2.26 Consider the following differential equation:

$$\frac{d^2y(t)}{dt^2} + 3\frac{dy(t)}{dt} + 2y(t) = 0, \quad y(0) = 1, \quad \dot{y}(0) = 0$$

- **(a)** Solve for $y(t)$, using the MATLAB Symbolic Math Toolbox.
- **(b)** Using Euler's approximation of derivatives with T arbitrary and input $x(t)$ arbitrary, derive a difference equation model. Using the M-file `recur` with $T = 0.4$, compute the approximation to $y(t)$.
- **(c)** Repeat the numerical approximation in part (b) for $T = 0.1$.
- **(e)** Plot the responses obtained in parts (a), (b), and (c) for $0 \leq t \leq 10$, and compare the results.

➢ without part d. Can analytically solve part a using any method. Use the <u>backward difference</u> Euler's approximation in parts b & c and list the I/O difference equation w/ coefficients evaluated in each case. For part e, plot the analytic result (solid line) from part a with the numerical results from part b (dots) and then plot parts a (solid line) & c (dots) on a separate plot. Use a legend on the plots.

a) From the MATLAB Command Window (extra blank lines removed for clarity)

```
>> dsolve('D2y = -3*Dy - 2*y','y(0) = 1, Dy(0) = 0')
 ans =  2*exp(-t)-exp(-2*t)
>> dsolve('D2y = -3*Dy - 2*y','y(0) = 1','Dy(0) = 0')
 ans =  2*exp(-t)-exp(-2*t)
```

Note that the initial conditions are in two formats (see MATLAB help for dsolve). So,

$$\underline{y(t) = 2e^{-t} - e^{-2t} \text{ for } t \geq 0}.$$

Since $x(t) = 0$, this is the natural or unforced solution to the differential equation.

b) Use the backward difference Euler's approximations

$$\left.\frac{df(t)}{dt}\right|_{t=nT} \approx \frac{f[n]-f[n-1]}{T} \text{ and } \left.\frac{d^2f(t)}{dt^2}\right|_{t=nT} \approx \frac{f[n]-2f[n-1]+f[n-2]}{T^2}$$

in the above differential equation with arbitrary input $x(t)$ to get

$$\frac{y[n]-2y[n-1]+y[n-2]}{T^2} + 3\frac{y[n]-y[n-1]}{T} + 2y[n] = x[n].$$

This can be simplified to the standard difference equation

$$y[n]-\left(\frac{2+3T}{1+3T+2T^2}\right)y[n-1]+\left(\frac{1}{1+3T+2T^2}\right)y[n-2]=\left(\frac{T^2}{1+3T+2T^2}\right)x[n] \quad \text{for } \ n\geq 1$$

or the recursive difference equation

$$y[n]=\left(\frac{2+3T}{1+3T+2T^2}\right)y[n-1]-\left(\frac{1}{1+3T+2T^2}\right)y[n-2]+\left(\frac{T^2}{1+3T+T^2}\right)x[n] \quad \text{for } \ n\geq 1$$

with initial conditions

$$y(0)=1 \quad \Rightarrow \quad y[0]=1$$

and

$$\dot{y}(0)=\left.\frac{d\,y(t)}{dt}\right|_{t=0}=0 \quad \Rightarrow \quad \frac{y[0]-y[0-1]}{T}=\frac{1-y[-1]}{T}=0 \quad \Rightarrow \quad y[-1]=1\,.$$

With T = 0.4 s, the difference equation is

$$y[n]-1.27\,y[n-1]+0.3968\,y[n-2]=0.0635\,x[n] \quad \text{for } \ n\geq 1.$$

c) With T = 0.1 s, the difference equation is

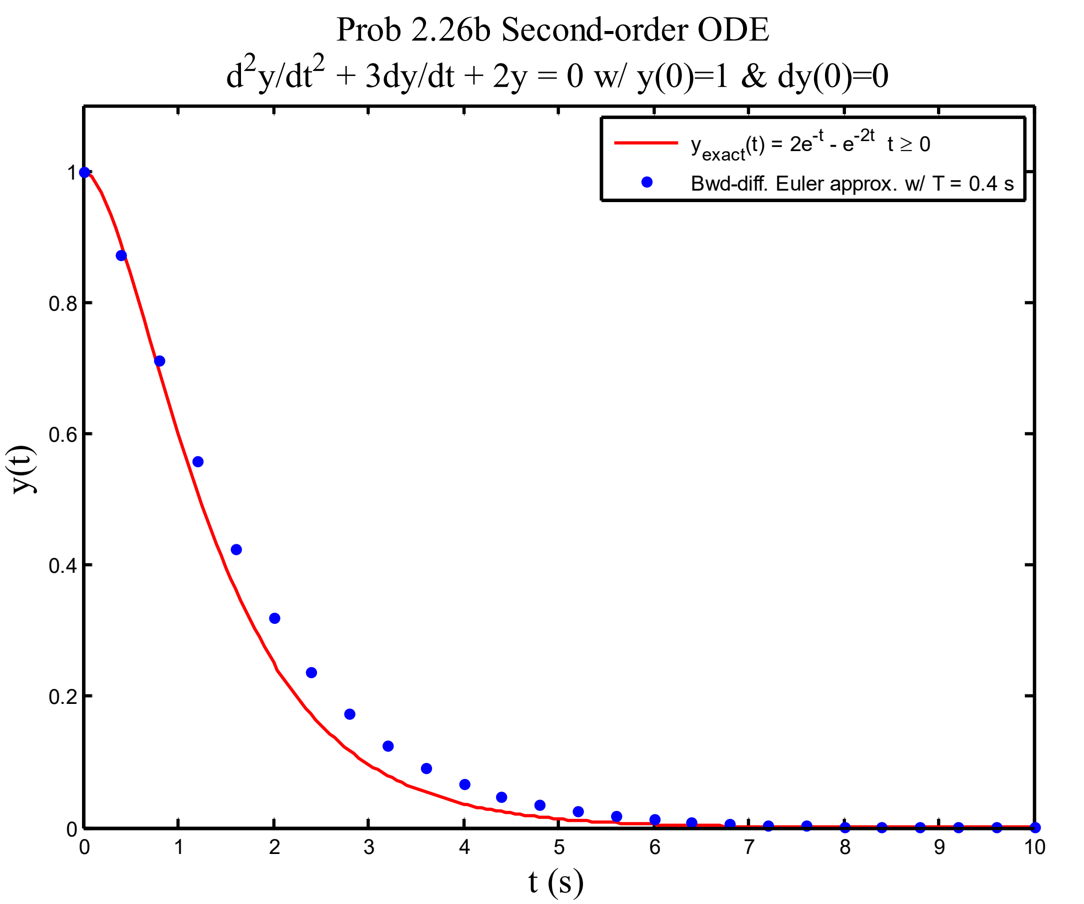
$$y[n]-1.7424\,y[n-1]+0.7576\,y[n-2]=0.00\overline{75}\,x[n] \quad \text{for } \ n\geq 1.$$

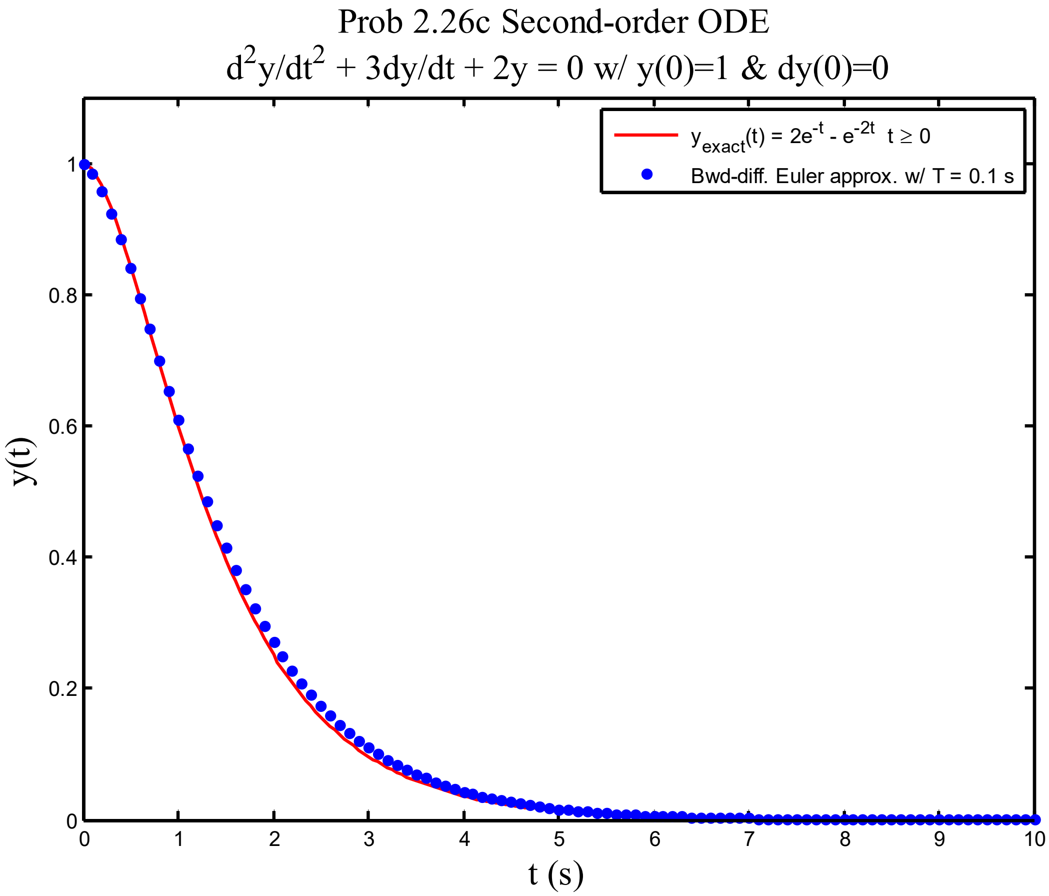
M-file for part b), similar m-file used for part c) with variable T = 0.1 s and label changes.

```
% Chapter 2 problem 2.26b (p2_26b.m)
%
% For a second-order ordinary differential equation (ODE)
%                d2y/dt2 + 3dy/dt + 2y = x(t)
% where y(0) = 1 & dy(0)/dt = 0 and x(t)=0.
% The ODE has an analytic solution-
%                y(t) = 2e^(-t)-e^(-2t).
% Find approximate numerical solution by using a backward-difference
% Euler's approximation for derivatives to change it into a
% second-order difference equation which can be solved
% recursively.  Compare numerical results with exact solution.
%
close all; clear; clc;
tstop = 10; % How far to go in time in seconds
% *** Backward-difference Euler approximation ***
T = 0.4;                    % Time step for numerical approximation
a1 = (-2-3*T)/(1 + 3*T + 2*T*T);
a2 = 1/(1 + 3*T + 2*T*T);
b0 = (T*T)/(1 + 3*T + 2*T*T);
a=[a1, a2];                  % [a] coefficient vector for y[] terms
b=[b0];                      % [b]  coefficient vector for x[n]
```

```
n = 1:1:round(tstop/T);        % Define index vector for recur()
x = zeros(1,length(n));        % input x[n]=0
x0 = []; y0=[1,1];           % initial conditions (oldest to youngest)
y = recur(a,b,n,x,x0,y0);      % yields output for n=1,2,3,...
yapprox = [y0(2),y]; n=[0,n];      % tack on value at t=n=0
%
% *** Analytic solution ****
t = 0:0.05:tstop;                  % Define time steps for analytic sol'n
yexact = 2*exp(-t)-exp(-2*t);
%
plot(t,yexact,'r',n*T,yapprox,'b.',[0 tstop],[0 0],'k-')
legend(' y_{exact}(t) = 2e^{-t} - e^{-2t}  t \geq 0',...
    [' Bwd-diff. Euler approx. w/ T = ',num2str(T),' s'],'Location','NE'),
axis([0 tstop 0 1.1]),
ylabel('y(t)','fontsize',16,'fontname','times')
xlabel('t (s)','fontsize',16,'fontname','times')
title({'Prob 2.26b Second-order ODE ';...
    'd^{2}y/dt^2 + 3dy/dt + 2y = 0 w/ y(0)=1 & dy(0)=0'},...
    'fontsize',16,'fontname','times')
set(findobj('type','line'),'linewidth',1.5)
set(findobj('type','line'),'markersize',14)
set(findobj('type','axes'),'linewidth',2)
```

e) Plots





- Of course, the analytic solution to the ODE is perfect.
- For the backward difference Euler's approximations to the ODE, the numerical solution with the smaller step size is more accurate.