Total time::
%-----
tot=floor(2/hstep/Period);yplot=zeros(tot,num_var+1);
    if Index_contact==1
        %% Point contact
        for k=1:tot
            time(k)=hstep*(k-1);
            time_n=time(k)+hstep;
            f(1)=len*force*sin(omega*(hstep/2+time(k))); % applied force at midpoint
time
            f(:)=f(1);
            yplot(k,num_var+1)=x(num_var);vold=v;
            xmid=x+hstep*v/2; % displacement at midpoint
            v=A_1*v+hstep*A_2*f-hstep*A_3*xmid; % update of velocity for smooth motion
            if xmid(num_var)>=g_fin % If violation of contact constraint
                error=1; % Initialize error
                v_smooth=v; % Smooth part of velocity update
                lamdn=zeros(num_var,1);ln=0; % Initialize Impulse
                %% Iteration for the contact problem
                %-----
                while error>error_tol
                    Du=Minv*lamdn; % Impact equation
                    v=v_smooth+Du;
                    gna=vold(num_var);
                    gne=v(num_var);
                    ksin=gne+e*gna;
                    lnpr=ln;
                    ln=min(ln-r*ksin,0); % projective equation
                    lamdn(num_var)=ln;
                    error=(abs(ln-lnpr));
                end
                elseif xmid(num_var)<=-g_fin % If violation of contact constraint
                    error=1; % Initialize error
                    v_smooth=v; % Smooth part of velocity update
                    lamdn=zeros(num_var,1);ln=0;
                    while error>error_tol
                        Du=Minv*lamdn; % Impact equation
                        v=v_smooth+Du;
                        gna=vold(num_var);
                        gne=v(num_var);
                        ksin=gne+e*gna;

```

```

lnpr=ln;
ln=max(ln-r*ksin,0); % projective equation
lamdn(num_var)=ln;
error=abs(ln-lnpr);
end
end
%-----
x=xmid+hstep*v/2; % Update of displacement
end
else
%% Horizontal wall
for k=1:tot
time(k)=hstep*(k-1);
time_n=time(k)+hstep;
f(1)=len*force*sin(omega*(hstep/2+time(k)));
f(:)=f(1);
yplot(k,num_var/2+1)=x(num_var/2);
xmid=x+hstep*v/2; % Displacement at midpoint (used for contact
constraint)
v=A_1*v+hstep*A_2*f-hstep*A_3*xmid;
if any(xmid(:)-g_fin>0) || any(xmid(:)+g_fin<0) % If violation of contact
constraint
error=1; % Initialize error
v_smooth=v; % Smooth part of velocity update
%% Initialize Impulse::
lamdn=zeros(num_var,1);ln=zeros(num_var,1);lnpr=zeros(num_var,1);
%% Index set::
%-----
indexg=[];
mm=1;
for j=1:num_var
if xmid(j)>=g_fin || xmid(j)<=-g_fin
indexg(mm)=j;
mm=mm+1;
flag_con=1;
end
end
%% Iteration for the contact problem
%-----
while error>error_tol %&& iter<20
for i=1:mm-1
var=indexg(i);
Du=Minv*lamdn; % Impact equation
v=v_smooth+Du;
gna=vold(var);
gne=v(var);
ksin=gne+e*gna;
lnpr(var)=ln(var);
if x(var)>0
ln(var)=min(ln(var)-r*ksin,0); % projective equation
lamdn(var)=ln(var);
else
ln(var)=max(ln(var)-r*ksin,0); % projective equation
lamdn(var)=ln(var);
end
end
error=abs(norm((ln-lnpr)));

```

```

        end
    end
    %-----
    x=xmid+hstep*v/2;    % Update of displacement
end
end
else % Moreau-Jean's theta method

%% Total mass and inverse::
%-----
M=mass*M_t+massr*M_r;
M_n=M+(hstep^2)*(theta^2)*K*stiffness;
Minv_n=inv(M_n);
A_M=-Minv_n*K*stiffness;fac=(hstep*theta+(hstep^2)*(theta^2));
%% Total time::
%-----
tot=floor(2/hstep/Period);
    if Index_contact==1
        %% Point contact
        for k=1:tot
            time(k)=hstep*(k-1);
            time_n=time(k)+hstep;
            f_p(1)=len*force*sin(omega*(time(k)));
            f_p(:)=f_p(1);
            f(1)=len*force*sin(omega*(hstep+time(k)));
            f(:)=f(1);
            yplot(k,num_var+1)=x(num_var);
            xmid=x+hstep*v;          % Displacement at midpoint (used for contact
constraint)
            vold=v;
            v=v+A_M*(hstep^2*theta*v+hstep*x)+hstep*Minv_n*(theta*f+(1-theta)*f_p);
            %% Iteration for the contact problem
            %-----
            if xmid(num_var)>=g_fin % If violation of contact constraint
                error=1;
                %% Initialize Impulse::
                lamdn=zeros(num_var,1); ln=0;lnpr=0;
                v_smooth=v;          % Smooth part of velocity update
                while error>error_tol
                    Du=Minv_n*lamdn; % Impact equation
                    v=v_smooth+Du;
                    gna=vold(num_var);
                    gne=v(num_var);
                    ksin=gne+e*gna;
                    lnpr=ln;
                    ln=min(ln-r*ksin,0); % projective equation
                    lamdn(num_var)=ln;
                    error=(abs(ln-lnpr));
                end
                elseif xmid(num_var)<=-g_fin % If violation of contact constraint
                    error=1;
                    %% Initialize Impulse::
                    lamdn=zeros(num_var,1);ln=0;lnpr=0;
                    v_smooth=v;
                    while error>error_tol
                        Du=Minv_n*lamdn; % Impact equation
                        v=v_smooth+Du;

```

```

gna=vold(num_var);
gne=v(num_var);
ksin=gne+e*gna;
lnpr=ln;
ln=max(ln-r*ksin,0);
lamdn(num_var)=ln;
error=abs(ln-lnpr);
    end
end
%-----
x=x+hstep*(theta*v+(1-theta)*vold); % Update of displacement
end
else
%% Horizontal wall
    for k=1:tot
time(k)=hstep*(k-1);
time_n=time(k)+hstep;
f_p(1)=len*force*sin(omega*(time(k)));
f_p(:)=f_p(1);
f(1)=len*force*sin(omega*(hstep+time(k)));
f(:)=f(1);
yplot(k,num_var/2+1)=x(num_var/2);
xmid=x+hstep*v; % Displacement at midpoint (used for contact
constraint)
vold=v;
v=v+A_M*(hstep^2*theta*v+hstep*x)+hstep*Minv_n*(theta*f+(1-theta)*f_p);
    if any(xmid(:)-g_fin>0) || any(xmid(:)+g_fin<0) % If violation of contact
constraint
        error=1; % Initialize error
        v_smooth=v; % Smooth part of velocity update
        lamdn=zeros(num_var,1);ln=zeros(num_var,1);lnpr=zeros(num_var,1);
%% Index set::
%-----
        indexg=[];
        mm=1;
        for j=1:num_var
            if xmid(j)>=g_fin || xmid(j)<=-g_fin
                indexg(mm)=j;
                mm=mm+1;
                flag_con=1;
            end
        end
%% Iteration for the contact problem
%-----
        while error>error_tol
            for i=1:mm-1
                var=indexg(i);
                Du=Minv_n*lamdn; % Impact equation
                v=v_smooth+Du;
                gna=vold(var);
                gne=v(var);
                ksin=gne+e*gna;
                lnpr(var)=ln(var);
                if x(var)>0
                    ln(var)=min(ln(var)-r*ksin,0); % projective equation
                end
                lamdn(var)=ln(var);
            end
        end
    end
end

```

```
ln(var)=max(ln(var)-r*ksin,0); % projective equation
lamdn(var)=ln(var);
end
end
error=abs(norm((ln-lnpr)));
end
end
%-----
x=x+hstep*(theta*v+(1-theta)*vold);
end
end
end
if Index_contact==1
    plot(time,yplot(:,num_seg+1))
else
    plot(time,yplot(:,num_seg/2+1))
end
ylabel('y')
xlabel('Time')
```