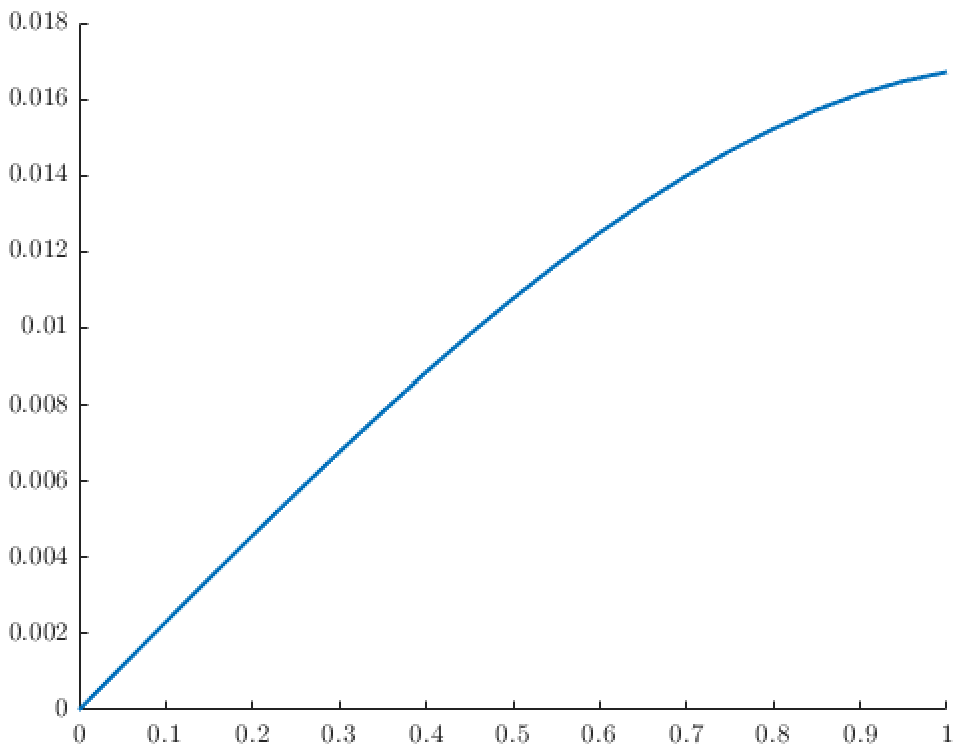
```
%AE 510 Class code for live lecture
%Author: Your instructor

set(groot,'defaulttextinterpreter','latex');
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaultLegendInterpreter','latex')
clc
clear

Len = 1;%length of the bar
Area = 0.0001; %Bar area
t = 0.01;
E = 70E9;%elastic modulus
P = 1000;%force at the middle
nelem =20;%number of elements (keep it an even number for this problem)
w = 1000; %rads/s
rho = 2700; %kg/m^3
M = 0.0270;

%%%%%%%%%%%PREPROCESSING%%%%%%%%%%
%coordinate matrix [x,y] for each node
co = [0 : Len/nelem: Len]';

%element-node connectivity matrix, length, area, modulus
for i = 1:nelem
    e(i,:) = [i,i+1];
end

Nel = size(e,1);%number of elements
Nnodes = size(co,1); %number of nodes
nne = 2; %number of nodes per element
dof = 1; %degree of freedom per node

%%%%%%%%%%%PREPROCESSING END%%%%%%%%%%


%%%Generic block: Initializes global stiffness matrix 'K' and force vector 'F'
K = zeros(Nnodes*dof,Nnodes*dof);
F = zeros(Nnodes*dof,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%Assemble Global system - generic FE code
for A = 1:Nel

    n = (co(e(A,2),:) - co(e(A,1),:));

    L = norm(n); %length of truss

    k11 = (E*Area/L);%k matrix part = EA/L*[c^2 cs;sc s^2]

    %local stiffness matrix and force vector
    localstiffness = [k11 -k11;-k11 k11];    %full local stiffness matrix
```

```
    localforce = zeros(nne*dof,1);%external forces are added at the end, so leave as zeros. Add local body force vector from PE minimization here

    x_2 = co(e(A,2),:);
    x_1 = co(e(A,1),:);

    F_1 = -(rho*t^2*w^2*(x_1 - x_2)*(2*x_1 + x_2))/6;
    F_2 = -(rho*t^2*w^2*(x_1 - x_2)*(x_1 + 2*x_2))/6;

    localforce(1) = F_1;
    localforce(2) = F_2;

    %DONT TOUCH BELOW BLOCK!! Assembles the global stiffness matrix, Generic block which works for any element

    for B = 1: nne
        for i = 1: dof
            nK1 = (e(A, B)-1)*dof+i;
            nKe1 = (B-1)*dof+i;
            F(nK1) = F(nK1) + localforce(nKe1);
            for C = 1: nne
                for j = 1: dof
                    nK2 = (e(A, C)-1)*dof+j;
                    nKe2 = (C-1)*dof+j;
                    K(nK1, nK2) = K(nK1, nK2) + localstiffness(nKe1, nKe2);
                end
            end
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end


%%%%%%%%%%%%%%%%%%%BOUNDARY CONDITIONS%%%%%%%%%%%%%%%%%%%%%%%

%external forces
P = M*w^2*Len;
lastnodenumber = length(co);%largest node number
pnode = round(lastnodenumber); %node at which force is applied, here last node
F(pnode) = F(pnode) + P; %given x- component of force in node 2

%Apply displacement BC by eliminating rows and columns of nodes 3-4 (corresponding to
%degrees of freedom 5 to 8) - alternative (and more generic method) is the penalty approach, or
%static condensation approach - see later class notes

deletedofs = [1];%first nodes
K(deletedofs,:) = [];
K(:,deletedofs) = [];
```

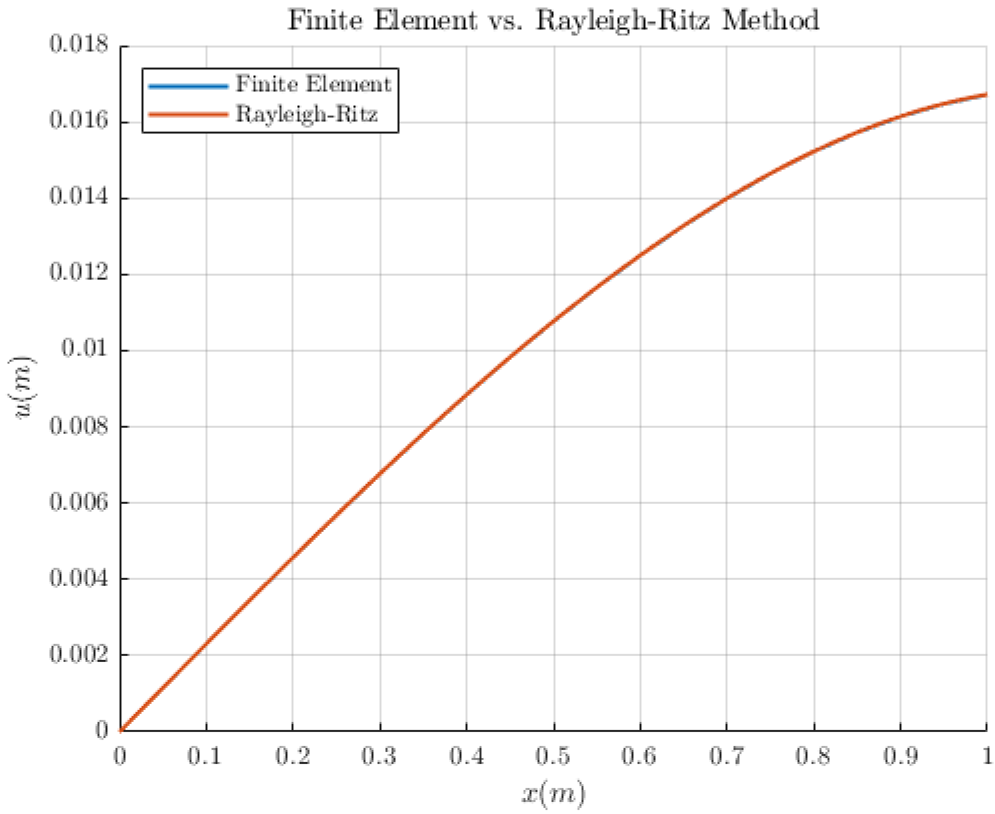
```
F(deletedofs,:) = [];

%%%%%%%%%%%%%%%%%%BOUNDARY CONDITIONS END%%%%%%%%%%%%%%%%%%%%%%


%solve for displacement unknowns (uk)
uk = K\F;

%expand u to include deleted displacement bcs
u = ones(Nnodes*dof,1);
u(deletedofs) = 0;
I = find(u == 1);
u(I) = uk;


figure()
hold on
plot(co,u,'LineWidth',1.5)
```



Rayleigh-Ritz

```
clear

t = 0.01; %cm
E = 70E9;
A = t^2;
```

```
rho = 2700; %kg/m^3
L = 1; %m
omega = 1000; %rad/s

syms b c d x

F_c = rho*A*omega^2*x;
F_point = 0.1*rho*A*omega^2*x;

u = b*x + c*x^2 + d*x^3;

term_1 = 1/2*E*A*diff(u,x)^2;
term_1_int = simplify(int(term_1,x,0,L));

term_2 = subs(F_point,x,L)*subs(u,x,L);

term_3 = u*F_c;
term_3_int = int(term_3,x,0,L);

Pi = term_1_int - term_2 - term_3_int;

eq1 = diff(Pi,b);
eq2 = diff(Pi,c);
eq3 = diff(Pi,d);

[b,c,d] = solve(eq1,eq2,eq3,b,c,d);

b = double(b);
c = double(c);
d = double(d);

%Plotting:

x = linspace(0,L,1000);

u = 0 + b.*x + c.*x.^2 + d.*x.^3;

plot(x,u,'LineWidth',1.5)
xlabel('$x (m)$')
ylabel('$u(m)$')
title('Finite Element vs. Rayleigh-Ritz Method')
legend('Finite Element', 'Rayleigh-Ritz','Location','northwest')
grid on
```



Published with MATLAB® R2022b