

10.5: Runge-Kutta Methods

The three methods that we have surveyed thus far (Forward Euler, Backward Euler, and Adams-Moulton) have all involved sampling the derivative function $F(y, t)$ at one of the discrete time steps $\{t_n\}$, and the solutions at those time steps $\{\vec{y}_n\}$. It is natural to ask whether we can improve the accuracy by sampling the derivative function at "intermediate" values of t and \vec{y} . This is the basic idea behind a family of numerical methods known as **Runge-Kutta** methods.

Here is a simple version of the method, known as **second-order Runge-Kutta** (RK2). Our goal is to replace the derivative term with a pair of terms, of the form

$$\vec{y}_{n+1} = \vec{y}_n + Ah\vec{F}_A + Bh\vec{F}_B, \quad (10.5.1)$$

where

$$\vec{F}_A = \vec{F}(\vec{y}_n, t_n) \quad (10.5.2)$$

$$\vec{F}_B = \vec{F}(\vec{y}_n + \beta\vec{F}_A, t_n + \alpha). \quad (10.5.3)$$

The coefficients $\{A, B, \alpha, \beta\}$ are adjustable parameters whose values we'll shortly choose, so as to minimize the local truncation error.

During each time step, we start out knowing the solution \vec{y}_n at time t_n , we first calculate \vec{F}_A (which is the derivative term that goes into the Forward Euler method); then we use that to calculate an "intermediate" derivative term \vec{F}_B . Finally, we use a weighted average of \vec{F}_A and \vec{F}_B as the derivative term for calculating \vec{y}_{n+1} . From this, it is evident that this is an *explicit* method: for each of the sub-equations, the "right-hand sides" contain known quantities.

We now have to determine the appropriate values of the parameters $\{A, B, \alpha, \beta\}$. First, we Taylor expand \vec{y}_{n+1} around t_n , using the chain rule:

$$\vec{y}_{n+1} = \vec{y}_n + h \left. \frac{d\vec{y}}{dt} \right|_{t_n} + \frac{h^2}{2} \left. \frac{d^2\vec{y}}{dt^2} \right|_{t_n} + O(h^3) \quad (10.5.4)$$

$$= \vec{y}_n + h\vec{F}(\vec{y}_n, t_n) + \frac{h^2}{2} \left[\frac{d}{dt} \vec{F}(\vec{y}(t), t) \right]_{t_n} + O(h^3) \quad (10.5.5)$$

$$= \vec{y}_n + h\vec{F}(\vec{y}_n, t_n) + \frac{h^2}{2} \left[\sum_j \frac{\partial \vec{F}}{\partial y_j} \frac{dy_j}{dt} + \frac{\partial \vec{F}}{\partial t} \right]_{t_n} + O(h^3) \quad (10.5.6)$$

$$= \vec{y}_n + h\vec{F}_A + \frac{h^2}{2} \left\{ \sum_j \left[\frac{\partial \vec{F}}{\partial y_j} \right]_{t_n} F_{Aj} + \left[\frac{\partial \vec{F}}{\partial t} \right]_{t_n} \right\} + O(h^3) \quad (10.5.7)$$

In the same way, we Taylor expand the intermediate derivative term F_B , whose formula was given above:

$$F_B = F_A + \beta \sum_j F_{Aj} \left[\frac{\partial F}{\partial y_j} \right]_{t_n} + \alpha \left[\frac{\partial F}{\partial t} \right]_{t_n}. \quad (10.5.8)$$

If we compare these Taylor expansions to the RK2 formula, then it can be seen that the terms can be made to match up to (and including) $O(h^2)$, if the parameters are chosen to obey the equations

$$A + B = 1, \quad \alpha = \beta = \frac{h}{2B}. \quad (10.5.9)$$

One possible set of solutions is $A = B = 1/2$ and $\alpha = \beta = h$. With these conditions met, the RK2 method has local truncation error of $O(h^3)$, one order better than the Forward Euler Method (which is likewise an explicit method), and comparable to the Adams-Moulton Method (which is an implicit method).

The local truncation error can be further reduced by taking more intermediate samples of the derivative function. The most commonly-used Runge-Kutta method is the **fourth-order Runge Kutta method** (RK4), which is given by

$$\vec{y}_{n+1} = \vec{y}_n + \frac{h}{6} (\vec{F}_A + 2\vec{F}_B + 2\vec{F}_C + \vec{F}_D) \quad (10.5.10)$$

$$\vec{F}_A = \vec{F}(\vec{y}_n, t_n), \quad (10.5.11)$$

$$\vec{F}_B = \vec{F}(\vec{y}_n + \frac{h}{2}\vec{F}_A, t_n + \frac{h}{2}), \quad (10.5.12)$$

$$\vec{F}_C = \vec{F}(\vec{y}_n + \frac{h}{2}\vec{F}_B, t_n + \frac{h}{2}), \quad (10.5.13)$$

$$\vec{F}_D = \vec{F}(\vec{y}_n + h\vec{F}_C, t_n + h). \quad (10.5.14)$$

This has local truncation error of $O(h^5)$. It is an explicit method, and therefore has the disadvantage of being unstable if the problem is stiff and h is sufficiently large.

This page titled [10.5: Runge-Kutta Methods](#) is shared under a [CC BY-SA 4.0](#) license and was authored, remixed, and/or curated by [Y. D. Chong](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.