

MATH 335: Numerical Analysis

Problem Set 18, Solutions

★38 (a) My routine looks like:

```
function [y, t] = modEuler (f, t0, tlast, y0, N)
% function [y, t] = modEuler (f, t0, tlast, y0, N)
%
% This routine calculates an approximate solution at mesh points
% t0, t1, ..., tN to the initial-value problem
%
%          y' = f(t, y),    subject to y(t0) = y0,
%
% using the 2nd-order Runge-Kutta method also known as the
% modified Euler method. We assume evenly-spaced mesh points
% with stepsize
%
%          h = (tlast - t0) / N.
%
% INPUTS:
%   f      function handle to the RHS in the ODE giving y'
%   t0     initial "time"
%   tlast  last "time" at which to approximate the solution
%   y0     initial value, the value of the solution at the initial time
%   N      number of steps to take to go from t0 to tN
%
% OUTPUTS:
%   y      approximate value of solution at mesh points
%   t      mesh points

h = (tlast - t0) / N;
t = [t0:h:tlast]';
y = y0;
for jj = 1:N
    k1 = f(t(jj), y(jj));
    k2 = f(t(jj) + h/2, y(jj) + h*k1/2);
    y(jj + 1) = y(jj) + h*k2;
end
y = y(:);
end
```

(b) My implementation of Euler's method:

```
function [y, t] = euler (f, t0, tlast, y0, N)
    h = (tlast - t0) / N;
    t = [t0:h:tlast]';
    y = y0;
    for jj = 1:N
        y(jj + 1) = y(jj) + h*f(t(jj), y(jj));
    end
    y = y(:);
end
```

My implementation of a 4th-order Runge-Kutta method:

```
function [y, t] = rk4(f, t0, tlast, y0, N)
    h = (tlast - t0) / N;
    t = [t0:h:tlast]';
    y = y0;
    for jj = 1:N
        k1 = f(t(jj), y(jj));
        k2 = f(t(jj) + h/2, y(jj) + h*k1/2);
        k3 = f(t(jj) + h/2, y(jj) + h*k2/2);
        k4 = f(t(jj) + h, y(jj) + h*k3);
        y(jj + 1) = y(jj) + h*(k1 + 2*k2 + 2*k3 + k4) / 6;
    end
    y = y(:);
end
```

(c) Let us solve this problem, say, on the interval $[1, 3]$. We will compare solution values at the right endpoint $t = 3$, using various stepsizes.

h	Euler			Mod. Euler		
	at $t = 3$	Error	$ E_{n-1}/E_n $	at $t = 3$	Error	$ E_{n-1}/E_n $
2^{-2}	1.7053	0.17137		1.9133	-0.3667	
2^{-3}	1.8032	0.07346	2.333	1.8845	-0.007909	46.36
2^{-4}	1.8421	0.03457	2.125	1.8784	-0.00178	4.443
2^{-5}	1.8598	0.01682	2.055	1.8770	-0.0004219	4.219
h	Heun			RK4		
	at $t = 3$	Error	$ E_{n-1}/E_n $	at $t = 3$	Error	$ E_{n-1}/E_n $
2^{-2}	1.8685	0.0081004		1.8763	0.0003407	
2^{-3}	1.8756	0.0010214	7.931	1.8766	0.000014742	23.11
2^{-4}	1.8765	0.00016888	6.048	1.8766	0.0000006819	21.62
2^{-5}	1.8766	0.000034059	4.958	1.8766	0.00000003537	19.28

11.4.3 The following commands give us the approximate values

$$y(1.02) \doteq 1.9985, \quad y(1.67) \doteq (2.3388), \quad y(1.98) \doteq 2.6926.$$

```
octave:27> function out = f(t, y), out = y*(1 - 1/t); end
octave:28> [yjs, tjs] = rk4(@f, 1, 2, 2, 10);
octave:34> c = polyfit(tjs, yjs, 2);
octave:36> polyval(c, [1.02 1.67 1.98])
ans =
    1.9985    2.3388    2.6926
```

CP 11.4.6 (b) The results are given below.

t_j	true solution	$y_1(t_j)$	$y_2(t_j)$	$e_1(t_j)$	$e_2(t_j)$
0.0	1.00	1.00	1.0000	0	0
0.1	1.21	1.21	1.2091	0	0.0009
0.2	1.44	1.44	1.4381	0	0.0019
0.3	1.69	1.69	1.6870	0	0.0030
0.4	1.96	1.96	1.9558	0	0.0042
0.5	2.25	2.25	2.2445	0	0.0055
0.6	2.56	2.56	2.2553	0	0.0069
0.7	2.89	2.89	2.8817	0	0.0084
0.8	3.24	3.24	3.2301	0	0.0099
0.9	3.61	3.61	3.5984	0	0.0116
1.0	4.00	4.00	3.9867	0	0.0133

(c) To see that, when applied to the first equation, Heun's method gives exactly correct values at mesh points, note that for $j = 1, 2, \dots, N$,

$$\begin{aligned}
 y_j &= y_{j-1} + \frac{h}{2}[f(t_{j-1}) + f(t_j)] \\
 &= y_{j-2} + \frac{h}{2}[f(t_{j-2}) + f(t_{j-1})] + \frac{h}{2}[f(t_{j-1}) + f(t_j)] = y_{j-2} + \frac{h}{2}[f(t_{j-2}) + 2f(t_{j-1}) + f(t_j)] \\
 &= \cdots = y_0 + \frac{h}{2}[f(t_0) + 2f(t_1) + \cdots + 2f(t_{j-1}) + f(t_j)] \\
 &= y_0 + \int_{t_0}^{t_j} f(t) dt \quad (\text{since comp. trap. rule exact for polynomial } f \text{ with } \deg f < 2) \\
 &= y_0 + \int_{t_0}^{t_j} y'(t) dt = y_0 + [y(t)]_{t_0}^{t_j} = y(t_j).
 \end{aligned}$$