Civ. Eng. 2 Mathematics: Trapezium and Runge-Kutta methods.

This sheet can be found on the Web: http://www.ma.ic.ac.uk/∼ajm8/Civ2

Euler's method is simple to use, but is not particularly accurate. Thus to keep our error satisfactorily low we may have to choose a very small value of h. Let's see if we can find a more accurate method which involves only a little more calculation.

An obvious generalisation of Euler's method is

$$y_{n+1} = y_n + h\big[c_1 f(x_n,\, y_n) + c_2 f(x_{n+1},\, y_{n+1})\big] \ ,$$

where c_1 and c_2 are constants which we can choose. In **Euler's (forward) method** $c_1 = 1$ and $c_2 = 0$. In the **backwards Euler method** $c_1 = 0$ and $c_2 = 1$.

We define the **local truncation error**, E_n, at the point $x = x_n$ to be **the amount by which the exact solution fails to satisfy our approximate equation**, so that

$$\begin{aligned} E_n &\equiv y(x_{n+1}) - y(x_n) - h\big[c_1 f(x_n,\, y(x_n)) + c_2 f(x_{n+1},\, y(x_{n+1})\big] \\ &= y(x_{n+1}) - y(x_n) - h\big[c_1 y'(x_n) + c_2 y'(x_{n+1})\big] \ , \end{aligned}$$

since $y' = f(x,\, y)$. We now expand $y(x_{n+1})$ and $y'(x_{n+1})$ as Taylor series, recalling that $x_{n+1} = x_n + h$, so that

$$\begin{aligned} E_n &= y(x_n) + hy'(x_n) + \tfrac{1}{2}h^2 y''(x_n) + \tfrac{1}{6}h^3 y'''(x_n) + \ldots \\ &\quad -y(x_n) - hc_1 y'(x_n) \\ &\qquad - hc_2\left[y'(x_n) + hy''(x_n) + \tfrac{1}{2}h^2 y'''(x_n) + \ldots\right] \\ &= [1 - c_1 - c_2]hy'(x_n) + \left[\tfrac{1}{2} - c_2\right]h^2 y''(x_n) + \left[\tfrac{1}{6} - \tfrac{1}{2}c_2\right]h^3 y'''(x_n) + \ldots \end{aligned}$$

We see therefore that for $c_1 = 1$, $c_2 = 0$ (Euler's method) $E_n = \frac{1}{2}h^2 y''(x_n) + O(h^3)$. Euler's method is **first order.** However, if $c_2 = \frac{1}{2}$ and $c_1 = \frac{1}{2}$ then $E_n = -\frac{1}{12}h^3 y'''(x_n) + O(h^4)$. This more accurate, **second order** method is known as **The Trapezium Method:**

$$y_{n+1} - y_n = \tfrac{1}{2}h\big[f(x_n,\, y_n) + f(x_{n+1},\, y_{n+1})\big] \ . \tag{1}$$

Let's try this on the same numerical example we used before:

$$y' = 2xy \qquad \text{with} \quad y(0) = 1 \qquad \text{using} \quad h = \tfrac{1}{4} \ . \tag{2}$$

We have $x_0 = 0$, $y_0 = 1$, $x_1 = \frac{1}{4}$ and $f(x,\, y) = 2xy$. Using (1) with $n = 0$,

$$y_1 - y_0 = \tfrac{1}{8}\big[2x_0 y_0 + 2x_1 y_1\big] \qquad \text{or} \quad y_1 - 1 = \tfrac{1}{8}[0 + \tfrac{1}{2}y_1] \ ,$$

giving $y_1 = \frac{16}{15}$. Now put $n = 1$ in (1), to obtain

$$y_2 - \tfrac{16}{15} = \tfrac{1}{8}\left[2\left(\tfrac{1}{4}\right)\left(\tfrac{16}{15}\right) + 2\left(\tfrac{1}{2}\right)y_2\right] \ ,$$

giving $y_2 = \frac{136}{105} \simeq 1.295$. Now the exact solution to (2) is $y = e^{x^2}$, so that $y(x_2) = e^{0.25} \simeq 1.284$. Euler's method gives the result $y_2 = \frac{9}{8} \simeq 1.125$ which is clearly less accurate.

Unfortunately, the Trapezium method suffers from a severe disadvantage, namely that it is an **implicit** method. It cannot be written in the form y_{n+1} =(something calculable). You may have noticed that we had to solve an equation to find the value of y_{n+1} given y_n. This wasn't so bad for a **linear** problem, but consider, say,

$$y' = y^{10} \qquad \text{with} \quad y(0) = 1 \qquad \text{using} \quad h = 1 \ .$$

Applying the Trapezium method to this problem gives for the first step,

$$y_1 = y_0 + \tfrac{1}{2}\left[y_0^{10} + y_1^{10}\right] \qquad \text{or} \quad y_1^{10} - 2y_1 + 3 = 0 \ .$$

The value of y_1 is given **implicitly** by this equation. We still have work to do to find it, and there may be more than one solution or, as in this case, no solution! Despite being **second order**, the method is not so straightforward to implement.

Second Order Runge-Kutta methods:

There is a large family of methods which, while having an $O(h^3)$ local error, are nevertheless **explicit**. The basic idea is first to use Euler's method (or similar) to provide a first estimate of the solution at some other point, and then to use this estimate to obtain an improved version. The two most popular methods are:

$$\left.\begin{aligned} y^*_{n+1} &= y_n + hf(x_n,\, y_n) \\ y_{n+1} &= y_n + \tfrac{1}{2}h\left[f(x_n,\, y_n) + f(x_{n+1},\, y^*_{n+1})\right] \end{aligned}\right\} \tag{3}$$

and

$$\left.\begin{aligned} y^*_{n+1} &= y_n + \tfrac{1}{2}hf(x_n,\, y_n) \\ y_{n+1} &= y_n + hf(x_n + \tfrac{1}{2}h,\, y^*_{n+1}) \end{aligned}\right\} \tag{4}$$

In these methods y^*_n is not part of the solution and is merely calculated on the way. In (3), y^*_{n+1} is an approximation to y_{n+1} obtained by Euler's method, which is then used in an approximate Trapezium method. In (4), y^*_{n+1} approximates $y(x_n + \frac{1}{2}h)$. Convince yourself that these two methods are **explicit** i.e. y_{n+1} follows by direct calculation from y_n.

4th Order Runge-Kutta methods

If we use 4 evaluations of the function f we can obtain a 4th order explicit method:

$$\left.\begin{aligned} y_{n+1} &= y_n + \tfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \\ &\text{where} \quad k_1 = f(x_n,\ y_n), \qquad k_2 = f\left(x_n + \tfrac{h}{2},\ y_n + \tfrac{h}{2}k_1\right), \\ k_3 &= f\left(x_n + \tfrac{h}{2},\ y_n + \tfrac{h}{2}k_2\right), \qquad k_4 = f(x_n + h,\ y_n + hk_3). \end{aligned}\right\} \tag{5}$$

This is the most famous of all ODE methods and was derived by Kutta in 1905. It was used in an on-board computer to integrate the equations of motion when landing men on the moon! This is many engineers' favourite numerical method and can easily be extended to solving systems of equations.