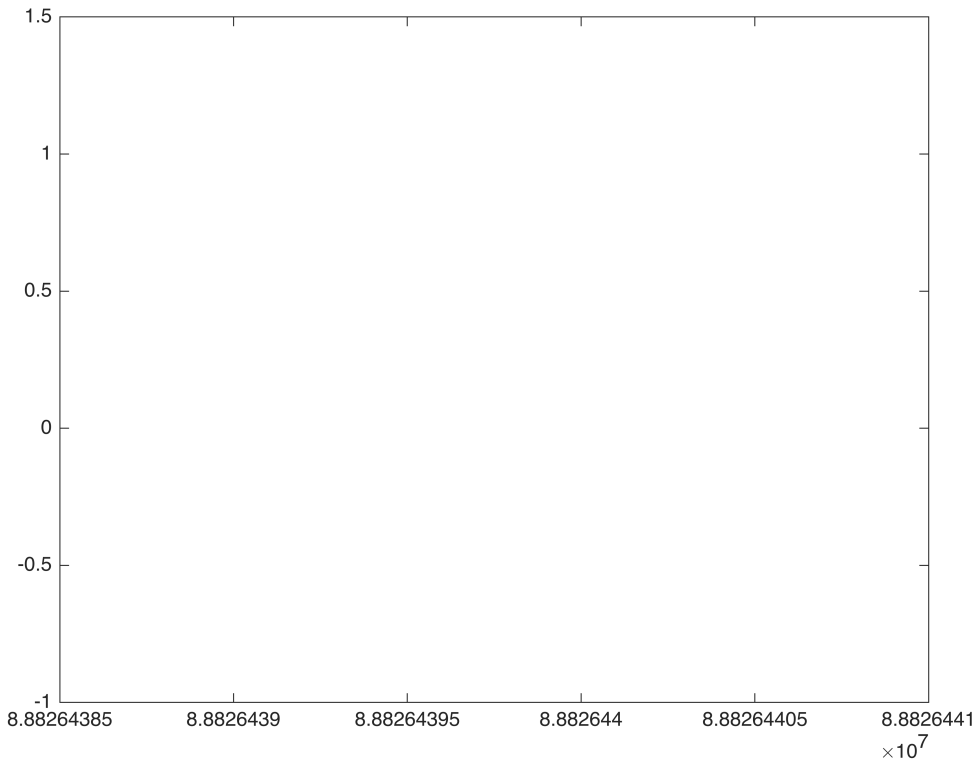
```
% Define the input parameters
L = 20; % Length of the column in meters
b = 5; % Width of the column in meters
t = 6; % Height of the column in meters
P = 100; % Compressive load in Newtons
% Define the material properties
E = 40e6; % Young's modulus in Pascals
nu = 0.50; % Poisson's ratio
rho = 300; % Density of the material in kg/m^3
% Calculate the area and moment of inertia of the cross-section
A = b*t;
I = (1/12)*b*t^3;
% Calculate the critical buckling load for the column
P_cr = pi^2*E*I/L^2;
% Check if the load is less than the critical buckling load
if P <= P_cr
        disp('The column is not in danger of buckling');
else
    disp('The column is in danger of buckling');
end
```

```
The column is not in danger of buckling
```

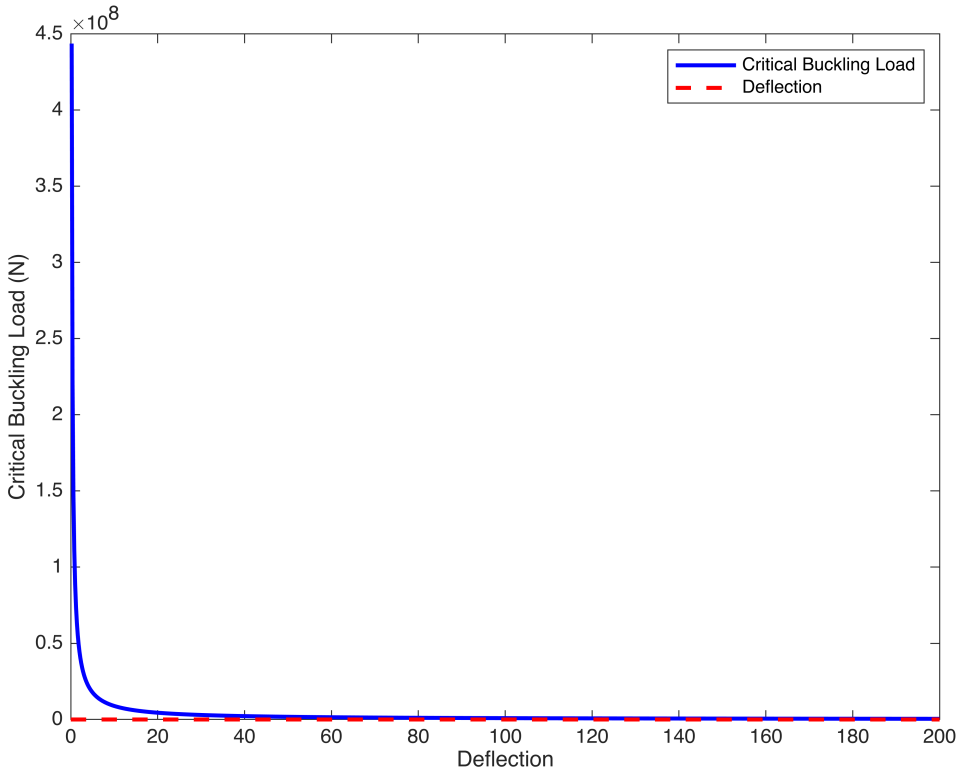
```
% Calculate the deflection of the column due to the compressive load
delta = (P*L^2)/(4*E*I);
% Display the deflection of the column
fprintf('The deflection of the column is %.4f meters\n', delta);
```

```
The deflection of the column is 0.0000 meters
```

```
plot(P_cr,delta);
```




```
P_range = linspace(0, 200, 1000); % Range of compressive loads in Newtons
% Calculate the corresponding critical buckling load and deflection for each value of
P_cr_range = pi^2*E*I./(P_range.*L.^2);
delta_range = (P_range.*L.^2)./(4*E*I);
% Plot P_cr and delta as functions of P
plot(P_range, P_cr_range, 'b-', 'LineWidth', 2);
hold on;
plot(P_range, delta_range, 'r--', 'LineWidth', 2);
% Add axis labels and legend
xlabel(' Deflection ');
ylabel('Critical Buckling Load (N)');
legend('Critical Buckling Load', 'Deflection');
```


×10^8
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Critical Buckling Load (N)
0
20
40
60
80
100
120
140
160
180
200
Deflection
Critical Buckling Load
Deflection