Runge–Kutta methods

From Wikipedia, the free encyclopedia

In numerical analysis, the **Runge–Kutta methods** are an important family of implicit and explicit iterative methods for the approximation of solutions of ordinary differential equations. These techniques were developed around 1900 by the German mathematicians C. Runge and M.W. Kutta.

See the article on numerical ordinary differential equations for more background and other methods.

Contents

The classical fourth-order Runge–Kutta method

One member of the family of Runge–Kutta methods is so commonly used, that it is often referred to as "RK4" or simply as "*the* Runge–Kutta method".

Let an initial value problem be specified as follows.

$$y' = f(t, y), \quad y(t_0) = y_0$$

Then, the RK4 method for this problem is given by the following equation:

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

where

$$k_1 = f\left(t_n, y_n\right)$$

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

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

$$k_4 = f\left(t_n + h, y_n + hk_3\right)$$

Thus, the next value (y_{n+1}) is determined by the present value (y_n) plus the product of the size of the interval (h) and an estimated slope. The slope is a weighted average of slopes:

- k_1 is the slope at the beginning of the interval;
- k_2 is the slope at the midpoint of the interval, using slope k_1 to determine the value of y at the point $t_n + h/2$ using Euler's method;
- k_3 is again the slope at the midpoint, but now using the slope k_2 to determine the y-value;

- k_4 is the slope at the end of the interval, with its y-value determined using k_3.

When the four slopes are averaged, more weight is given to the slopes at the midpoint:

$$\text{slope} = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}.$$

The RK4 method is a fourth-order method, meaning that the error per step is on the order of h^5, while the total accumulated error has order h^4.

Note that the above formulas are valid for both scalar and vector valued functions (y can be a vector).

Explicit Runge–Kutta methods

The family of explicit Runge–Kutta methods is a generalization of the RK4 method mentioned above. It is given by

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i,$$

where

$$\begin{aligned}
k_1 &= f(t_n, y_n), \\
k_2 &= f(t_n + c_2 h, y_n + a_{21} h k_1), \\
k_3 &= f(t_n + c_3 h, y_n + a_{31} h k_1 + a_{32} h k_2), \\
&\vdots \\
k_s &= f(t_n + c_s h, y_n + a_{s1} h k_1 + a_{s2} h k_2 + \cdots + a_{s,s-1} h k_{s-1}).
\end{aligned}$$

(Note: the above equations have different but equivalent definitions in different texts).

To specify a particular method, one needs to provide the integer s (the number of stages), and the coefficients a_{ij} (for $1 \leq j < i \leq s$), b_i (for $i = 1, 2, ..., s$) and c_i (for $i = 2, 3, ..., s$). These data are usually arranged in a mnemonic device, known as a *Runge–Kutta tableau*:

$$\begin{array}{c|cccc}
0 & & & & \\
c_2 & a_{21} & & & \\
c_3 & a_{31} & a_{32} & & \\
\vdots & \vdots & & \ddots & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} \quad b_s
\end{array}$$

The Runge–Kutta method is consistent if

$$\sum_{j=1}^{i-1} a_{ij} = c_i \text{ for } i = 2, \ldots, s.$$

There are also accompanying requirements if we require the method to have a certain order p, meaning that the truncation error is $O(h^{p+1})$. These can be derived from the definition of the truncation error itself. For example, a 2-stage method has order 2 if $b_1 + b_2 = 1$, $b_2 c_2 = 1/2$, and $b_2 a_{21} = 1/2$.

Examples

The RK4 method falls in this framework. Its tableau is:

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

However, the simplest Runge–Kutta method is the (forward) Euler method, given by the formula $y_{n+1} = y_n + hf(t_n,y_n)$. This is the only consistent explicit Runge–Kutta method with one stage. The corresponding tableau is:

0	
	1

An example of a second-order method with two stages is provided by the midpoint method

$$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}f(t_n, y_n)\right).$$

The corresponding tableau is:

0		
1/2	1/2	
	0	1

Usage

What follows is an example usage of a two-stage explicit Runge Kutta method, viz.,

0		
2/3	2/3	
	1/4	3/4

to solve the initial-value problem

$$y' = (\tan y) + 1, \quad y(1) = 1,\ t \in [1, 1.1]$$

with step size h=0.025.

The tableau above yields the equivalent corresponding equations below defining the method:

$$\begin{aligned} u_1 &= y_n \\ u_2 &= y_n + 2/3hf(t_n, u_1) \\ y_{n+1} &= y_n + h(1/4f(t_n, u_1) + 3/4f(t_n + 2/3h, u_2)) \end{aligned}$$

$t_0 = 1$

$y_0 = 1$

$t_1 = 1.025$

$u_1 = y_0 = 1$ $f(t_0,u_1) = 2.557407725$ $u_2 = y_0 + 2 / 3hf(t_0,u_1) = 1.042623462$

$y_1 = y_0 + h(1 / 4 * f(t_0,u_1) + 3 / 4 * f(t_0 + 2 / 3h,u_2)) = 1.066869388$

$t_2 = 1.05$

$u_1 = y_1 = 1.066869388$ $f(t_1,u_1) = 2.813524695$ $u_2 = y_1 + 2 / 3hf(t_1,u_1) = 1.113761467$

$y_2 = y_1 + h(1 / 4 * f(t_1,u_1) + 3 / 4 * f(t_1 + 2 / 3h,u_2)) = 1.141332181$

$t_3 = 1.075$

$u_1 = y_2 = 1.141332181$ $f(t_2,u_1) = 3.183536647$ $u_2 = y_2 + 2 / 3hf(t_2,u_1) = 1.194391125$

$y_3 = y_2 + h(1 / 4 * f(t_2,u_1) + 3 / 4 * f(t_2 + 2 / 3h,u_2)) = 1.227417567$

$t_4 = 1.1$

$u_1 = y_3 = 1.227417567$ $f(t_3,u_3) = 3.796866512$ $u_2 = y_3 + 2 / 3hf(t_3,u_1) = 1.290698676$

$y_4 = y_3 + h(1 / 4 * f(t_3,u_1) + 3 / 4 * f(t_3 + 2 / 3h,u_2)) = 1.335079087$

The numerical solutions correspond to the underlined values. Note we calculate $f(t_i,u_1)$ to avoid recalculation in the y_is.

References

- George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler. Computer Methods for Mathematical Computations. Englewood Cliffs, NJ: Prentice-Hall, 1977. *(See Chapter 6.)*
- Ernst Hairer, Syvert Paul Nørsett, and Gerhard Wanner. *Solving ordinary differential equations I: Nonstiff problems*, second edition. Berlin: Springer Verlag, 1993. ISBN 3-540-56670-8.
- William H. Press, Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling. *Numerical Recipes in C*. Cambridge, UK: Cambridge University Press, 1988. *(See Sections 16.1 and 16.2.)*
- Runge-Kutta 4th order method Notes PPT Matlab Mathematica Maple Mathcad (http://numericalmethods.eng.usf.edu/topics/runge_kutta_4th_method.html) at *Holistic Numerical Methods Institute*