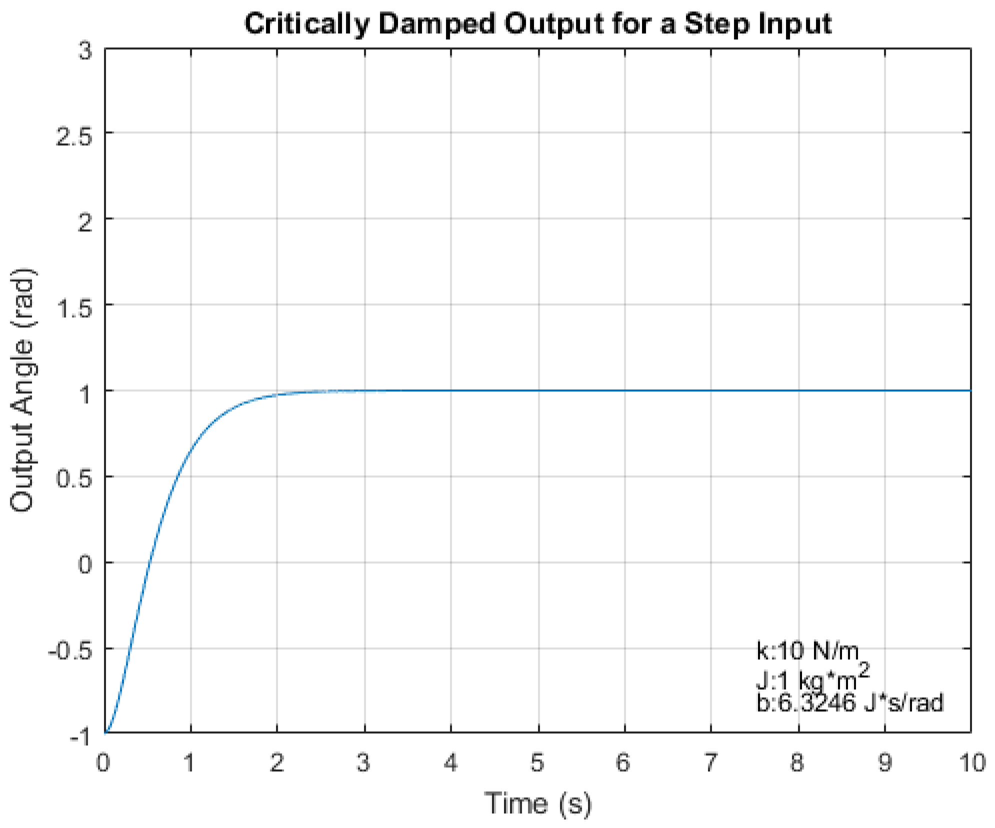
```
function prob1c
```

Simulating a spring mass damper system using ODE45

```
u = 1; %u is a unit step function
k = 10; %N/m
J = 1; %kg*m^2
% b = 5.5; %J*s/rad
b = (2*sqrt(J*k)); %for a critically damped system
%timeframe
ti = 0; %s, initial time
tf = 10; %s, end time
tint = ti:.001:tf; %time interval

%initial conditions
thetaoi = -1;
thetaodi = 0;
y0 = [thetaoi thetaodi]; %these are switched for ode23

[t,theta] = ode45( @state,tint,y0);

function [thetaod] = state(t,x)
x1d = x(2);
x2d = k/J*1-k/J*x(1)-b/J*x(2);

thetaod = [x1d; x2d];
end
%plotting the results
damper = num2str(b); %making a string out of the damper value
massmo = num2str(J);
springcon = num2str(k);
txt1 = strcat('b: ',damper,' J*s/rad');
txt2 = strcat('J: ',massmo,' kg*m^2');
txt3 = strcat('k: ',springcon,' N/m');

figure(3)
plot(tint,theta(:,1))
title('Critically Damped Output for a Step Input')
xlabel('Time (s)')
ylabel('Output Angle (rad)')
xmin = ti;
xmax = tf;
ymin = -1;
ymax = 3;
axis ([xmin xmax ymin ymax])
text(.75*xmax,.8*ymin,txt1)
text(.75*xmax,.65*ymin,txt2)
text(.75*xmax,.5*ymin,txt3)
grid on
```



```
end
```

Published with MATLAB® R2016b