

Deriving the Runge-Kutta Method

Deriving the midpoint method

The Taylor method is the gold standard for generating better numerical solutions to first order differential equations. A serious weakness in the Taylor method, however, is the need to compute a large number of partial derivatives and do other symbolic manipulation tasks.

For example, the second order Taylor method for the equation

$$y'(t) = f(t, y(t))$$

is

$$y_{i+1} = y_i + h f(t_i, y_i) + \frac{h^2}{2} \left(\frac{\partial f}{\partial t}(t_i, y_i) + f(t_i, y_i) \frac{\partial f}{\partial y}(t_i, y_i) \right)$$

Higher order formulas get even uglier.

The Midpoint method arises from an attempt to replace the second order Taylor method with a simpler "Euler-like" formula

$$y_{i+1} = y_i + h f(t_i + \alpha, y_i + \beta)$$

We can solve for the best values for α and β by applying a first order Taylor expansion to the term $f(t_i + \alpha, y_i + \beta)$:

$$y_{i+1} = y_i + h \left(f(t_i, y_i) + \frac{\partial f}{\partial t}(t_i, y_i) \alpha + \frac{\partial f}{\partial y}(t_i, y_i) \beta + \frac{\partial^2 f}{\partial t \partial y}(t_i, y_i) \alpha \beta \right)$$

The choices of α and β that make this look as close as possible to the second order Taylor formula above are

$$\alpha = \frac{h}{2}$$

$$\beta = \frac{h}{2} f(t_i, y_i)$$

leading to the so-called midpoint rule:

$$y_{i+1} = y_i + h f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} f(t_i, y_i)\right)$$

This formula has a simple interpretation. Essentially what we are doing here is driving an Euler estimate half way across the interval $[t_i, t_{i+1}]$ and computing the slope

$$f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} f(t_i, y_i)\right)$$

at that midpoint. We then rewind back to the point (t_i, y_i) and drive an Euler estimate all the way across the interval to t_{i+1} using this new midpoint slope in place of the old Euler slope.

The Runge-Kutta Method

The textbook points out that it is possible to derive similar methods by starting with more complex Euler-like formulas with more free parameters and then trying to match those Euler-like methods to higher order Taylor formulas. The Runge-Kutta method is essentially an attempt to match a more complex Euler-like formula to a fourth order Taylor method.

The problem with this is that the Euler-like formula needed to match all the complexity of the fourth order Taylor method formula is quite complex. The textbook states in exercise 31 at the end of section 5.4 that the formula required is

$$y_{i+1} = y_i + \frac{h}{6} f(t_i, y_i) + \frac{h}{3} f\left(t_i + \alpha_1 h, y_i + \delta_1 h f(t_i, y_i)\right) + \frac{h}{6} f\left(t_i + \alpha_2 h, y_i + \delta_2 h f\left(t_i + \gamma_2 h, y_i + \gamma_3 h f(t_i, y_i)\right)\right) + \frac{h}{6} f\left(t_i + \alpha_3 h, y_i + \delta_3 h f\left(t_i + \gamma_4 h, y_i + \gamma_5 h f\left(t_i + \gamma_6 h, y_i + \gamma_7 h f(t_i, y_i)\right)\right)\right)$$

It is very messy to do so, but this form can be expanded out and matched against the Taylor formula of order four. This allows us to solve for all the unknown coefficients.

A somewhat cleaner alternative derivation is based on the following argument. Another way to solve for y_{i+1} is to compute this integral

$$\int_{t_i}^{t_{i+1}} y'(t) dt = y(t_{i+1}) - y(t_i) = y_{i+1} - y_i$$

We can imagine beginning to compute the integral by noting that $y'(t) = f(t, y(t))$

$$\int_{t_i}^{t_{i+1}} y'(t) dt = \int_{t_i}^{t_{i+1}} f(t, y(t)) dt$$

Unfortunately, we can not do the integral on the right hand side exactly, because we don't know what $y(t)$ is. That is, after all, the unknown we are trying to solve for. Even though we can't compute the integral on the right exactly, we can estimate it. For example, applying Simpson's rule to the integral produces the estimate

$$\int_{t_i}^{t_{i+1}} f(t, y(t)) dt \approx \frac{h}{3} \left(f(t_i, y(t_i)) + 4 f\left(\frac{t_i+t_{i+1}}{2}, y\left(\frac{t_i+t_{i+1}}{2}\right)\right) + f(t_{i+1}, y(t_{i+1})) \right)$$

The Runge-Kutta method takes this estimate as a starting point. The thing we need to do to make this estimate work is to find a way to estimate the unknown terms $y((t_i + t_{i+1})/2)$ and $y(t_{i+1})$.

The first step is to rewrite the estimate as

$$\frac{h}{3} \left(f(t_i, y(t_i)) + 2 f\left(\frac{t_i+t_{i+1}}{2}, y\left(\frac{t_i+t_{i+1}}{2}\right)\right) + 2 f\left(\frac{t_i+t_{i+1}}{2}, y\left(\frac{t_i+t_{i+1}}{2}\right)\right) + f(t_{i+1}, y(t_{i+1})) \right)$$

We write the middle term twice because we are going to develop two different estimates for $y((t_i + t_{i+1})/2)$. The thinking is that the mistakes we make in developing those two interior estimates may partly cancel each other out.

Here is how we will develop our estimates.

1. $y(t_i)$ is just y_i .
2. We estimate the first $y((t_i + t_{i+1})/2)$ by driving the original Euler slope $k_1 = f(t_i, y_i)$ half-way across the interval:

$$k_1 = f(t_i, y(t_i))$$

$$y\left(\frac{t_i + t_{i+1}}{2}\right) \approx y_i + h/2 k_1$$

3. As in the midpoint rule, we compute a second slope at that midpoint we just estimated. We then rewind to the start and drive that slope half-way across the interval again.

$$k_2 = f(t_i + h/2, y_i + h/2 k_1)$$

$$y\left(\frac{t_i + t_{i+1}}{2}\right) \approx y_i + h/2 k_2$$

4. We use the second estimated midpoint to compute another slope and

then drive that slope all the way across the interval.

$$k_3 = f(t_i + h/2, y_i + h/2 k_2)$$

$$y(t_{i+1}) = y_i + h k_3$$

$$k_4 = f(t_i + h, y_i + h k_3)$$

Substituting all of these estimates into the Simpson's rule formula above gives

$$y_{i+1} - y_i = \int_{t_i}^{t_{i+1}} f(t, y(t)) dt \approx \frac{h}{3} \left(f(t_i, y(t_i)) + 2 f\left(\frac{t_i+t_{i+1}}{2}, y\left(\frac{t_i+t_{i+1}}{2}\right)\right) + 2 f\left(\frac{t_i+t_{i+1}}{2}, y\left(\frac{t_i+t_{i+1}}{2}\right)\right) + f(t_{i+1}, y(t_{i+1})) \right)$$

or

$$y_{i+1} = y_i + \frac{h}{3}(k_1 + 2 k_2 + 2 k_3 + k_4)$$

Summary of the method

$$k_1 = f(t_i, y_i)$$

$$k_2 = f(t_i + h/2, y_i + h/2 k_1)$$

$$k_3 = f(t_i + h/2, y_i + h/2 k_2)$$

$$k_4 = f(t_i + h, y_i + h k_3)$$

$$y_{i+1} = y_i + \frac{h}{3}(k_1 + 2 k_2 + 2 k_3 + k_4)$$