DIFFERENTIAL EQUATIONS
NUMERICAL METHODS 2

RUNGE KUTTA 4

The most widely known member of the Runge–Kutta family is generally referred to as "RK4", the "classic Runge–Kutta method" or simply as "the Runge–Kutta method". The '4' refers to the estimated error of the order of h^4. Again, RK4 will only solve 1st order ODE, with initial values, although there is a technique to transpose a 2nd degree ODE to a 1st. See p6 of this paper. Let an initial value problem be specified as follows:

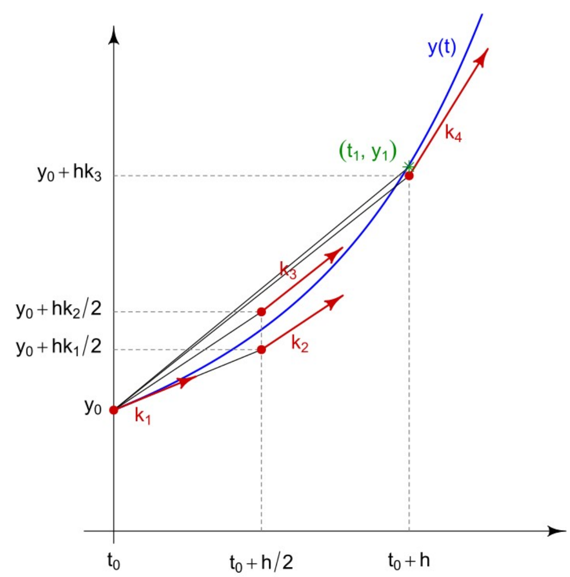
$$\frac{dy}{dt} = f(t,y), \quad y(t_0) = t_0$$

RK4 basically computes the next value y_{n+1} using the current y_n plus the weighted average of four increments.

Here is an unknown function (scalar or vector) of time, which we would like to approximate. We are told that, $\frac{dy}{dt}$, the rate at which it changes, is a function of t and of y itself. At the initial time t_0 the corresponding y value is y_0. The function and the initial conditions, are given. Now pick a step-size *h* > 0 and define:

$$y_{n+1} = y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4)$$
$$t_{n+1} = t_n + h$$

1. k_1 is the slope at the beginning of the time step (this is the same as k_1 in the first order method, Euler).
2. If we use the slope k_1 to step halfway through the time step, then k_2 is an estimate of the slope at the midpoint. This is the same as the slope, k_2, from the second order midpoint method. This slope proved to be more accurate than k_1 for making new approximations for y(t).
3. If we use the slope k_2 to step halfway through the time step, then k_3 is another estimate of the slope at the midpoint.
4. Finally, we use the slope, k_3, to step all the way across the time step (to t_0+h), and k_4 is an estimate of the slope at the endpoint.



We then use a weighted sum of these slopes to get our initial estimate of $y(t_0+h)$.

$$y(t_0+h) = y(t_0)\frac{k_1+2k_2+2k_3+k_4}{6}h \;=\; y(t_0) + \left(\tfrac{1}{6}k_1 + \tfrac{1}{3}k_2 + \tfrac{1}{3}k_3 + \tfrac{1}{6}k_4\right) \;=\; y(t_0) + mh$$

(RK 0, p23).

This method gives surprisingly accurate results, without the need of using extremely small values of h.

$k_1 = (h)f(x_0,y_0),$

$k_2=(h)f\left(x_0+\frac{h}{2},y_0+\frac{k_1}{2}\right),$

$k_3=(h)f\left(x_0+\frac{h}{2},y_0+\frac{k_2}{2}\right),$

$k_4 = (h)f(x_0 + h,y_0 + k_3),$

$K= \left(\frac{1}{6}\right)(k_1+2k_2+2k_3+k_4).$

Now we set $y_1 = y_0 + K$ (05)

Having determined y_1, we proceed to approximate $x_2 = x_1 + h$ in the same way.

$k_1 = (h)f(x_1,y_1),$

$k_2=(h)f\left(x_1+\frac{h}{2},y_1+\frac{k_1}{2}\right),$

$k_3=(h)f\left(x_1+\frac{h}{2},y_1+\frac{k_2}{2}\right),$

$k_4 = (h)f(x_1 + h,y_1 + k_3),$

$K= \left(\frac{1}{6}\right)(k_1+2k_2+2k_3+k_4).$

Now we set $y_2 = y_1 + K$

And take this as the approximate value of the exact solution at $x_2 = x_1 + h$.

Continue the same way until the final h is reached. (Ross, 456).

*Note that this uses "x" for the independent value. This is because this section is copied from my math text rather than engineering notes.

EXAMPLE 03

$y' = y - t^2 + 1$, $y(0) = 0.5$

The exact solution for this problem is $y(x) = t^2+2t+1-\frac{1}{2}e^t$, and we are interested in the value of y for $0 \le t \le 2$.

We solve this using RK4 with h = 0.5, from t = 0 to t = 2.

$t_0 = 0$, $y_0 = 0.5$

$t_1 = 0.5$

$k_1 = (h)f(t_0,y_0) = (0.5)(f(0,0.5)) = (0.5)(1.5) = 0.75$

$k_2 = (h)f\left(t_0+\frac{h}{2},y_0+\frac{k_1}{2}\right) = (0.5)(f(0.25,0.875)) = (0.5)(1.8125) = 0.90625$

$k_3 = (h)f\left(t_0+\frac{h}{2},y_0+\frac{k_2}{2}\right) = (0.5)(f(0.25,0.953125)) = (0.5)(1.890625) = 0.9453125$

$k_4 = (h)f(t_0+h,y_0+k_3) = (0.5)(f(0.5,1.4453125)) = (0.5)(2.1953125) = 1.09765625$

$y_1 = y_0 + (k_1 + 2k_2 + 2k_3 + k_4) / 6 =$

$0.5 + (0.75 + 0.90625 + 096453125 + 1.09765625) / 6 = 1.42513020833333$

$t_2 = 1$

$k_1 = (h)f(t_1,y_1) = (0.5)(f(0.5,1.4251302083333333)) = (0.5)(2.175130208333333) =$
1.0875651041666667

$k_2 = (h)f\left(t_1+\frac{h}{2},y_1+\frac{k_1}{2}\right) = (0.5)(f(0.75,1.968912760416667) =$
$(0.5)(2.40641276047) = 1.203206380208333$

$k_3 = (h)f\left(t_1+\frac{h}{2},y_1+\frac{k_2}{2}\right) = (0.5)(f(0.75,2.0267333984375) =$
$(0.5)(2.46423339844) = 1.23211669921875$

$k_4 = (h)f(t_1+h,y_1+k_3) = (0.5)(f(1,2.657246907552083)) =$
$(0.5)(2.657246907552083) = 1.328623453776042$

$y_2 = = y_1 + (k_1 + 2k_2 + 2k_3 + k_4) / 6 = 2.639602661132812$

…

$t_4 = 4$
$y_4 = 1.378409485022227 + 1.316761856277783 + 1.301349949091673 + 1.154084459568063 = 5.301605229265987$

t (i)	Exact y(ti)	RK4 y(i)	Error |Exact-RK4|
0.0	0.5	0.5	0.0
0.5	1.42563936464993	1.42513020833333	0.000509156316603
1.0	2.64085908577047	2.63960266113281	0.001256424637665
1.5	4.00915546483096	4.00681897004445	0.002336494786515
2.0	5.30547195053467	5.30160522926598	0.003866721268688

(RK 3, p1).

While smaller values of h provided better approximations with Euler, we can see that the same applies to RK4. However, where Euler uses a smaller h with a factor of 2, producing improved accuracy by 2, RK4 will reduce errors by a factor of 16! (Ross, 459). A simple way to determine whether your h value is small enough is to do the problem again with h/2. Then compare the results. I know that this is a lot of work. Perhaps both trials can be done with a small data sample. However, changing to a more precise method, might be better. See Felhberg and ABAM below.

Dawkins describes a method to transform a 2nd degree ODE to a 1st degree. See Dawkins 276 PDF, also p6 of this paper. There are several other PDF describing this technique in Methods folder.

FELHBERG

The trickiest part of RK is determining the right step size. Many problems in celestial mechanics, chemical reaction kinematics, and other areas have long periods of time where nothing much is happening (and for which large step-sizes are appropriate) mixed in with periods of intense activity where a small step-size is vital. What we need is an algorithm which includes a method for choosing the appropriate step-size at each step. The Runge-Kutta-Fehlberg methods do just this, which is why they have largely replaced the Runge-Kutta methods in practice.

Let us assume that for constant C′

$$\left|C'\right| \approx \frac{|y_1' - y_2''|}{h^3}$$

Once we have this approximation for C′, we can pick a step-size h_1 to get the local error of the size we want. If we want the local error to be about size T, we just take a step-size h_{new} where

$$h_{new} = h\left(\frac{T}{|y_1' - y_2''|}\right)^{1/3}$$

You might be a little worried about how all the errors in the different approximations mount up as we carry out all these computations to get our new step-size . This is a serious consideration and is dealt with by introducing a chicken factor, usually taken to be 0.9. We actually use a step-size

$$h_1 = (0.9)h\left(\frac{T}{|y_1' - y_2''|}\right)^{1/3}$$

Fehlberg uses exactly this technique to pick the right step-size. Suppose the initial value problem we want to solve is

$$\frac{dy}{dx} = f(x,y), \quad y(x_0) = y_0$$

We have an initial step-size h (taken to be whatever value you fancy, we will update it automatically as needed). We compute the improved Euler and RK3 estimates in the usual fashion.

$$k1 = f(x_0, y_0)$$
$$k2 = f(x_0 + h, y_0 + (h)(k1))$$
$$k3 = f(x_0 + h/2, y_0 + (h)((k1 + k2)/4)$$
$$y'_1 = y_0 + (h)(k1 + k2)/2$$
$$y''_2 = y_0 + (h)\left(\frac{k1+k2+4k3}{6}\right)$$
$$|\text{error}| \approx |y'_1 - y''_2|$$

If this error is small enough, say within a tolerance of $T = (0.001)(\max(|y_0|,1))$, then we accept this step-size for the current step and let

$$x_1 = x_0 + h$$
$$y_1 = y''_1$$

If the error is greater than T, we reject this step-size for the current step and leave x_0 and y_0 as they are. In either case, we choose a new step-size

$$h_{new} = 0.9h\left(\frac{T}{|y'_1 - y''_2|}\right)^{1/3}$$

We then either compute the next step with the new step-size (if our error was less than T) or we repeat the current step with the new step-size (if the error was greater than T) and try again to find x_1 and y_1. (RKF2, 3).

EXAMPLE 04

Approximate y(1) if dy/dx = x + y, with y(0) = 0. Use Felhberg with tolerance T = 0.01. We need to pick an initial step-size to get things started. We have to go from $x_0 = 0$ to x = 1, so why not go for it all in one shot and guess initially h = 1. Choosing a step-size of about the length of the interval divided by 16 or32 is more typical. I wanted to be sure I had to reject the estimate sometime in the course of the example so I decided to start off wrong with too large a step size to be sure that happened. We carry out the following computations.

K1 = 0
K2 = 1
K3 = 0.75
$y'_1 = 0.5$
$y''_2 = 0.6666666666666\ldots$
$|y'_1 - y''_2| = 0.166666666666\ldots$

The estimated error is greater than the tolerance 0.01, so we reject the initial step-size of h = 1. We compute a new step-size and try again.

$h_{new} = (0.9)(1)(0.01/1.66666\ldots)^{1/3} = 0.3523380877$
K1 = 0
K2 = 0.35233880877
K3 = 0.2072045759
$y'_1 = 0.062071064$
$y''_1 = 0.069361064$
$|y'_1 - y''_1| = 0.00729 < 0.01$

This time the estimated error is less than the tolerance so we accept the step-size and estimate and compute

$x_1 = x_0 + h$
$y_1 = y''_1$

We now compute a new step-size and go on to the next step (Twice in this problem, we compute a new step-size and it turns out to exactly equal the old step-size. This is a freak accident.)

$h_{new} = (0.9)(0.3523380877)(0.01/0.00729\ldots)^{1/3} = 0.3523380877$

$K1 = 0.4216991517$
$K2 = 0.9276179121$
$K3 = 0.7162817215$
$y_2' = 0.3061881158$
$y_2'' = 0.3165523026$
$|y_2' - y_2''| = 0.013641868 > 0.01$
This time the estimated error is greater than the tolerance so we reject the step-size and try again with the same x_1 and y_1.
$h_{new} = (0.9)(0.3523380877)(0.1/0.0103641868)^{1/3}$
$K1 = 0.4216991515$
$K2 = 0.8671284185$
$K3 = 0.6793383478$
$y_2' = 0.2712937907$
$y_2'' = 0.2785837907$
$|y_2' - y_2''| = 0.00729 < 0.01$
This estimated error is less than the tolerance so we accept the step-size and compute
$x_2 = x_1 + h = 0.6656837532$
$y_2 = y_2'' = 0.2785837907$
We now compute the new step-size for the next step and repeat the process
$h_{new} = (0.9)(0.3133456655)(0.1/0.0729)^{1/3} = 0.3133456655$
$K1 = 0.9442675439$
$K2 = 1.553495351$
$K3 = 1.296606171$
$y_3' = 0.669915379$
$y_3'' = 0.679849358$
$|y_3' - y_3''| = 0.0099695568 < 0.01$
Estimated error is less than tolerance, so we accept and compute
$x_3 = x_2 + h = 0.9790294187$
$y_3 = y_3'' = 0.679849358$
Next step-size
$h_{new} = (0.9)(0.3133456655)(0.1/0.0099695568)^{1/3} = 0.2822978588$
But this step-size is too large since $x_3 + h = 1.261327277 > 1$, and so it would put us past our final value for x. Therefore we shrink to hit x = 1 exactly.
$h_{new} = 1 - x_3 = 0.0209705813$
$K1 = 1.658914354$
$K2 = 1.714673334$
$K3 = 1.687086169$
$y_4' = 0.7152579833$
$y_4'' = 0.7152620701$
$|y_4' - y_4''| = 0.00000408681 < 0.01$
This estimated error is less than the tolerance, so we accept this estimate and make the final computations
$x_4 = x_3 + h = 1$
$y_4 = y_4'' = 0.7152620701$
$y_{actual} = 0.7182818285$
(RKF2, 5).

EXAMPLE 05
$y' = y - t^2 + 1$ $y(0) = 0.5$ $t_f = 2$ $h = 0.2$ $T = 0.0001$
$y(x)_{actual} = c_1e^x + x^2 + 2x + 1$ (courtesy of Wolfram Alpha)
This technique must be run on a computer or calculator. So, I will just give the answer.

The paper (RK 3) that I copied this from yields intermediate values for t. Obviously making this suitable for graphing.

t	EXACT	NUMERICAL	ABS ERR
0.0	0.500000000000000	0.500000000000000	0.000000000000000
0.2	0.829298620919915	0.829293333333333	0.000005287586582
0.4	1.214087651179360	1.214076210666660	0.000011440512698
0.6	1.648940599804740	1.648922017041600	0.000018582763146
0.8	2.127229535753760	2.127202684947940	0.000026850805823
1.0	2.640859085770470	2.640822692728750	0.000036393041726
1.2	3.179941538631720	3.179894170232230	0.000047368399496
1.4	3.732400016577660	3.732340072854980	0.000059943722683
1.6	4.283483787802440	4.283409498318400	0.000074289484036
1.8	4.815176267793520	4.815085694579430	0.000090573214092
2.0	5.305471950534670	5.305363000692650	0.000108949842019

Note that the computer balked on many of the final digits, changing them to zeros.

(RK 3, p3#2).

CHANGING THE ORDER OF A DE

A system of differential equations can arise from a population problem in which we keep track of the population of both the prey and the predator. The differential equation that governs the population of either the prey or the predator should in some way depend on the population of the other. This will lead to two differential equations that must be solved simultaneously in order to determine the population of the prey and the predator.

However, systems can arise from nth order linear differential equations as well. Here is an example of two different first order (the order is the largest derivative present in the equation) linear differential equations (any DE with the form of y=mx+b or $\frac{d^2y}{dx^2} + 5\frac{dy}{dx}$ + 6y = 0. See my paper Diff Eq 0, p1-2). Here is a system of first order, linear DE.

$$x_1' = x_1 + 2x_2$$
$$x_2' = 3x_1 + 2x_2$$

Often we find linear DE of higher orders. Since the numerical methods we have covered are only suitable for first order DE, we need a method to convert a higher order to a first order.

EXAMPLE 06

Write the following DE as a system of first order linear DE.

$$2y'' - 5y' + y = 0 \qquad y(3) = 6 \qquad y'(3) = -1$$

This can be done with a simple change of variable. We'll start by defining two new functions.

$x_1(t) = y(t)$
$x_2(t) = y'(t)$

Notice that if we differentiate both sides of these we get,

$x_1(t) = y' = x_2$
$x_2' = y'' = -\frac{1}{2}y + \frac{5}{2}y' = -\frac{1}{2}x_1 + \frac{5}{2}x_2$

We can also convert the initial conditions to the new functions.

$x_1(3) = y(3) = 6$
$x_2(3) = y'(3) = -1$

Putting this all together gives the following:

$x_1' = x_2 \qquad\qquad x_1(3) = 6$

$x_2' = -\frac{1}{2}x_1 + \frac{5}{2}x_2 \qquad x_2(3) = -1$

EXAMPLE 07
Write this 4th order DE as a system of 1st order linear DE.

$y^{(4)} + 3y'' - \sin(t)\, y' + 8y = t^2 \quad y(0) = 1 \quad y'(0) = 2 \quad y''(0) = 3 \quad y'''(0) = 4$

Just as we did before, we'll need to define some new functions. This time we'll need 4 new functions.

$x_1 = y \quad \Rightarrow \quad x_1' = y' = x_2$

$x_2 = y' \quad \Rightarrow \quad x_2' = y'' = x_3$

$x_3 = y'' \quad \Rightarrow \quad x_3' = y''' = x_4$

$x_4 = y''' \quad \Rightarrow \quad x_4' = y^{(4)} = -8y + \sin(t)y' - 3y'' + t^2 = -8x_1 + \sin(t)x_2 - 3x_3 + t^2$

Then this system, along with the initial conditions is

$x_1' = x_2$

$x_2' = x_3$

$x_3' = x_4$

$x_4' = -8x_1 + \sin(t)x_2 - 3x_3 + t^2$

Now, when we finally get around to solving these, we will see that we generally don't solve systems in this form. Systems of DE can be converted to matrices, and this is the form that we use to solve them.

EXAMPLE 08
Convert the following system to matrix form.

$x_1' = 4x_1 + 7x_2$

$x_2' = -2x_1 - 5x_2$

First write the system so that each side is a vector,

$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 4x_1 & 7x_2 \\ -2x_1 & -5x_2 \end{bmatrix}$

Now the RS can be written as a matrix multiplication,

$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 4 & 7 \\ -2 & -5 \end{bmatrix}$

Define:

$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

Then

$\vec{x}' = \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}$

Repeat this process for examples 06 and 07.

EXAMPLE 09
For example 06:

$x_1' = x_2 \qquad x_1(3) = 6$

$x_2' = -\frac{1}{2}x_1 + \frac{5}{2}x_2 \qquad x_2(3) = -1$

First define,

$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

This system is then,

$\vec{x}' = \begin{bmatrix} 0 & 1 \\ -\frac{1}{2} & \frac{5}{2} \end{bmatrix} \qquad \vec{x}(3) = \begin{bmatrix} x_1(3) \\ x_2(3) \end{bmatrix} = \begin{bmatrix} 6 \\ -1 \end{bmatrix}$

EXAMPLE 10
For example 07:

$x_1' = x_2$ $\qquad x_1(0) = 1$
$x_2' = x_3$ $\qquad x_2(0) = 2$
$x_3' = x_4$ $\qquad x_3(0) = 3$
$x_4' = -8x_1 + \sin(t)x_2 - 3x_3 + t^2$ $\qquad x_4(0) = 4$

Now, we have to be careful with the t^2 in the last equation. We'll start by writing the system as a vector again. Then we'll break it into two vectors. One will contain the unknown functions, the other, the known functions.

$$\begin{bmatrix} x_1' \\ x_2' \\ x_3' \\ x_4' \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ -8x_1+\sin(t)x_2-3x_3+t^2 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ -8x_1+\sin(t)x_2-3x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ t^2 \end{bmatrix}$$

Now the first vector can be written as a matrix multiplication,

$$\vec{x}' = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -8 & \sin(t) & -3 & 0 \end{bmatrix} \vec{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ t^2 \end{bmatrix}$$

Where,

$$\vec{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

Note that for large systems such as this we will go one step further,

$$\vec{x}' = A\vec{x} + \vec{g}(t)$$

Finally, $\vec{x}' = A\vec{x} + \vec{g}(t)$ is a homogeneous system, while, $\vec{g}(t) = \vec{0}$ is non homogeneous if $\vec{g}(t) \neq \vec{0}$. (See my paper titled "Basics," p2, in folder, Math, Papers, DE).

(DE_276; Second Order Differential Equations; 5520Notes DE LA; all in folder: BOOKS, PDF, MATH, DE and Lin Alg).

ADAMS – BASHFORTH / ADAMS – MOULTON

Where RK4 is a single step method, basing each prediction on only one previously predicted point, (ABAM) is a multi step method. As such, it cannot calculate the first few input y values. Therefore, it is necessary to turn to a one step method such as Euler to find these values. Then one begins using the multi step after a sufficient number of starting values are found. ABAM is also a predictor-corrector method. This uses a formula to first predict an approximation $\hat{y}_{n+1}$, which is then used indirectly in the correcting formula to find y_{n+1}.

ABAM can be used to approximate the value of $\phi(x_{n+1})$ from the solution ϕ of the IVP

$$y' = f(x,y) \qquad (06)$$
$$y(x_0) = y_0$$

at $x_{n+1} = x_0 + (n + 1)h$, provided we have previously found approximations y_n, y_{n-1}, y_{n-2}, y_{n-3}, corresponding to the four previous points x_n, x_{n-1}, x_{n-2}, x_{n-3}.

The method follows: we use (06) to determine y' at each of x_n, x_{n-1}, x_{n-2}, and x_{n-3}. In particular we set $y'_n = f(x_n,y_n)$, $y'_{n-1} = f(x_{n-1},y_{n-1})$, $y'_{n-2} = f(x_{n-2},y_{n-2})$, and $y'_{n-3} = f(x_{n-3},y_{n-3})$.

Using these initial values, we find an initial approximation $\hat{y}_{n+1}$ for $\phi(x_{n+1})$ with the predicting formula:

$$\hat{y}_{n+1} = y_n + \frac{h}{24}\left(55y'_n - 59y'_{n-1} + 37y'_{n-2} - 9y'_{n-3}\right) \qquad (07)$$

Inserting numbers into this, we find the number

$$\hat{y}'_{n+1} = f(x_{n+1}, \hat{y}_{n+1}). \qquad (08)$$

This is used to find y_{n+1} in this correcting formula:

$$y_{n+1} = y_n + \frac{h}{24}\left(9\hat{y}'_{n+1} + 19y'_n - 5y'_{n-1} + y'_{n-2}\right) \qquad (09)$$

Once the values y_0, y_1, y_2, y_3 have been determined, we can start using ABAM with n = 3 to determine y_4. Then with y_4 use this to find y_5, …

Because of the accuracy of RK4, this is the preferred starting method for values y_0 to y_3. (Ross, 463).

EXAMPLE 11

Use RK4 to approximate the solutions for y′ = 2x + y, y(0) = 1, for 0.2, 0.4, and 0.6. Then use ABAM for 0.8 to 2.0. Also finish RK4 for 0.8 to 2.0 for comparison purposes. Finally make a table with actual solutions, RK4, ABAM, and actual values, and calculate % errors. The actual solution is $y(x) = -2x + 3e^x - 2$.

Begin with running RK4 for four iterations for y′ = 2x + y, for h = 0 to 0.6.

$x_0 = 0$, $y_0 = 1.00000000000$,
$x_1 = 0.2$, $y_1 = 1.26420000000$,
$x_2 = 0.4$, $y_2 = 1.67545388000$,
$x_3 = 0.6$, $y_3 = 2.26631936903$,

The first three RK4 are used directly in ABAM. These we will calculate in y′, using the RK4 ‘x′ and ‘y′ values. (This is not a derivative, merely a marker).

$y'_0 = f(x_0y_0) = f(0.0, 1.00000000000) = 1.00000000000$
$y'_1 = f(x_1y_1) = f(0.2, 1.26420000000) = 1.66420000000$
$y'_2 = f(x_2y_2) = f(0.4, 1.67545388000) = 2.47545388000$
$y'_3 = f(x_3y_3) = f(0.6, 2.26631936903) = 3.46631936903$

Starting with $x_0 = 0.6$, we start ABAM intermediate calculations (07). These should be placed into a table. y'_0, y'_1, y'_2, y'_3 were calculated immediately above.

Now, using (07) the predicting formula with n = 3, h = 0.2, we find

$$\begin{aligned}\hat{y}_4 &= y_3 + \frac{0.2}{24}\left(55y'_3 - 59y'_2 + 37y'_1 - 9y'_0\right)\\ &= 2.2663194 + \frac{(55)(4.6765836) - (59)(3.4663193) + (37)(2.4754538) - 9.0000000}{120.0}\\ &= 3.0760793\end{aligned}$$

Having determined $\hat{y}_4$ we determine $\hat{y}'_4$ by substituting (0.80, 3.0760793) into the given

$$\begin{aligned}y' &= 2x + y\\ &= 4.6760793\end{aligned}$$

Use this new value of $\hat{y}'_4$ in the correcting formula (08) to find y_4. Again, n = 3, h = 0.2. Pay attention to the difference between $\hat{y}'_{n+1}$ and the other y'_n, no hats.

$$\begin{aligned}y_4 &= y_n + \frac{0.2}{24}\left(9\hat{y}'_{n+1} + 19y'_n - 5y'_{n-1} + y'_{n-2}\right)\\ &= 2.2663194 + \frac{(9)(4.6760793) + (19)(3.46631936903) - (5)(2.47545388000) + 1.66420000000}{120.0}\\ &= 3.0765836\end{aligned}$$

Now, n = 4, using the new value for y_4, first find y'_4

$y'_4 = f(0.80, 3.0765836) = 4.6765836$

Next use the predicting formula (07) with n = 4, and h = 0.20

$$\hat{y}_5 = 3.0765836 + \frac{(55)(4.6765836) - (59)(3.46631936903) + (37)(2.47545388000) - (9)(1.66420000000)}{120.0}$$

$= 4.1541941$

Next with n = 4, h = 0.20 and $\hat{y}_5$, find $\hat{y}'_5$

$\hat{y}'_5 = f(x_5, \hat{y}_5) = f(1.0, 4.1541941) = 6.1541941$

Now use these in the correcting formula

$$y_5 = y_4 + \frac{h}{24}\left(9\hat{y}'_5 + 19y'_4 - 5y'_3 + y'_2\right)$$

$$= 3.0765836 + \frac{1}{120}[(9)(6.1541941) + (19)(4.6765836) - (5)(3.46631936903)]$$

$$+ 2.47545388000 = 4.1548061.$$

(Ross, 465 #8.13).

If you must do this by hand, it is best to put intermediate results into a table. Below is an example. Notice that the first three rows represent the initial RK4 values.

y′ = 2x + y

N	Xn	Yn	Yn'	Xn+1	Yhat(n+1)	Y'hat(n+1)	Yn+1
0	0.00	1.00000	1.00000				
1	0.20	1.26420	1.66420				
2	0.40	1.67545	2.47545				
3	0.60	2.26632	3.46632	0.80	3.07608	4.67608	3.07658
4	0.80	3.07658	4.67658	1.00	4.15419	6.15419	4.15481
5	1.00	4.15481	6.15481	1.20	5.55956	7.95956	5.56031
6	1.20	5.56031	7.96031	1.40	7.36465	10.16465	7.36557
7	1.40	7.36557	10.16557	1.60	9.65795	12.85795	12.54893
8	1.60	9.65907	12.85907	1.80	12.54756	16.14756	12.54893
9	1.80	12.54893	16.14893	2.00	16.16550	20.16550	16.16717
10	2.00	16.16717					

Error values for the exact solution:

$f(x) = -2x + 3e^x - 2$

Xn	Exact	ABAM	ABAM Err	RK	RK Err
0.20	1.264208	1.264200	0.000008	1.264200	0.000008
0.40	1.675474	1.675454	0.000020	1.675454	0.000020
0.60	2.266356	2.266319	0.000037	2.266319	0.000037
0.80	3.076623	3.076584	0.000039	3.076562	0.000060
1.00	4.154845	4.154806	0.000039	4.154753	0.000092
1.20	5.560351	5.560312	0.000038	5.560216	0.000135
1.40	7.365600	7.365565	0.000035	7.365408	0.000192
1.60	9.659097	9.659070	0.000028	9.658829	0.000268
1.80	12.548942	12.548927	0.000016	12.548574	0.000369
2.00	16.167168	16.167171	0.000003	16.166668	0.000501

The RK method requires four separate evaluations per step, while ABAM calculates only twice, once for y'_n, and again to find $\hat{y}'_{n+1}$. One advantage of RK is that the iteration size may be changed any time. This must not be done with ABAM. This is actually done in RKFehlberg, which includes a test to compare the approximation with the error value. Fehlberg will then adjust the step size if necessary. (Ross, 468).

BIBLIOGRAPHY

5520Notes. https://www.math.cuhk.edu.hk/course_builder/1415/mmat5520/5520notes.pdf

Dawkins, Paul. "Paul's Online Math Notes." http: //tutorial.math.lamar.edu/. Downloaded: 18 Feb 2017. This PDF is "DE Complete."

DE_276. Paul Dawkins. PDF. http: //tutorial.math.lamar.edu/.

RK0. Erik Cheever.   https://lpsa.swarthmore.edu/NumInt/NumIntAll.html

RK3.
https://math.okstate.edu/people/yqwang/teaching/math4513_fall11/Notes/rungekutta.pdf

RKF2. (onlinehw.math.ksu.edu/math340book/chap1/xc1.php)

Ross, Shepley L. Introduction To Ordinary Differential Equations. 4th ed. New York: John Wiley & Sons, 1989.

Second Order Differential Equations. PDF. (mathisfun.com).