P346 Project Report
Adaptive Step Size Runge-Kutta Method

Ayush Singhal[1, *]

[1] *School of Physical Sciences, National Institute of Science Education and Research, HBNI, Jatni, Khordha, Odisha-752050, India*

(Dated: December 1, 2022)

The classic Runge-Kutta method (RK4) is a very useful tool in finding numerical solutions to ordinary differential equations accurately, while having a reasonable computational expense. This, along with other numerical methods, were developed around 1900 by Carl Runge and Wilhelm Kutta. However, the method's efficiency is heavily dependent upon the step size it uses. In this project, a version of the RK4 method that can appropriately change its step size, given a specific tolerance value, is discussed in detail.

I. INTRODUCTION

The RK4 method (multiple-step 4th order Runge-Kutta) is found to have no flexibility while it performs computations. The step size, h, once chosen, remains constant during the execution of code for a given problem. There are issues in obtaining accurate solutions when the step size is either too large, or too small.

We know from our coursework in P346 that choosing smaller step sizes and higher order methods are two possible routes to reducing the size of the error we get in our solution. However, this is not always the case (say, in an N-body problem). This issue can sometimes be solved by being able to dynamically change the step size for our numerical differential equation solver, when given a fixed amount of accuracy for our solution. We will now discuss such a method that allows us to do the same, which is known as the Runge-Kutta-Fehlberg method.

II. WORKING PRINCIPLE

The method that is described in Fehlberg's 1969 paper ([1]) is novel due to the reason that it is an 'embedded' method from the RK family of methods. In embedded methods, identical function evaluations are used together to create methods of multiple orders and similar error constants. Fehlberg's algorithm (also known as the RKF45 method) is a 4th order method, but it has a 5th order error estimator. With the performance of an extra calculation, the solution's error can be estimated and regulated accordingly by using the higher-order embedded method that allows for a step size to be determined dynamically.

III. DERIVATION

In general, $(p-1)^{th}$ order adaptive methods require a p^{th} order step in order to compute the local truncation error for a step in question. Since these two steps occur in a single, interwoven step for a loop, a lower computational cost is entailed that what would've been entailed for a method of a higher order.

During the execution of the integration, the step size is changed in such a way so as to keep the estimated error below a pre-determined threshold. If the error is higher than permissible, the step is repeated with a lower step size. if the error is small, the step size is increased to reduce the computational cost. This results in the maintenance of an optimal step size, which saves time spent in computation. Hence, additional effort is not needed to find an appropriate step size for a given problem.

The higher order step is computed as,

$$y_{n+1} = y_n + h\Sigma_{i=1}^{s} b_i k_i \tag{1}$$

We also know that the lower order step is computed as,

$$y^{'}_{n+1} = y_n + h\Sigma_{i=1}^{s} b^{'}_i k_i \tag{2}$$

Here, k_i are the same as for the higher order method. Hence, we get the error between the two for the (n+1)th step as,

$$e_{n+1} = y_{n+1} - y^{'}_{n+1} = h\Sigma_{i=1}^{s}(b^{'}_i - b_i)k_i \tag{3}$$

And here, e is the higher order estimator. For this family of adaptive methods, it is seen that the general Butcher tableau is given as,

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{2,1} & & & & \\ c_3 & a_{3,1} & a_{3,2} & & & \\ . & . & . & & & \\ . & . & & . & & \\ . & . & & & . & \\ c_s & a_{s,1} & a_{s,2} & \dots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \dots & b_{s-1} & b_s \\ & b^{'}_1 & b^{'}_2 & \dots & b^{'}_{s-1} & b^{'}_s \end{array}$$

* ayush.singhal@niser.ac.in

Hence, we can write out the Butcher tableau for RKF45 as,

$$\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{4} & \frac{1}{4} & & & & & \\
\frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & & \\
\frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & \frac{7296}{2197} & & & \\
1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & & \\
\frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} & \\
\hline
 & \frac{16}{235} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{1}{6} \\
 & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0
\end{array}$$

These coefficients are applied in the code to find the increments and the summations of the weights for the increments, which is then used to compute the appropriate adjustment for the step size.

The truncation error (E), which is the 5th order estimator, is calculated using the coefficients obtained in the embedded pair (the lower two rows, or weights) as,

$$E = |\Sigma_{i=1}^{6}(b_i^{'} - b_i)k_i| \tag{4}$$

From the truncation error, we compute the new step size as,

$$h_{new} = kh\left(\frac{\epsilon}{E}\right)^{\frac{1}{4}} \tag{5}$$

Here, the k factor can be between 0.8 and 1.0, depending upon the amount of fluidity required in changing the step size. If $E > \epsilon$, then we replace h with h_{new} and repeat the step. But if $E \leqslant \epsilon$, then the step is completed successfully and we replace h with h_{new} for the following step.

IV. TEST SCENARIOS

Now, we will have a look at the efficiency and accuracy that our adaptive step alogirthm is able to provide for some scenarios where first order derivatives are known, along with some boundary conditions.

A. Solution for a first-order ODE

We pick a differential equation whose solution curve has at least one sharp bend.

$$\frac{dy}{dt} = t^2 - \frac{3y}{t} \tag{6}$$

The general solution to the differential equation is obtained as,

$$y = \frac{t^3}{6} - \frac{C}{t^3} \tag{7}$$

Solving for $y(1) = \frac{1}{2}$, we see that $C = \frac{1}{3}$. Hence, the solution curve is,

$$y = \frac{t^3}{6} - \frac{1}{3t^3} \tag{8}$$

We run our test by setting the boundary condition as $y(0.1) = 333.33$ (which corresponds quite closely to $C = \frac{1}{3}$). We compare our obtained solution to values of y from equation (3).

B. Earth Hole problem

We take a common classical mechanics problem for our 2nd scenario. Let us imagine a hole that passes through the diameter of the Earth. While standing near this hole, if we drop a light, small body (compared to the Earth) into this hole, how long will it take to return to the point it was released from?

We start by writing the differential equation for the acceleration that is experienced the body that is to be dropped.

$$\frac{d^2y}{dt^2} = -\frac{G.M(y)}{y^2} \tag{9}$$

Here, G is the gravitational constant, y is the position of the body with respect to the centre of the Earth, and $M(y)$ is the mass which pulls on the body (which can be seen to be a function of y). We see that this mass is given as,

$$M(y) = \rho\frac{4}{3}\pi y^3 = \frac{M_{Earth}}{\frac{4}{3}\pi R_{Earth}^3}\frac{4}{3}\pi y^3 = \frac{M_{Earth}y^3}{R_{Earth}^3} \tag{10}$$

Here, ρ is the density of Earth, M_{Earth} is the mass of the Earth and R_{Earth} is the radius of the Earth, which is 6370 km $= 6.37 \times 10^6$ m.

Putting equation (5) into equation (4), we get,

$$\frac{d^2y}{dt^2} = -\frac{G}{y^2}\frac{M_{Earth}y^3}{R_{Earth}^3} = -g\frac{y}{R_{Earth}} \tag{11}$$

Here, g is gravitational acceleration at the surface of the Earth (9.81 ms^{-2})

Hence, it is seen that the motion of the body follows the same differential equation as a simple harmonic oscillator (SHO). Accordingly, we can write the solution, the first and second derivatives with respect to time, and also the time period of oscillation - which would also be the amount of time the body would take to complete it's journey back to the side of the hole it was released from.

We write y and its derivates with boundary conditions in mind.

$$y = R_{Earth}cos(\omega t) \tag{12}$$

$$\frac{dy}{dt} = -R_{Earth}\omega sin(\omega t) \tag{13}$$

$$\frac{d^2y}{dt^2} = -R_{Earth}\omega^2 cos(\omega t) \tag{14}$$

Since we know that maximum acceleration is g, we can say that the value of ω^2 should be $\frac{g}{R_{Earth}}$. Therefore, we get the time period of oscillation as

$$T = 2\pi\sqrt{\frac{R_{Earth}}{g}} \simeq 5063\ s \tag{15}$$

We apply our method on the first derivative (8) and we compare our solution with the solution curve (7) to see how well the computed solution holds up against the analytical prediction (10).

C. Predator-Prey system

The earlier scenarios had problems where an analytical solution could actually be obtained. However, a numerical method can only be truly tested by its accuracy in cases where analytical solutions are unobtainable. Hence, we invoke the case of a predator-prey model where the rate of population change for two species (let's say, X and Y) with a prey-predator relationship is dictated by the following two differential equations,

$$\frac{dx}{dt} = ax - \alpha xy \tag{16}$$

$$\frac{dy}{dt} = -cy + \gamma xy \tag{17}$$

Here, x is the population of species X (the prey species), and y is the population of species Y (the predator species). a is known as the prey birth rate and c is known as the predator death rate, respectively. While α and γ can be called as the 'interaction coefficients' for the two species.

From observation, it can be seen that the two differential equations can not be separated from each other, i.e., they are coupled differential equations. As per the literature, this system of equations behaves in an oscillatory manner, but without a fixed period. We qualitatively examine the behaviour of this period in our analysis.

V. RESULTS

A. Solution for a first-order ODE

The algorithm performs well in this case. It responds to the problem by shortening the step size at the inflection point. It also gradually expands the step size as the curve moves away from the inflection point in the graph of the function. At the end, it is seen that the step size contracts to meet the limit of $t = 10$ that has been set in the parameters for the code.

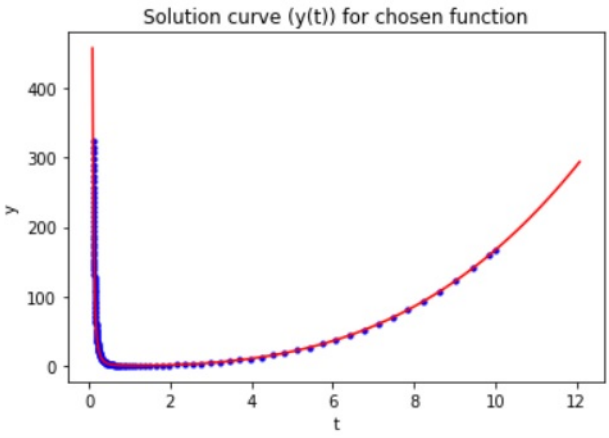

FIG. 1. Solution in Scenario 1 using RKF45

B. Earth Hole problem

It was seen that the computed solution is highly accurate as compared to the analytical solution. The modification made to the code successfully allows for the prevention of the division by zero error when Δ is to be calculated. We also successfully verify the solution to the problem that we obtained while obtaining the analytical solution - it takes almost 5063 seconds for the body dropped from one end of the Earth to return to the same end again. The order of error is seen to be less than 1 part in 10^5 if the position of the body at $t = 5063$s is checked.

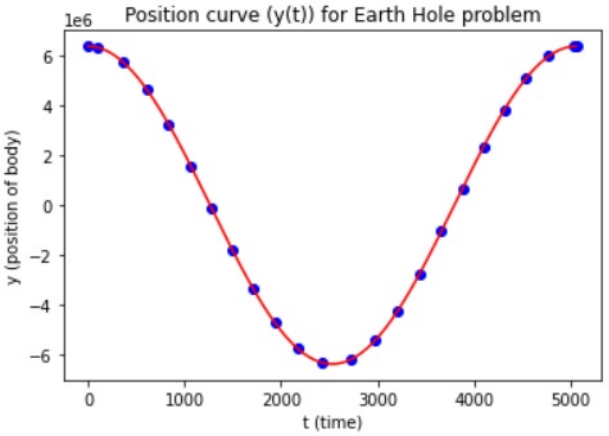

FIG. 2. Solution in Scenario 2 using RKF45

Again, we observe that the step size changes marginally to become smaller when reaching the other side of the Earth, since the position curve experiences an inflection point during that time. It remains large when the body is moving the fastest (near the centre of the Earth).

C. Predator-Prey system

Similar to [2], the birth rate (a) and death rate (c) parameters were varied and the results were recorded for various cases. It was seen that the maximum populations of the prey and predator species increased dramatically with each "period", as opposed to a much slower increase. It is difficult to tell if this is a more accurate or a less accurate solution that the one provided in Juarlin's paper, since no analytical solution exists for this system of equations. However, we can clearly see how the step size changes multiple times during a single period to accommodate for both the prey and predator solution curves. In all the cases seen below, t was initialised at $t = 0$ and terminated at $t = 500$, taking an execution time of about a second.

First (Figure 3), we set a, c, α, γ, $x(t = 0)$, and $y(t = 0)$ as 0.1 to get a base case with which to compare our other results, as was done in [2]. Here, we see the same sort of oscillatory behaviour as seen in [2], but with a key difference - the maximum population and the period of oscillation keep increasing as time progresses.

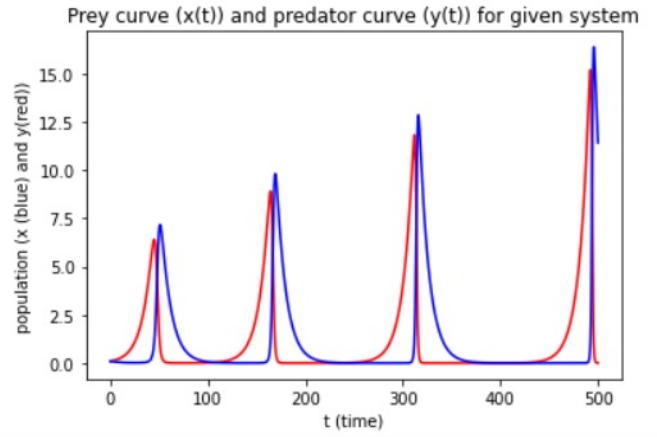

FIG. 3. Scenario 3 with $a = 0.1$ and $c = 0.1$

Next (Figure 4), we set a as 0.3 and c as 0.2. We can see that this results in a much smaller period for the oscillation of the population curves, as the prey species is produced much quicker while the predator is also eliminated quicker (leading the prey species to proliferate).

Finally (Figure 5), we look at a system where the predators die rapidly ($c = 0.3$) while the prey species are born at the same rate as the first case ($a = 0.1$). We see that a large amount of prey animals are needed before the predator population rises significantly.

It is ultimately difficult to comment on the accuracy of these results, as not much literature exists on this particular application of the RKF45 algorithm.

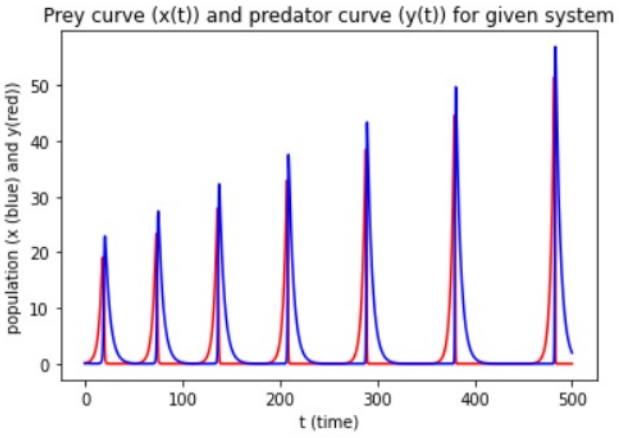

FIG. 4. Scenario 3 with $a = 0.3$ and $c = 0.2$

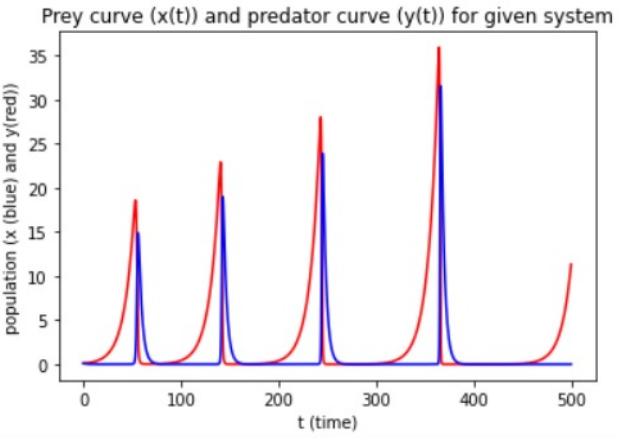

FIG. 5. Scenario 3 with $a = 0.1$ and $c = 0.3$

VI. CONCLUSION

We hence come to the conclusion that the Runge-Kutta-Fehlberg modification to the original RK4 algorithm offers some valuable computational discounts in some cases, while resulting in only a marginally higher processing time per step. It is a wise idea to compare the performance for such numerical algorithms before using them on much larger data sets, to check the suitability for a given algorithm to a particular problem.

[1] E. Fehlberg, "Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems", NASA Technical Reports, TR R-315 (1969).

[2] E. Juarlin, "Solution of Simple Prey-Predator Model by Runge-Kutta Method", J. Phys.: Conf. Ser **1341** 062024 (2019).