MATH7003-00: Assignment #7

Sangman Jung
e-mail: sangmanjung@khu.ac.kr
Kyung Hee University – May 20, 2020

Problem.

1. Consider the example in slide 19 and implement MATLAB code to obtain the Table 6.24. Discuss about the results. [1].

2. Consider the example in slide 19 and use smaller step sizes, $h=(1/2)^k$, $k=3,4,5,6,7$ and study the order of convergence at $x=2.0$ by running the MATLAB code for those h values.

Example. Consider the problem

$$y' = \frac{1}{1+x^2} - 2y^2 \qquad y(0)=0$$

with the solution $Y = x/(1+x^2)$. The method (6.10.21) was used with a fixed stepsize, and the results are shown in Table 6.24. The step sizes are $h=0.25$ and $2h=0.5$. The column 'Ratio' gives the ratio of the errors for corresponding node points as h is halved. The last column is an example of formula (6.10.24) from the following material on Richardson extrapolation. Because $T_n(Y)=O(h^5)$ for method (6.10.21), Theorem 6.9 implies that the rate of convergence of $y_h(x)$ to $Y(x)$ is $O(h^4)$. The theoretical value of Ratio is 16, and as h decreases further, this value will be realized more closely.

Table 6.24 Example of Runge–Kutta method (6.10.21)

x	$y_h(x)$	$Y(x)-y_h(x)$	$Y(x)-y_{2h}(x)$	Ratio	$\frac{1}{15}[y_h(x)-y_{2h}(x)]$
2.0	.39995699	4.3E − 5	1.0E − 3	24	6.7E − 5
4.0	.23529159	2.5E − 6	7.0E − 5	28	4.5E − 6
6.0	.16216179	3.7E − 7	1.2E − 5	32	7.7E − 7
8.0	.12307683	9.2E − 8	3.4E − 6	36	2.2E − 7
10.0	.09900987	3.1E − 8	1.3E − 6	41	8.2E − 8

Discussion.

Our choice of the numerical scheme is the **fourth-order Runge-Kutta method** (RK4 method) as follows.

$$y_{n+1} = y_n + \frac{h}{6}\left[V_1 + 2V_2 + 2V_3 + V_4\right] \text{ where}$$

$$V_1 = f(x_n, y_n),\quad V_2 = f\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hV_1\right),\quad V_3 = f\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hV_2\right),\quad V_4 = f(x_n + h, y_n + hV_3).$$

Note that the truncation error of this scheme is $T_n(Y)=O(h^5)$. Then, we also know that the rate of

convergence of $y_h(x)$ to $Y(x)$ is $O(h^4)$ (refer to **Corollary** in the section 6.10). In our problem, we let $h = 0.25$ $2h = 0.5$ and $0 \le x \le 10$. In the previous sections, since we learned and used the single step method like Euler's method, for example, there is no detailed description for RK4 method in this work.

Additionally, we investigated the order of convergence of this method in limited case, and showed that if h is smaller, then the numerical ratio is closer to theoretical ratio, using MATLAB.

Note that the ratio in Table 6.24 is that:

$$\text{Ratio} = \frac{\text{Error}_{2h}}{\text{Error}_{h}}, \quad \text{where } \text{Error}_{2h} = Y(x_{2h}) - y_{2h}(x_{2h}),\ \text{Error}_{h} = Y(x_h) - y_h(x_h).$$

Also, the ratio in the additional problem (Order of convergence) is that:

$$\text{Ratio}_{k} = \frac{\text{Error}_{2k}}{\text{Error}_{k}}$$

where $\text{Error}_{2k} = Y(x_{2((1/2)^k)}) - y_{2((1/2)^k)}(x_{2((1/2)^k)})$, $\text{Error}_{k} = Y(x_{(1/2)^k}) - y_{(1/2)^k}(x_{(1/2)^k})$ for $k = 3,4,5,6,7$.

The results of the problem is as follows.

명령 창

Table 6.24 Example of Runge-Kutta method (6.10.21)

x	y_{h}(x)	Y(x)-y_{h}(x)	Y(x)-y_{2h}(x)	Ratio	[y_{h}(x)-y_{2h}(x)]/15
2	0.39995699	4.3e-05	1.0e-03	24	6.7e-05
4	0.23529159	2.5e-06	7.0e-05	28	4.5e-06
6	0.16216179	3.7e-07	1.2e-05	32	7.7e-07
8	0.12307683	9.2e-08	3.4e-06	36	2.2e-07
10	0.09900987	3.1e-08	1.3e-06	41	8.2e-08

Table 6.24(HW#7) Order of convergence at x = 2.0

k	h = (1/2)^k	y_{h}(x)	Y(x)-y_{h}(x)	Y(x)-y_{2h}(x)	Ratio
3	0.1250	0.39999779	2.2e-06	4.3e-05	19
4	0.0625	0.39999987	1.3e-07	2.2e-06	18
5	0.0313	0.39999999	7.5e-09	1.3e-07	17
6	0.0156	0.40000000	4.5e-10	7.5e-09	16
7	0.0078	0.40000000	2.8e-11	4.5e-10	16

fx >>

Figure 1. The results of the problem using MATLAB.

In **Figure 1.**, we can see that if h is smaller, then our numerical ratio is closer to 16 that is theoretical ratio. The figure below is the ratio graph per step size $h = (1/2)^k$, $k = 3,4,5,6,7$. This supports our intuition of the problem.

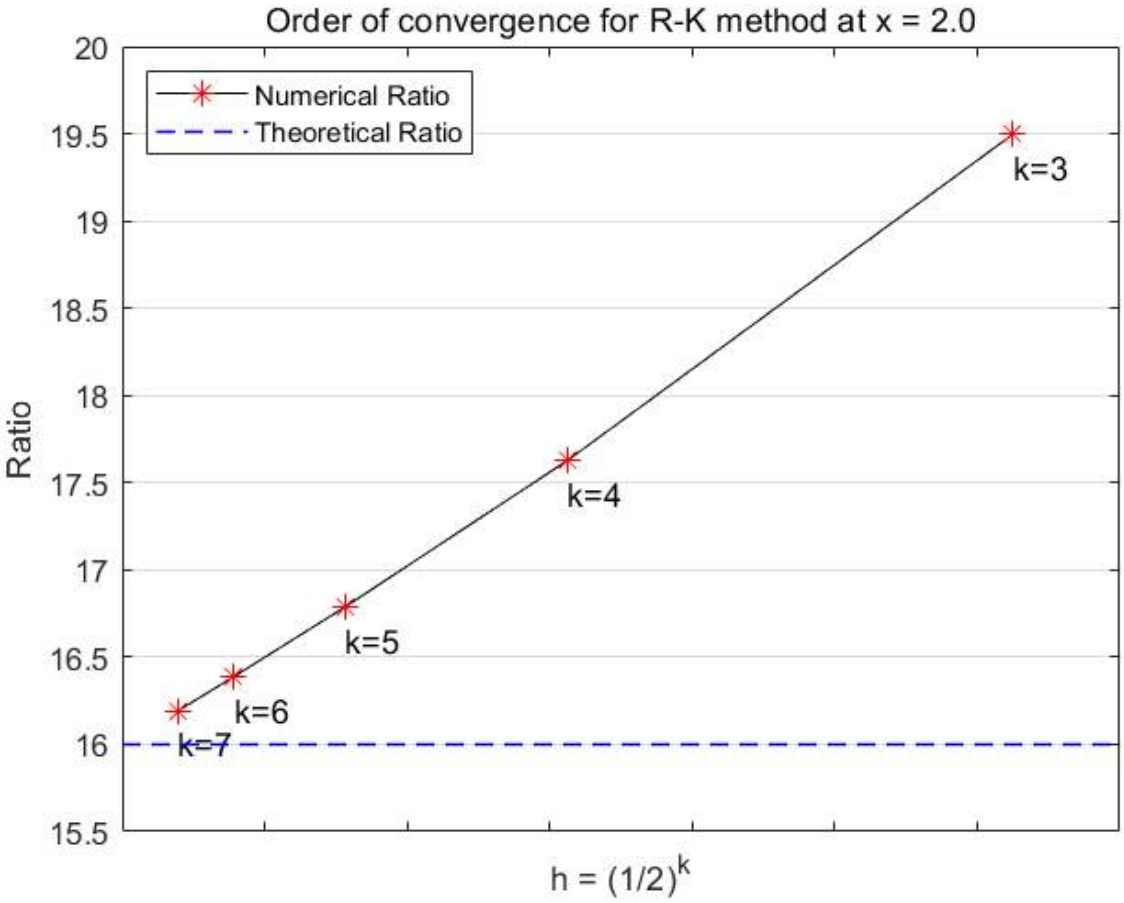

Figure 2. The graph for the ratio curve with respect to k. The black curve and red marker is numerical ratio for RK4 method. The blue dashed line is theoretical ratio. At fixed position $x = 2$, we can see that if h is decreasing (k is increasing), then the numerical ratio approximate 16, as theoretical ratio.

Table_6_24_and_Order_of_convergence_HW7.m

```
%% MATH7003-00: Assignment #7, 2019310290 Sangman Jung.
clear, close, clc
% functions
f = @(x,y) (1./(1+x.^2)) - 2*y.^2; % ODE
Y = @(x) x./(1+x.^2); % exact solution
ktable = 3:7;
K = 1./2.^(ktable); % additional topics in HW#7
h = [0.25 0.5 K 2*K]; % step size h (1x12)

% Runge-Kutta method iteration
for h_iter = 1:length(h) % h = 0.25, 0.5 & h = (1/2)^k
    x = 0:h(h_iter):10; % variable x per step size
    y = zeros(1,length(x)); % allocate the numerical solution y
    y(1) = 0; % initial condition
    for n = 1:length(x)-1 % Runge-Kutta iteration
        V1(n) = f(x(n),y(n));
        V2(n) = f(x(n)+h(h_iter)/2,y(n)+(h(h_iter)*V1(n))/2);
        V3(n) = f(x(n)+h(h_iter)/2,y(n)+h(h_iter)/2*V2(n));
        V4(n) = f(x(n)+h(h_iter),y(n)+h(h_iter)*V3(n));
        y(n+1) = y(n) + h(h_iter)*(V1(n)+2*V2(n)+2*V3(n)+V4(n))/6;
    end
    xvals(h_iter) = {x'}; % save the values of x for h and 2h
    yvals(h_iter) = {y'}; % save the values of y for h and 2h
end

% print out setting for Table 6.24
xh = cell2mat(xvals(1)); yh = cell2mat(yvals(1));
x2h = cell2mat(xvals(2)); y2h = cell2mat(yvals(2));

% print out setting for h = (1/2)^k
xk3 = cell2mat(xvals(3)); yk3 = cell2mat(yvals(3));
xk4 = cell2mat(xvals(4)); yk4 = cell2mat(yvals(4));
xk5 = cell2mat(xvals(5)); yk5 = cell2mat(yvals(5));
xk6 = cell2mat(xvals(6)); yk6 = cell2mat(yvals(6));
xk7 = cell2mat(xvals(7)); yk7 = cell2mat(yvals(7));
x2k3 = cell2mat(xvals(8)); y2k3 = cell2mat(yvals(8));
x2k4 = cell2mat(xvals(9)); y2k4 = cell2mat(yvals(9));
x2k5 = cell2mat(xvals(10)); y2k5 = cell2mat(yvals(10));
x2k6 = cell2mat(xvals(11)); y2k6 = cell2mat(yvals(11));
x2k7 = cell2mat(xvals(12)); y2k7 = cell2mat(yvals(12));

% find the index of Table 6.24
Table = 2:2:10;
ind_xh = zeros(1,length(Table));
ind_x2h = zeros(1,length(Table));
for i = Table
    ind_xh(i) = find(xh == i);
    ind_x2h(i) = find(x2h == i);
end
ind_xh = nonzeros(ind_xh);
ind_x2h = nonzeros(ind_x2h);

% find the index of h = (1/2)^k
kx = 2; % we want to see only the ratio at x = 2.0
ind_xk3 = find(xk3 == kx); ind_xk4 = find(xk4 == kx);
ind_xk5 = find(xk5 == kx); ind_xk6 = find(xk6 == kx);
ind_xk7 = find(xk7 == kx);
ind_x2k3 = find(x2k3 == kx); ind_x2k4 = find(x2k4 == kx);
ind_x2k5 = find(x2k5 == kx); ind_x2k6 = find(x2k6 == kx);
ind_x2k7 = find(x2k7 == kx);
```

```
% calculate the columns for Table 6.24
yh_table = yh(ind_xh);
Error_h = Y(xh(ind_xh))-yh(ind_xh);
Error_2h = Y(x2h(ind_x2h))-y2h(ind_x2h);
Ratio = Error_2h./Error_h;
Richardson = (yh(ind_xh) - y2h(ind_x2h))/15;

% calculate the columns for h = (1/2)^k
yk_t = [yk3(ind_xk3) yk4(ind_xk4) yk5(ind_xk5) yk6(ind_xk6) yk7(ind_xk7)];
y2k_t = [y2k3(ind_x2k3) y2k4(ind_x2k4) y2k5(ind_x2k5) y2k6(ind_x2k6) y2k7(ind_x2k7)];

Error_k = [Y(xk3(ind_xk3))-yk_t(1) Y(xk4(ind_xk4))-yk_t(2) ...
    Y(xk5(ind_xk5))-yk_t(3) Y(xk6(ind_xk6))-yk_t(4) ...
    Y(xk7(ind_xk7))-yk_t(5)];

Error_2k = [Y(x2k3(ind_x2k3))-y2k_t(1) Y(x2k4(ind_x2k4))-y2k_t(2) ...
    Y(x2k5(ind_x2k5))-y2k_t(3) Y(x2k6(ind_x2k6))-y2k_t(4)...
    Y(x2k7(ind_x2k7))-y2k_t(5)];

Ratio_k = [Error_2k(1)/Error_k(1) Error_2k(2)/Error_k(2) Error_2k(3)/Error_k(3) ...
    Error_2k(4)/Error_k(4) Error_2k(5)/Error_k(5)];

% Table 6.24
fprintf('Table 6.24  Example of Runge-Kutta method (6.10.21)\n');
fprintf('-------------------------------------------------------------------------------------------------\n');
fprintf('   x        y_{h}(x)    Y(x)-y_{h}(x)    Y(x)-y_{2h}(x)    Ratio    [y_{h}(x)-y_{2h}(x)]/15\n');
fprintf('-------------------------------------------------------------------------------------------------\n');
for t = 1:length(Table)
    fprintf('  %2s      % 1.8f     % 1.1e         % 1.1e       % 2.0f          % 1.1e\n',...
       num2str(Table(t)),yh_table(t),Error_h(t),Error_2h(t),Ratio(t),Richardson(t));
end
fprintf('-------------------------------------------------------------------------------------------------\n\n\n');

% The study of order of convergence at x = 2.0
fprintf('Table 6.24(HW#7) Order of convergence at x = 2.0\n');
fprintf('-----------------------------------------------------------------------------\n');
fprintf('    k    h = (1/2)^k    y_{h}(x)    Y(x)-y_{h}(x)    Y(x)-y_{2h}(x)     Ratio    \n');
fprintf('-----------------------------------------------------------------------------\n');
for T = ktable
    fprintf('    %s      %1.4f     % 1.8f     % 1.1e         % 1.1e         % 2.0f  \n',...
       num2str(T),h(T),yk_t(T-2),Error_k(T-2),Error_2k(T-2),Ratio_k(T-2));
end
fprintf('-----------------------------------------------------------------------------\n');

% Ratio graph
for i = ktable
    figtxt(i-2) = {sprintf('k=%d',i)};
end
figure('Name','Convergence analysis for R-K method')
plot(h(3:7),Ratio_k,'k-*','MarkerEdgeColor','r','MarkerSize',8); ylim([15.5 20]);
xticklabels('');
ax=gca; ax.YGrid='on';
xlabel('h = (1/2)^{k}'); ylabel('Ratio');
title('Order of convergence for R-K method at x = 2.0');
text(h(3:7),Ratio_k-0.19,figtxt,'FontSize',11)
hold on
line([0 0.14],[16 16],'Color','b','LineStyle','--','LineWidth',0.9);
legend('Numerical Ratio','Theoretical Ratio','Location','northwest');
hold off
```

References.

[1] Atkinson, K. E. (2008). An introduction to numerical analysis. John wiley & sons.

[2] Atkinson, K., Han, W., & Stewart, D. E. (2011). Numerical solution of ordinary differential equations (Vol. 108). John Wiley & Sons.

[3] Atkinson, K. E., & Han, W. (1985). Elementary numerical analysis (p. 17). New York et al.: Wiley.