2.8 LAB 3: RK2 methods

In this lab, you will extend the code for Euler's method from Lab 2 to implement higher-order methods to solve IVPs of the form

$$y(t_0) = y_0, \quad y'(t) = f(t, y) \text{ for } t > t_0.$$

In particular, you will write programs to implement certain RK2 and RK3 methods.

2.8.1 RK2

A generic one-step method is written as

$$y_{i+1} = y_i + h\Phi(t_i, y_i; h) \quad \text{for } i = 1, 2, \dots, n.$$

To get a Runge-Kutta 2 ("**RK2**") method, set

$$\begin{aligned} k_1 &= f(t_i, y_i), && (2.8.1a)\\ k_2 &= f(t_i + \alpha h, y_i + \beta h k_1), && (2.8.1b)\\ \Phi(t_i, y_i; h) &= a k_1 + b k_2. && (2.8.1c) \end{aligned}$$

In Section 2.4, we saw that if we pick any $b \neq 0$, and let

$$a = 1 - b, \qquad \alpha = \frac{1}{2b}, \qquad \beta = \alpha, \qquad (2.8.2)$$

then we get a second-order method: $|\mathcal{E}_n| \leqslant Kh^2$.

For example, if we choose $b = 1$, we get the so-called *Modified* or *mid-point* Euler Method from Section 2.4. However, *any* value of b, other than $b = 0$ should give a second-order method.

Download the MATLAB script `Euler_Solution.m` and run it. Make sure you understand how it works.

Next, adapt this to implement an RK2 method as follows.

1. Take b in (2.8.1c) to be the last digit of your ID number, unless that is 0, in which case take $b = -1$. (For example, if your ID number is 01234567, take $b = 7$. If your ID number is 76543210, take $b = -1$).

   Compute the values of a, α and β according to (2.8.2).

2. Choose an initial value problem to solve, and for which you know the exact solution. To avoid having a problem that is too simple,

   - your solution should involve trigonometric, logarithmic or nth-root functions.
   - f should depend explicitly on both t and y.

   (Hint: decide on the solution first, and then differentiate that to get f). You also need to choose an initial time, t_0, and a final time for the simulation, t_n.

3. The MATLAB program should approximate the solution to this IVP using your RK2 method for $n = 2$, $n = 4$, $n = 8$, ..., $n = 512$ (at least). For each n it should output the estimate for $y(t_n)$ and the error $|\mathcal{E}_n| = |y(t_n) - y_n|$.

4. The program should produce a figure displaying a log-log plot of these errors against the corresponding values of n, as well as n^{-2} against n. If your method is second-order, then these two lines should be parallel.

2.8.2 What to upload

In `Assignments and Labs` section of the MA385 Blackboard module, click on "Lab 3". Upload your RK2 solver.

Add appropriate comments to the top of your file(s) indicating who wrote it, when they wrote it, what it does, and how it does it (*"Who/When/What/How?")*. Include your ID number, and give the program a sensible name, which includes something distinctive like your name or ID number.

Make sure your programmes run as-is before uploading. If you don't, you might have given it an invalid name, such as one containing spaces or mathematical symbols. The deadline for uploading your code is **Friday, 23 November** (however, you should be able to complete this assignment in this week's labs).