C++: Practical session 3

1 Description

Write a C++ program to solve the ODE

$$y = y(t),\ \frac{\mathrm{d}y}{\mathrm{d}t} \equiv f(y,t) = \sqrt{y},\ y(0) = y_0. \tag{1.1}$$

using the Euler update formula:

$$y^{n+1} = y^n + (\Delta t) f(y^n, t) \tag{1.2}$$

where $y^n = y(n\Delta t)$ is a discrete approximation to $y(t)$.

The user should be able to enter T, the maximum t value for which to compute the solution, as well as the initial value y^0 and the time-step Δt. Suggested values are: $y^0 = 1$, $T = 10$, $\Delta t = 0.001$.

The function $y^n(t)$ should be output to a file so that you can plot it in `gnuplot`.

Consider carefully what kind of loop you should use. What happens if $T/\Delta t$ is not an integer? What happens if this is an integer, n, but $n\Delta t \neq T$, as is entirely possible with floating-point arithmetic?

2 Further work:

a) Determine the exact solution (careful with integration constants), and compare to the numerical solution.

b) Experiment with varying the time-step and see how it affects the solution.

c) Implement the 2nd-order Runge-Kutta scheme:

$$\begin{aligned} k_1 &= f(y^n, t) \\ k_2 &= f(y^n + k_1\Delta t, t + \Delta t) \\ y^{n+1} &= y^n + \tfrac{1}{2}\Delta t(k_1 + k_2) \end{aligned}$$

and compare the results to those of Euler.

When implementing this, you might wish to specify `f(double x)` as a separate function.

d) Calculate the difference between the exact and numerical solutions at $t = T$ and output this.

e) Plot the variation of this error with Δt.

f) Try solving a different ODE.