Problem 1.

Find the characteristic equation and reduced characteristic equation of the 3-step method

$$y_{n+1} = -\frac{27}{11}y_n + \frac{27}{11}y_{n-1} + y_{n-2} + h\left[\frac{3}{11}f_{n+1} + \frac{27}{11}f_n + \frac{27}{11}f_{n-1} + \frac{2}{11}f_{n-2}\right]$$

Find the (exact) roots of the reduced characteristic equation, and determine if this method is stable for small $h\lambda$ or not.

Solution

TODO

Problem 2.

The IVP

$$y' = -y + 2e^{-t}\cos(2t)$$
$$y(0) = 0$$

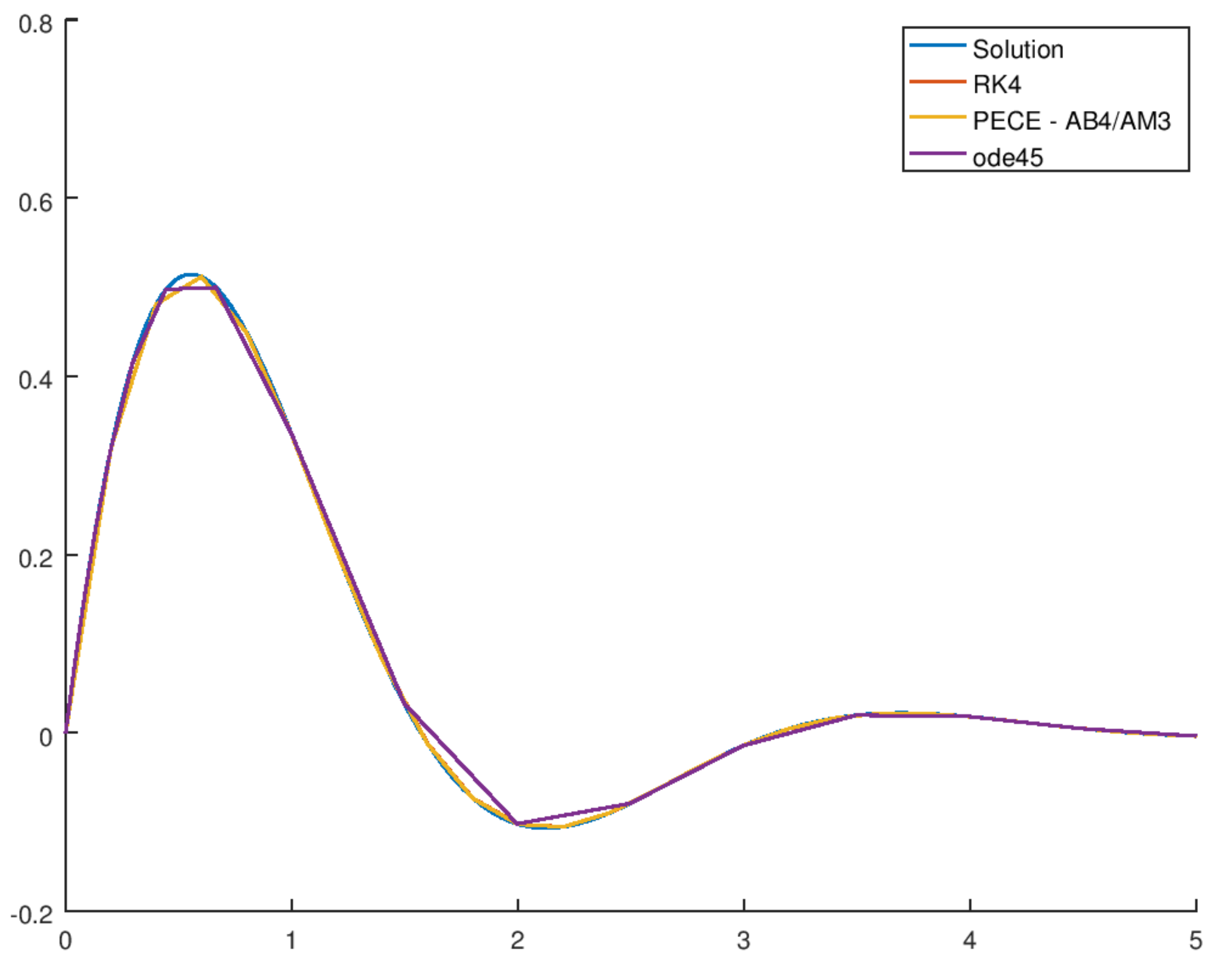
has the true solution $y(t) = e^{-t}\sin(2t)$.
Solve it numerically from t=0 to t=5, using

(a) the classical 4-stage Runge-Kutta method with h=0.2

(b) a PECE predictor-corrector method based on 4-step AB, 3-step AM, h=0.2
Use RK values from part (a) for the startup.

(c) the Matlab built-in ODE45. This routine will pick its own steps.

Solution



	Actual	RK4	PECE - AB4/AM3	ODE45 (h unknown)
h = 0.2				
Value	-0.013911	-0.013911	-0.013776	-0.014068
Error		$2.1822 * 10^{-7}$	$1.3560 * 10^{-4}$	$1.5663 * 10^{-4}$
h = 0.1				
Value	-0.013911	-0.013911	-0.013908	
Error		$1.2907 * 10^{-8}$	$3.3871 * 10^{-6}$	

```
function p2

clear all;
close all;
hold on;

% Actual solution.
_f = @(t) e.^(-t) .* sin(2 .* t);
_t = linspace(0, 5, 200);
_y = _f(_t);

plot(_t, _y, 'DisplayName', 'Solution', 'LineWidth', 1);
val = _f(3)

% Numerical Solutions
f = @(t, y) -y + 2 .* e^(-t) * cos(2*t);
h = 0.2; % Step Size
t = 0:h:5; % Interval
y = [0]; % Initial point
steps = length(t);

% RK4
for n = 1:(steps-1)

    % 4-Step
    k1 = f(t(n),        y(n));
    k2 = f(t(n) + h/2,        y(n) + (h/2) * k1);
    k3 = f(t(n) + h/2,        y(n) + (h/2) * k2);
    k4 = f(t(n) + h,        y(n) + h * k3);

    % Compute y_{n+1}
    y(n+1) = y(n) + (h/6) * (k1 + 2*k2 + 2*k3 + k4);

end

val = y((3/h) + 1)
err = norm(_f(3) - val, inf)
```

```
plot(t, y, 'DisplayName', 'RK4', 'LineWidth', 1);

% PECE
y = y(1:4); % Pull startup values from RK4

for n = 5:steps

    % P: 4-Step AB
    y(n) = y(n-1) + h * (
        (55/24) * f(t(n-1), y(n-1)) -
        (59/24) * f(t(n-2), y(n-2)) +
        (37/24) * f(t(n-3), y(n-3)) -
        (3/8)        * f(t(n-4), y(n-4))
    );

    % C: 3-Step AM
    y(n) = y(n-1) + h * (
        (3/8)        * f(t(n), y(n)) +
        (19/24)        * f(t(n-1), y(n-1)) -
        (5/24)        * f(t(n-2), y(n-2)) +
        (1/24)        * f(t(n-3), y(n-3))
    );

end

val = y((3/h) + 1)
err = norm(_f(3) - val, inf)
plot(t, y, 'DisplayName', 'PECE - AB4/AM3', 'LineWidth', 1);

% ODE45
% Note I am using octave, so the implementation may differ slightly
% from the one provided by Matlab.

[solx, soly] = ode45(f, [0 5], [0 0]);
val = interp1(solx, soly(:,1), 3, method='linear')
err = norm(_f(3) - val, inf)
plot(solx, soly(:,1), 'DisplayName', 'ode45', 'LineWidth', 1);

legend('show');

end
```

Problem 3.

Use the Matlab routine $fzero$ or something comparable to find the two solutions of

$$e^x - x - 2 = 0.$$

There is one positive and one negative solution.

Solution

TODO