Runge-Kutta Algorithm

The Runge-Kutta algorithm is *the* magic formula behind most of the physics simulations shown on this web site. The Runge-Kutta algorithm lets us solve a differential equation numerically (that is, approximately); it is known to be very accurate and well-behaved for a wide range of problems.

Consider the single variable problem

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

with initial condition $x(0) = x_0$. Suppose that x_n is the value of the variable at time t_n. The Runge-Kutta formula takes x_n and t_n and calculates an approximation for x_{n+1} at a brief time later, t_n+h. It uses a weighted average of approximated values of $f(t, x)$ at several times within the interval (t_n, t_n+h). The formula is given by

$$x_{n+1} = x_n + \tfrac{h}{6}(a + 2b + 2c + d)$$

where

$$a = f(t_n, x_n)$$

$$b = f(t_n + \tfrac{h}{2}, x_n + \tfrac{h}{2} a)$$

$$c = f(t_n + \tfrac{h}{2}, x_n + \tfrac{h}{2} b)$$

$$d = f(t_n + h, x_n + h c)$$

To run the simulation, we start with x_0 and find x_1 using the formula above. Then we plug in x_1 to find x_2 and so on.

Multi-variable Runge-Kutta Algorithm

With multiple variables, the Runge-Kutta algorithm looks similar to the previous equations, except that the variables become vectors.

The Runge-Kutta Algorithm is fairly simple, but to describe it precisely we need to develop some notation. Suppose there are m variables $x_1, x_2, ..., x_m$ each of which vary over time. For example, in the single spring simulation, x_1 is position, x_2 is velocity. Suppose further that there are m differential equations for these m variables

$$x_1' = f_1(x_1, x_2, ..., x_m)$$

$$x_2' = f_2(x_1, x_2, ..., x_m)$$

$$...$$

$$x_m' = f_m(x_1, x_2, ..., x_m)$$

Notice there are no derivatives on the right hand side of any of those equations, and there are only first derivatives on the left hand side. These equations can be summarized in vector form as

$$\bar{\mathbf{x}}' = \bar{\mathbf{f}}(\bar{\mathbf{x}})$$

where $\bar{\mathbf{x}} = (x_1, x_2, ..., x_m)$ and we allow some loose "vector of functions" concept where $\bar{\mathbf{f}} = (f_1, f_2, ..., f_m)$. Next we label our time states $\bar{\mathbf{x}}_n, \bar{\mathbf{x}}_{n+1}$ which are separated by time interval of length h. That is, $\bar{\mathbf{x}}_n$ is the value of the m variables at time t_n. And $x_{1,n}$ is the value of the first variable x_1 at time t_n.

$$\bar{\mathbf{x}}_n = (x_{1,n}, x_{2,n}, ..., x_{m,n})$$

$$\bar{\mathbf{x}}_{n+1} = (x_{1,n+1}, x_{2,n+1}, ..., x_{m,n+1})$$

Suppose we have the state of the simulation at time t_n as $\bar{\mathbf{x}}_n$. To compute the state a short time h later and put the results into $\bar{\mathbf{x}}_{n+1}$, the Runge-Kutta algorithm does the following:

$$\bar{\mathbf{a}}_n = \bar{\mathbf{f}}(\bar{\mathbf{x}}_n)$$

$$\bar{\mathbf{b}}_n = \bar{\mathbf{f}}(\bar{\mathbf{x}}_n + \tfrac{h}{2} \bar{\mathbf{a}}_n)$$

$$\bar{\mathbf{c}}_n = \bar{\mathbf{f}}(\bar{\mathbf{x}}_n + \tfrac{h}{2} \bar{\mathbf{b}}_n)$$

$$\bar{\mathbf{d}}_n = \bar{\mathbf{f}}(\bar{\mathbf{x}}_n + h \bar{\mathbf{c}}_n)$$

$$\bar{\mathbf{x}}_{n+1} = \bar{\mathbf{x}}_n + \tfrac{h}{6}(\bar{\mathbf{a}}_n + 2\bar{\mathbf{b}}_n + 2\bar{\mathbf{c}}_n + \bar{\mathbf{d}}_n)$$

The new vector $\bar{\mathbf{x}}_{n+1}$ gives you the state of the simulation after the small time h has elapsed. To spell out the above in more detail, we can drop the vector notation and write the Runge-Kutta algorithm like this:

$$x_{j,n} = f_j(x_{1,n}, x_{2,n}, \ldots, x_{m,n})$$

$$x_{j,n} = f_j\big((x_{1,n} + \tfrac{h}{2} x_{1,n}), (x_{2,n} + \tfrac{h}{2} x_{2,n}), \ldots, (x_{m,n} + \tfrac{h}{2} x_{m,n})\big)$$

$$x_{j,n} = f_j\big((x_{1,n} + \tfrac{h}{2} x_{1,n}), (x_{2,n} + \tfrac{h}{2} x_{2,n}), \ldots, (x_{m,n} + \tfrac{h}{2} x_{m,n})\big)$$

$$x_{j,n} = f_j\big((x_{1,n} + h\, x_{1,n}), (x_{2,n} + h\, x_{2,n}), \ldots, (x_{m,n} + h\, x_{m,n})\big)$$

$$x_{j,n+1} = x_{j,n} + \tfrac{h}{6}(x_{j,n} + 2x_{j,n} + 2x_{j,n} + x_{j,n})$$

The above equations are applied for each variable $j=(1, ..., m)$ to get the full set of variables in the vector $\bar{\mathbf{x}}_{n+1}$.

Time As A Variable

Most of the simulations shown on this website *do not* have differential equations that depend explicitly on time. That is, you won't see the variable t on the right-hand side of the differential equations. One simulation that *does* depend on time is the Chaotic Driven Pendulum because the driving force (which applies the twist to the pendulum) varies over time according to $\cos(k\, t)$.

When time appears explicitly in the differential equations we can add a time variable t to the state vector $\bar{\mathbf{x}}$. Suppose we assign this role to the variable x_2. This new variable has the extremely simple differential equation

$$x_2' = 1$$

That says that the rate of change of the variable x_2 is a constant. Since we are taking derivatives with respect to time we can also write the above equation as

$$x_2' = \frac{d}{dt} x_2 = 1$$

This integrates very easily to give $x_2 = t$, which is what we wanted: time as a variable. Suppose that in the driven pendulum simulation we set up x_2 in this way. Then the driving force is given by $\cos(k\, x_2)$.

You may ask: *Why have time as a variable? We already know the value of t at each time step!* The Runge-Kutta algorithm works by averaging the predicted rates at various points in the time interval from t to $t+h$. Therefore, when the rates (differential equations) depend explictly on t, we also need to know the value of t at those points *within* the time interval. Putting time in as a variable makes for nicer cleaner computer code.

Time Passed Directly

If you want to, you can avoid keeping time as an additional variable. The following is an equivalent formulation of the Runge-Kutta algorithm where time t is passed in as a variable to each function in $\bar{\mathbf{f}}$.

$$\bar{\mathbf{a}}_n = \bar{\mathbf{f}}(t, \bar{\mathbf{x}}_n)$$

$$\bar{\mathbf{b}}_n = \bar{\mathbf{f}}(t + \tfrac{h}{2}, \bar{\mathbf{x}}_n + \tfrac{h}{2} \bar{\mathbf{a}}_n)$$

$$\bar{\mathbf{c}}_n = \bar{\mathbf{f}}(t + \tfrac{h}{2}, \bar{\mathbf{x}}_n + \tfrac{h}{2} \bar{\mathbf{b}}_n)$$

$$\bar{\mathbf{d}}_n = \bar{\mathbf{f}}(t + h, \bar{\mathbf{x}}_n + h \bar{\mathbf{c}}_n)$$

$$\bar{\mathbf{x}}_{n+1} = \bar{\mathbf{x}}_n + \tfrac{h}{6}(\bar{\mathbf{a}}_n + 2\bar{\mathbf{b}}_n + 2\bar{\mathbf{c}}_n + \bar{\mathbf{d}}_n)$$

This is equivalent to the formulation where time is kept as one of the variables in $\bar{\mathbf{x}}$. Whether you use this formulation or the earlier (cleaner) one is entirely up to you.

The source code for the Runge-Kutta algorithm is available.

This web page was first published April 2001.

© Erik Neumann, 2004-2016
revised Feb 6 2021