Hand in Assignment 3 – Explicit Integrators

Fikri Farhan Witjaksono *Oliver Wallin*

October 18, 2019

- # Introduction

This assignment handles application of various estimation methods namely Euler,RK2 and RK4.

- # Question 1 General Runge-Kutta 2 methods for a dynamic model

a) Provide condition a,b such that the error e_k of the method is of order 3

$We\ consider\ the\ time\ interval\ [t_k, t_{k+1}],\ such\ that,\ u = u_k\ is\ constant\ on\ the\ time\ interval.$

$We\ consider\ the\ function\ K_1\ is\ independent\ of\ time \rightarrow K_1 = f$

$We\ now\ consider\ that\ K_2\ is\ independent\ of\ time \rightarrow$
$K_2 = f(x(t_k) + a\Delta tK_1, u_k) \rightarrow f(x(t_k) + a\Delta tf, u_k)$

$Equation\ (2C)\ from\ question$

$$x_{k+1} = x(t_k) + \Delta t \sum_{i=1}^{2} b_i K_i = x(t_k) + \Delta t(b_1K_1 + b_2K_2) \rightarrow x(t_k) + \Delta tb_1f + \Delta tb_2f + \Delta t^2ab_2$$

$Equation\ (3)\ from\ question$

$$x(t_{k+1}) = x(t_k) + \Delta tf(x(t_k), u_k) + \frac{\Delta t^2}{2} f(x(t_k), u_k) + \vartheta(\Delta t^3)$$

$After\ calculation\ and\ substitution\ (e_k = x_{K+1} - x(t_{k+1}) = \vartheta(\Delta t^3))$

$$e_k = (x(t_k) + \Delta tb_1f + \Delta tb_2f + \Delta t^2ab_2) - (x(t_k) + \Delta tf(x(t_k), u_k) + \frac{\Delta t^2}{2} f(x(t_k), u_k) + \vartheta(\Delta t^3)) = \vartheta(\Delta t^3)$$

$Consider\ \Delta tf = 1$

$$e_k = (x(t_k) + (b_1 + b_2) + \Delta t^2ab_2ff_x) - (x(t_k) + \Delta tf + \frac{\Delta t^2}{2} ff_x + \vartheta(\Delta t^3)) = \vartheta(\Delta t^3)$$

$After\ further\ calculating\ we\ get$
$(b_1 + b_2)\Delta tf = \Delta tf \rightarrow b_1 + b_2 = 1$
and

$$\Delta t^2ab_2ff_x = \frac{\Delta t^2}{2} ff_x \rightarrow ab_2 = \frac{1}{2}$$

$$\therefore The\ conditions: b_1 + b_2 = 1, ab_2 = \frac{1}{2}$$

b) RK2 methods are at the best of order 2. How does that relate to $e_k = \vartheta(\Delta t^3)$

$We\ consider\ the\ global\ error:$

$$e_k = \|x_n - x(t_f)\| = \vartheta(N^{\Delta t^3}) = \vartheta\left(\frac{t_f}{\Delta t}\Delta t^3\right) = \vartheta(t^2)$$

$The\ global\ error\ is\ of\ order\ 2, which\ states\ that\ we\ have\ a\ RK2.$

c) For what values of n,b,c, the RK2 is exact on?

From assignment question

$$x(t) = x(t_k) + \sum_{i=1}^{n} \frac{a_i}{i!}(t - t_k)^i$$

For RK2:

$$K_1 = f(x(t_k), u(t_k), t_k) = \sum_{i=1}^{n} \frac{a_i}{i!} i(t - t_k)^{i-1}$$

But $(t - t_k) = 0$, *so* $K_1 = 0$

$$K_2 = f(x(t_k) + a\Delta t K_1, u(t_k + c\Delta t), t_k + c\Delta t) = \sum_{i=1}^{n} \frac{a_i}{i!} i(t_k + c\Delta t - t_k)^{i-1} = \sum_{i=1}^{n} \frac{a_i}{i!} i(c\Delta t)^{i-1}$$

$$x_{k+1} = x(t_k) + \Delta t(b_1 K_1 + b_2 K_2) = x(t_k) + \Delta t b_1 K_1 + \Delta t b_2 [\sum_{i=1}^{n} \frac{a_i}{i!} i(c\Delta t)^{i-1}]$$

$$(x(t_{k+1}) - x_{k+1}) = x(t_k) - x(t_k) + \sum_{i=1}^{n} \frac{a_i}{i!} (t_{k+1} - t_k)^i - \Delta t b_2 \sum_{i=1}^{n} \frac{a_i}{i!} i(c\Delta t)^{i-1}$$

After evaluation and by substitution:

$$(n = 1, c = 1, b = 1) \rightarrow (x(t_{k+1}) - x_{k+1}) = 0$$
$$(n = 2, c = 1, b = 1) \rightarrow (x(t_{k+1}) - x_{k+1}) = 0$$

• Question 2 Accuracy and stability

a) Code an explicit Euler scheme (RK1) and an RK2 and RK4

Let $\dot{x} = \lambda x$ *and* $\lambda = -2, x_0 = 0, \Delta t = 0.1$

exact solution: $x(t) = x(0)e^{\lambda t}$

The explicit Euler method used in the Matlab code is shown below:

$$\dot{x} = \mathrm{f}(x_k, u_k)$$

$$x_{k+1} = x_k + \Delta t \cdot \mathrm{f}(x_k, u_k)$$

$$t_{k+1} = t_k + \Delta t$$

By plotting the t_{k+1} as x-axis and x_{k+1} as y-axis, we obtained the simulation plot for Euler method as shown in Figure 2.

Whereas for Runge-Kutta Method, we could use the formula for General Runge-Kutta Method to be applied for the order 2 and order 4 problem as shown below:

$$k_1 = f(x_k + \Delta t \sum_{j=1}^{s} a_{ij}k_j\,, u(t_k) + c_1, \Delta t)$$
$$\dots$$
$$k_i = f(x_k + \Delta t \sum_{j=1}^{s} a_{ij}k_j\,, u(t_k) + c_i, \Delta t)$$
$$\dots$$
$$k_s = f(x_k + \Delta t \sum_{j=1}^{s} a_{sj}k_j\,, u(t_k) + c_s, \Delta t)$$
$$\{$$
$$x_{k+1} = x_k + \Delta t \sum_{i=1}^{s} b_i k_i\,, t_{k+1} = t_k + \Delta t$$

k_1, k_i, k_s *could be interpreted as the* 1*st*, *i th*, *s th stages of the method iteration*

x_{k+1} *could be interpreted as iteration of the approximation method*

The easiest way to solve integration method using Runge-Kutta method is by using the Butcher's Tableau. Butcher Tableau defines the summarization of Runge-Kutta (RK) method coefficients in a simple way. The Butcher Tableau is shown as below in Figure 1. For RK2, s is defined as 2. Moreover, for RK4, s is defined as 4. Moreover, in order to code in Matlab, we could use the algorithm as shown in Appenix A.2.

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array}$$

Figure 1. Butcher's Tableau RK coefficients

Table 1: Explicit Euler

0	
	1

Table 2: Runge-Kutta 2

0		
$\frac{1}{2}$	$\frac{1}{2}$	
	0	1

Table 3: Runge-Kutta 4

0				
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{1}{2}$		$\frac{1}{2}$		
1			1	
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$

Figure 2. Coefficient for this problem for various methods

Consequently, we could obtain the simulation plot for Runge-Kutta 2nd order and 4th order by defining t_{k+1} as x-axis and x_{k+1} as y-axis.

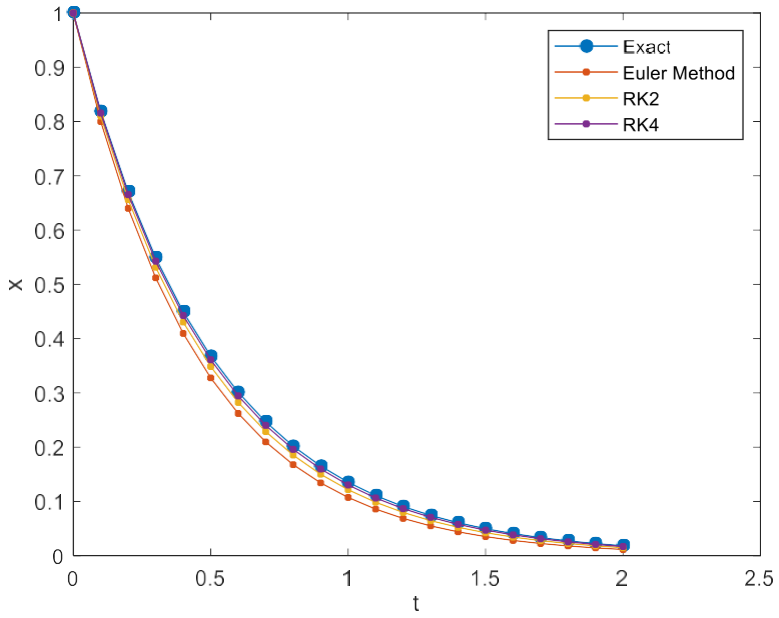


Figure 2. Simulation Result using Euler and Runge-Kutta (RK) Method vs True Solution

Observing Figure 2, we could see that RK4 method has the best accuracy against the Exact value followed by RK2 method. This is shown by the iteration of each time step which is particularly closest to exact value for RK4.

a) Compare the accuracy against the true solution

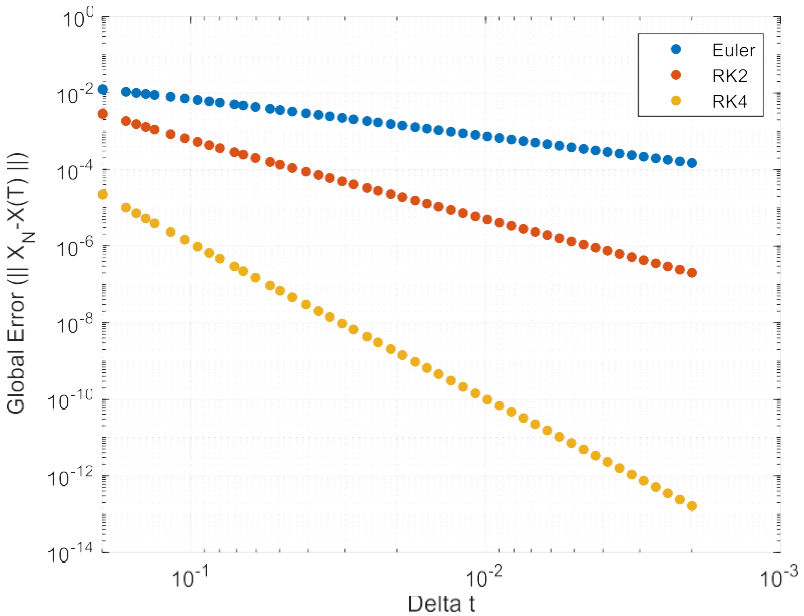


Figure 3. Accuracy Comparison between Euler and Runge-Kutta Method

Table 1. Simulation vs Theoretical Global Error Order

Method	Simulation Order of Accuracy (Δt^{ϑ})	Theoretical Order of Accuracy (Δt^{ϑ})
Euler	1.420	1
RK2	2.485	2
RK4	4.744	4

Based on Figure 3, we could observe that

$$|| \mathrm{X}_N - X(T) ||_{RK4} < || \mathrm{X}_N - X(T) ||_{RK2} < || \mathrm{X}_N - X(T) ||_{Euler}$$

$$|| \mathrm{X}_N - X(T) ||_{method} \leq \mathrm{c}\Delta t^{\vartheta}$$

This means that Global Error of Runge-Kutta 4 is the lowest in the simulation result.

Moreover, based on Table 1, we could observe that simulation order of accuracy seems to be better than theoretical order of accuracy. This means that the simulation produces error which is good enough since the simulation order is higher than the theoretical order. The simulation order of accuracy could be calculated by measuring the slope between global error and the Δt. Then the lowest value of the order

c) For what values of λ<0 will the different schemes become unstable?

For Euler method, we could approximate x_{k+1} with the equation below

$$x_{k+1} = x_k + \lambda \Delta t x_k = x_k(1 + \lambda \Delta t) \ldots . (1)$$

(1) will becomes unstable if $|1 + \lambda \Delta t|$>1. Consequently, since we have the value of $\lambda < 0$ and $\Delta t > 0$, the integration will be unstable if $1 + \lambda \Delta t > -1 \rightarrow \lambda \Delta t > -2 \rightarrow \Delta t > -2/\lambda$

For the Runge-Kutta integration method,

$$x(t_{k+1}) = x_k + \Delta t \cdot \dot{x}(t)|_{tk} + \frac{\Delta t^2}{2} \cdot \ddot{x}(t)|_{tk} + \cdots + \frac{\Delta t^{\vartheta}}{\vartheta!} \cdot x^{\vartheta}|_{tk} + \vartheta(\Delta t^{\vartheta+1})$$

Applied to the problem $\dot{x} = \lambda x$ with exact solution $x(t) = x(0)e^{\lambda t}$

$$x_{k+1} = (1 + \lambda \Delta t + \cdots + \frac{(\lambda \Delta t)^{\vartheta}}{\vartheta!})\, x_k + \vartheta(t^3)$$

Hence the stability could be defined for RK2 and RK4 respectively below

$$\left|1 + \lambda \Delta t + \frac{(\lambda \Delta t)^2}{2!}\right| < 1 \; [RK2]$$

Hence, it will become unstable if $\lambda \leq -20$ assuming that $\Delta t = 0.1$ and $\lambda < 0$ through mathematical calculation. Moreover, it is proven through matlab simulation as well in Appendix A.1

$$\left|1 + \lambda \Delta t + \frac{(\lambda \Delta t)^2}{2!} + \frac{(\lambda \Delta t)^3}{3!} + \frac{(\lambda \Delta t)^4}{4!}\right| < 1 \; [RK4]$$

In the case of RK4, it will become unstable if $\lambda \leq -27.85$ assuming that $\Delta t = 0.1$ and $\lambda < 0$ through mathematical calculation. Moreover, it is proven through matlab simulation as well in Appendix A.1.

- # Question 3 Van-der-Pol oscillator

a) Simulate and Observe the dynamics using Matlab integrator ODE45

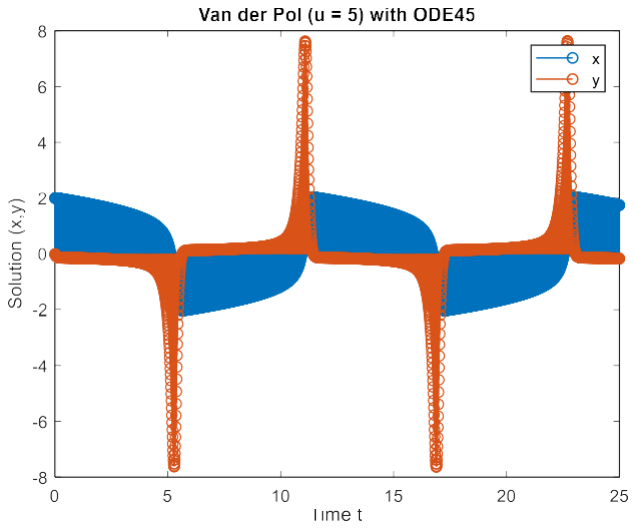


Figure 4. Van Der Pol Plot with ODE 45

By observing Figure 4, we could see that the solution using the adaptive integrator function ODE45 in matlab will result in variation of time step size by making the large step size during slow dynamics while on the other hand using small time step size during fast dynamic in order to maintain stability of the simulation. Hence during fast dynamic, we could see the slope increases rapidly as we very small time step. In order to limit the computational expenses at particular RK step which requires smaller time step, the system integrator uses two Butcher tableau.

b) Compare the solution with your own RK4 scheme

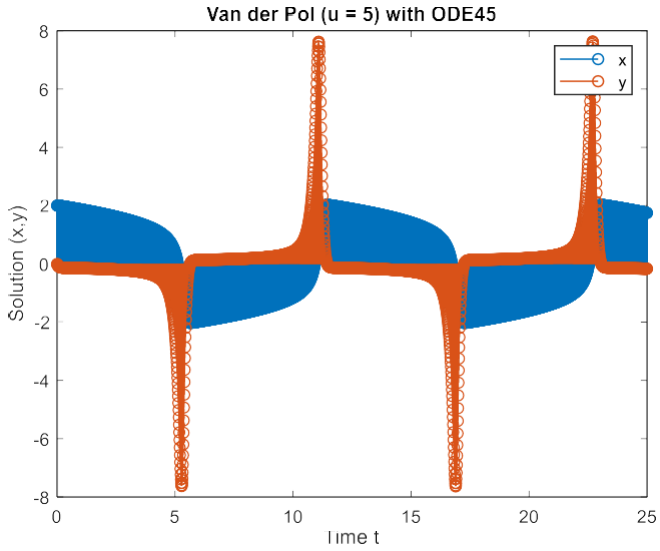


Figure 5 (a). Van der Pol Oscillator solution with ODE45 with $\Delta t = 10^{-2}$

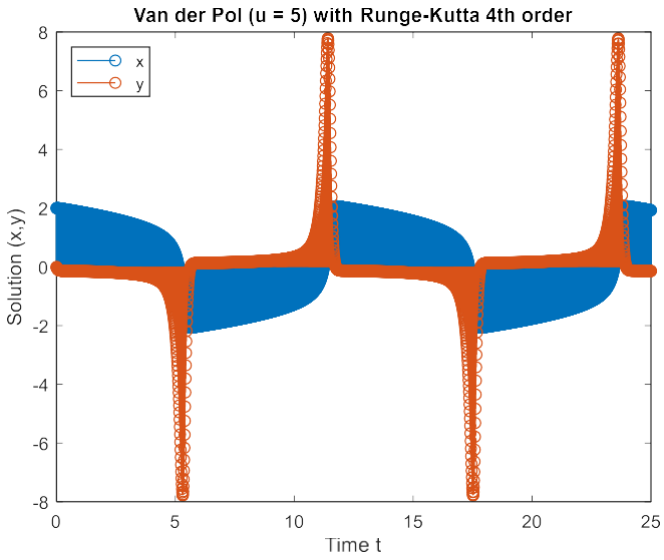


Figure 5 (b). Van der Pol Oscillator solution with manual RK4 method with $\Delta t = 10^{-2}$

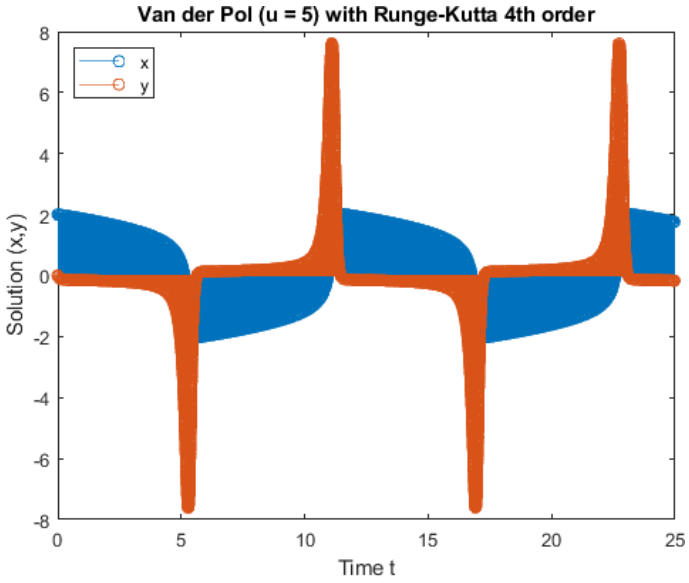


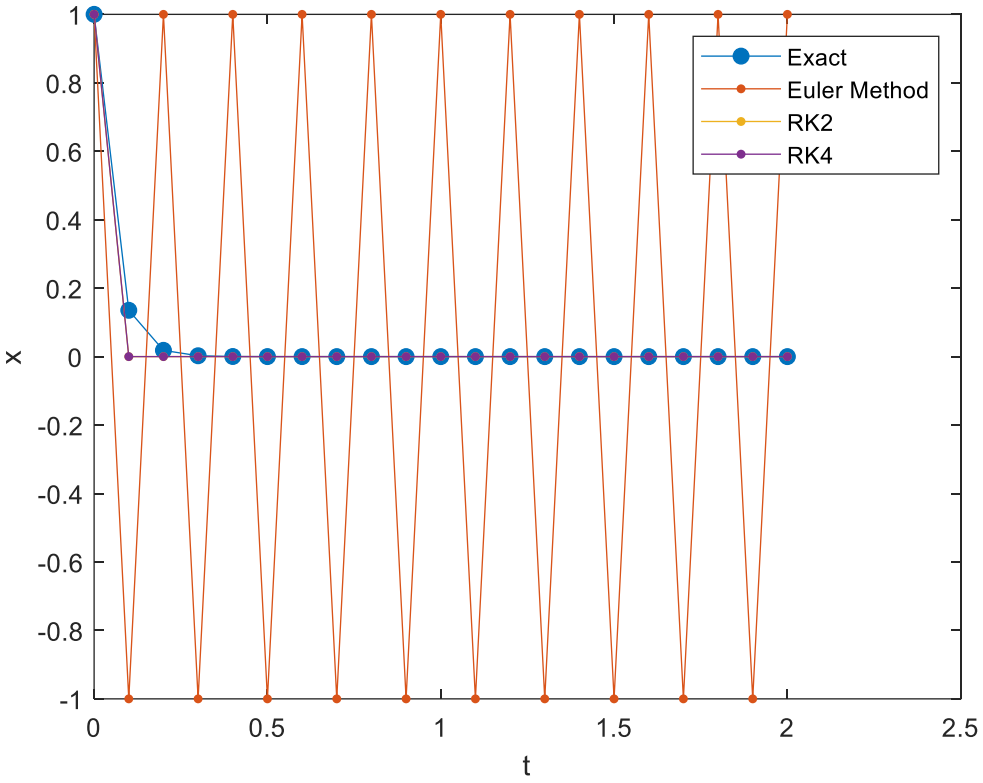
Figure 5 (c). Van der Pol Oscillator solution with manual RK4 method with $\Delta t = 5x10^{-4}$

Table 2. Accuracy comparison between different time step

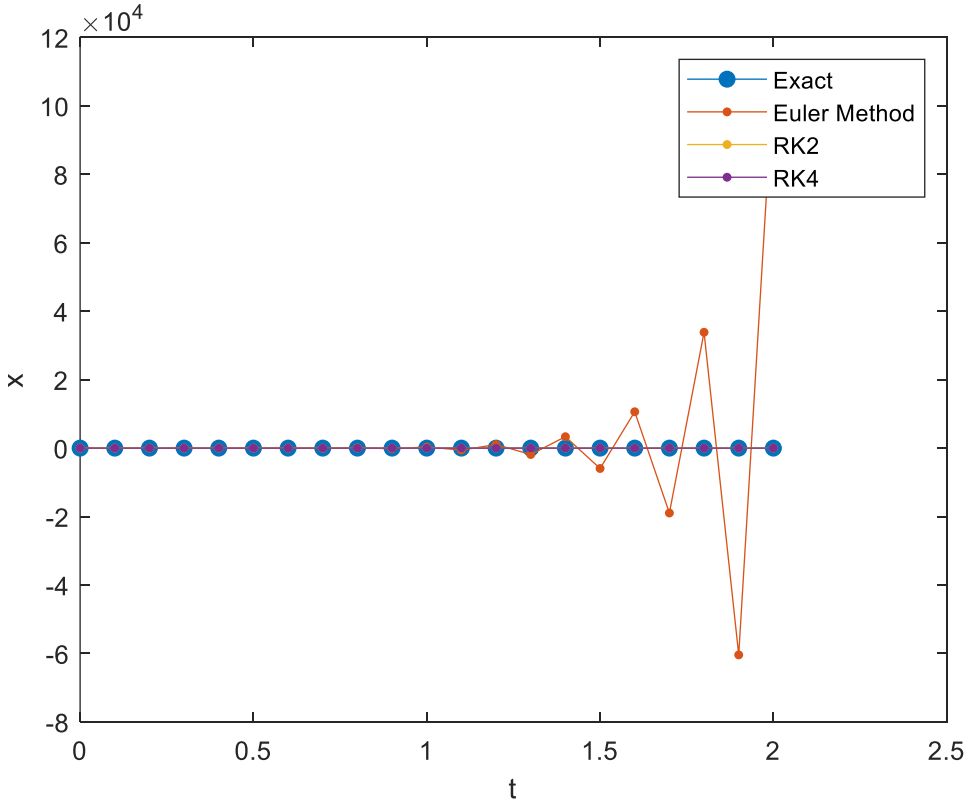
	Target ODE45	$\Delta t = 10^{-2}$	$\Delta t = 5x10^{-4}$
Max point of y	7.6399	7.7809	7.6279
Min point of y	-7.6509	-7.786	-7.6445

As can be observed from Figure 5 (a),(b),(c) as well as Table 2 above, it could be concluded that by using the manual Runge-Kutta method (RK4), in order to get the same accuracy as ODE45 function simulation result, we need much smaller time step. Hence, it will increase the computational time/cost, hence lower efficiency.

APPENDIX A.1 (RK2 and RK4 instability)



RK2 instability



RK4 Instability

APPENDIX A.2 (RK4 Algorithm)

For the sake of completeness, let us write the pseudo-code corresponding to an RK4 method.

Algorithm: RK4 method for explicit ODEs

Input: Initial conditions x_0, input profile $u(.)$, step size Δt

for $k = 0, \ldots, N-1$ **do**

Compute:

$$k_1 = f(x_k, u(t_k)) \tag{6.31a}$$

$$k_2 = f\left(x_k + \frac{\Delta t}{2} k_1,\ u\left(t_k + \frac{\Delta t}{2}\right)\right) \tag{6.31b}$$

$$k_3 = f\left(x_k + \frac{\Delta t}{2} k_2,\ u\left(t_k + \frac{\Delta t}{2}\right)\right) \tag{6.31c}$$

$$k_4 = f(x_k + \Delta t k_3,\ u(t_k + \Delta t)) \tag{6.31d}$$

Assemble the RK step

$$x_{k+1} = x_k + \Delta t \left(\frac{1}{6} k_1 + \frac{1}{3} k_2 + \frac{1}{3} k_3 + \frac{1}{6} k_4\right) \tag{6.32}$$

return $x_{1,\ldots,N}$