Second-Order Runge-Kutta — Theory

January 29, 2025

Velocity from Acceleration — Recap

On January 24, we settled on and used the following update strategy:

$$t_{i+1} = t_i + \Delta t$$

$$x(t_{i+1}) = x(t_i) + \frac{v(t_i)+v(t_{i+1})}{2} \cdot \Delta t$$

$$v(t_{i+1}) = v(t_i) + a\left(t_i + \frac{\Delta t}{2}\right) \cdot \Delta t$$

I warned you that there was going to be a complication.

The Complication — Recap

In the formula

$$v(t_{i+1}) \approx v(t_i) + a\left(t_i + \frac{\Delta t}{2}\right) \cdot \Delta t$$

or with Newton's 2nd Law substituted in,

$$v(t_{i+1}) \approx v(t_i) + \frac{F\left(t_i + \frac{\Delta t}{2}\right)}{m} \cdot \Delta t$$

the complication is that in most physical systems

$$a(t) = \frac{F(t)}{m}$$

is not directly a given function of time. Far more commonly it is indirectly a function of time via the position and velocity:

$$a(t) = \frac{F(x(t), v(t))}{m}$$

Sometimes the force is both directly and indirectly a function of time, in which case we would write:

$$a(t) = \frac{F(t, x(t), v(t))}{m}$$

A Simplification

So that we don't overdo the complexity right off the bat, let's focus on a simpler and quite common situation,

$$a(t) = \frac{F(x(t))}{m}$$

In other words, let's leave out the possibility that *F* depends directly on the time, and also leave out the possibility that *F* depends on the velocity. We are only allowing *F* to depend indirectly on the time via the position.

Leaving out velocity-dependent forces means that systems with friction, drag, or magnetism aren't ones we can yet work on. Leaving out time-dependence means that we can't work on systems that have externally-controlled forces, like the force of a person periodically pushing on the back of a kid on a swing-set.

However, a lot of interesting systems can be idealized as frictionless — for example, the motion of a mass on a spring, or the motion of a planet around the Sun — and many, if not most, interesting systems do not have externally applied and controlled forces.

A Chicken-and-Egg Problem

Here are our three update equations, but rewritten to emphasize that the only way that the acceleration a depends on t is via $x(t)$:

$$t_{i+1} = t_i + \Delta t$$

$$x(t_{i+1}) = x(t_i) + \frac{v(t_i)+v(t_{i+1})}{2} \cdot \Delta t$$

$$v(t_{i+1}) = v(t_i) + a\left(x\left(t_i + \frac{\Delta t}{2}\right)\right) \cdot \Delta t$$

Notice that I am still using the trapezoid approximation to update x and I am still using the midpoint time in $a(x(t))$ to update v, exactly as we did above.

Do you see the chicken-and-egg problem!?

We can't compute the updated $x(t_{i+1})$ without knowing $v(t_{i+1})$

We can't compute the updated $v(t_{i+1})$ without knowing a at the midpoint time, which in turn requires knowing x at the midpoint time, $x\left(t_i + \frac{\Delta t}{2}\right)$.

A Worse Chicken-and-Egg Problem

We might try a trapezoid approximation for the calculation of $v(t_{i+1})$, but that makes the chicken-and-egg problem if anything a little more poignant:

$$t_{i+1} = t_i + \Delta t$$

$$x(t_{i+1}) = x(t_i) + \frac{v(t_i)+v(t_{i+1})}{2} \cdot \Delta t$$

$$v(t_{i+1}) = v(t_i) + \frac{a(x(t_i))+a(x(t_{i+1}))}{2} \cdot \Delta t$$

Good luck telling Mathematica how to do that pile of circular reasoning.

A Way Out

Imagine that instead of using midpoint or trapezoid, we begin by making the simple left-hand approximation for $x(t_{i+1})$, and we'll even give it its own symbol to emphasize that it is the dirt-simple left-hand approximation:

$$x^*(t_{i+1}) \approx x(t_i) + v(t_i)\,\Delta t$$

Definitely we can do that without encountering any circular reasoning because the right-hand side only contains $x(t_i)$ and $v(t_i)$. Then we make a version of the trapezoid approximation to $a_{i\to i+1,\text{avg}}$, by trying

$$a_{t_i \to t_{i+1},\text{avg}} \approx \frac{a(x(t_i))+a(x^*(t_{i+1}))}{2}$$

Then we put that version of the trapezoid approximation into the update equation for $v(t_{i+1})$:

$$v(t_{i+1}) = v(t_i) + \frac{a(x(t_i))+a(x^*(t_{i+1}))}{2} \cdot \Delta t$$

Still no circular reasoning! Finally, we use the trapezoid approximation again to get a more sophisticated approximation to $x(t_{i+1})$:

$$x(t_{i+1}) = x(t_i) + \frac{v(t_i)+v(t_{i+1})}{2} \cdot \Delta t$$

You might have a feeling that this is *ad hoc,* but at least the chicken-and-egg problem has been evaded.

This procedure and generalizations of it have stood the test of time. They were studied by Carl Runge in 1895, Heun in 1900, and Martin Wilhelm Kutta in 1901 and this version and other Runge-Kutta procedures that we will soon get to are still widely used.

Summary — Second-Order Runge-Kutta

We were motivated to try this update procedure when F or a depended on time indirectly via the position x:

$$t_{i+1} = t_i + \Delta t$$

$$x(t_{i+1}) = x(t_i) + \frac{v(t_i)+v(t_{i+1})}{2} \cdot \Delta t$$

$$v(t_{i+1}) = v(t_i) + \frac{a(x(t_i))+a(x(t_{i+1}))}{2} \cdot \Delta t$$

We realized that this system suffered a chicken-and-egg problem.

We decided to try this update procedure instead:

$$t_{i+1} = t_i + \Delta t$$

$$x^*(t_{i+1}) = x(t_i) + v(t_i)\,\Delta t$$

$$v(t_{i+1}) = v(t_i) + (a(x(t_i)) + a(x^*(t_{i+1}))) \cdot \frac{\Delta t}{2}$$

$$x(t_{i+1}) = x(t_i) + (v(t_i) + v(t_{i+1})) \cdot \frac{\Delta t}{2}$$

This update procedure is an example of Second-Order Runge-Kutta. There is a one-parameter family of such procedures. This particular member of the family is what we are going to use shortly to get the motion of a mass on a spring.

A Note on Sources

At some point in the future, you may want a reference that gets to a sophisticated punch-line much more quickly. I am deliberately building up a sophisticated procedure one modest step at a time. More advanced references cut straight to the chase. When preparing these notes, I referred to Section 3.1 of Gregory Fasshauer's notes for IIT Math 472, online at https://www.math.iit.edu/~fass/472Notes.pdf. The ultimate reference, by a person who advanced the theory in the 1960s, seems to be J.C. Butcher, *Numerical Methods for Ordinary Differential Equations,* Wiley, 2003. A more introductory and physics-focused book is Alejandro L. Garcia, *Numerical Methods for Physics,* Prentice-Hall, 1994. The code listings in it are in MATLAB. Another introductory and physics-focused book is Nicholas J. Giordano, *Computational Physics,* Prentice-Hall, 1997. The code listings in it are in BASIC. As computational methods have gotten more popular, there have followed an endless number of other textbooks. See also the fine discussion below, which I am at a bit of a loss as to the origin of my screenshot (could it be https://math.stackexchange.com/questions/528856/):

$$x(t+\Delta t) = x(t) + (\Delta t)x'(t) + \frac{(\Delta t)^2}{2!}x''(t) + \text{higher order terms.} \tag{1}$$

$$f(t+\Delta t, x+\Delta x) = f(t,x) + (\Delta t)f_t(t,x) + (\Delta x)f_x(t,x) + \text{higher order terms.} \tag{2}$$

We are interested in the following ODE:

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

The value $x(t)$ is known and $x(t+h)$ is desired. We can solve this ODE by integrating:

$$x(t+h) = x(t) + \int_t^{t+h} f(\tau, x(\tau))\mathrm{d}\tau.$$

Unfortunately, actually doing that integration exactly is often quite hard (or impossible), so we approximate it using quadrature:

$$x(t+h) \approx x(t) + h\sum_{i=1}^{N} \omega_i f(t+\nu_i h, x(t+\nu_i h)).$$

The accuracy of the quadrature depends on the number of terms in the summation (the order of the Runge–Kutta method), the weights, ω_i, and the position of the nodes, ν_i.

Even this quadrature can be quite difficult to calculate, since on the right-hand side we need $x(t+\nu_i h)$, which we don't know. We get around this problem in the following manner:

Let $\nu_1 = 0$, which makes the first term in the quadrature $K_1 := hf(t, x(t))$. This we do know and we can also use it to approximate $x(t+\nu_2 h)$ by writing the second term in the quadrature as $K_2 := hf(t+\alpha h, x(t)+\beta K_1)$.

With this, the quadrature formula is:

$$x(t+h) = x(t) + \omega_1 K_1 + \omega_2 K_2. \tag{3}$$

Some notes:

1. If we wanted to find a third-order method, we would introduce $K_3 = hf(t+\tilde{\alpha}h, x+\tilde{\beta}K_1+\tilde{\gamma}K_2)$. If we wanted a fourth-order method, we would introduce K_4 in a similar way.
2. This is an explicit method, i.e. we have chosen K_2 to depend on K_1 but K_1 does not depend on K_2. Similarly, K_3 (if we introduce it) depends on K_1 and K_2 but they do not depend on K_3. We could allow the dependence to run both ways but then the method is implicit and becomes much harder to solve.
3. We still need to choose α, β and the ω_i. We will do that now using the Taylor series expansions I introduced at the very beginning.

In equation (3), we substitute in the Taylor series expansion (1) on the left-hand side:

$$x(t) + hx'(t) + \frac{h^2}{2!}x''(t) + \mathcal{O}\left(h^3\right) = x(t) + \omega_1 K_1 + \omega_2 K_2.$$

Since $x' = f$ and thus $x'' = f_t + ff_x$ (suppressing arguments for ease of notation), we have:

$$hf + \frac{h^2}{2}(f_t + ff_x) + \mathcal{O}\left(h^3\right) = \omega_1 K_1 + \omega_2 K_2.$$

Now substitute for K_1 and K_2:

$$hf + \frac{h^2}{2}(f_t + f f_x) + \mathcal{O}\left(h^3\right) = \omega_1 h f + \omega_2 h f(t + \alpha h, x + \beta K_1).$$

Taylor-expand on the right-hand side using (2):

$$hf + \frac{h^2}{2}(f_t + f f_x) + \mathcal{O}\left(h^3\right) = \omega_1 h f + \omega_2(h f + \alpha h^2 f_t + \beta h^2 f f_x) + \mathcal{O}\left(h^3\right).$$

Thus the Runge–Kutta method will agree with the Taylor series approximation to $\mathcal{O}\left(h^3\right)$ if we choose:

$$\omega_1 + \omega_2 = 1,$$

$$\alpha\omega_2 = \frac{1}{2},$$

$$\beta\omega_2 = \frac{1}{2}.$$

The canonical choice for the second-order Runge–Kutta methods is $\alpha = \beta = 1$ and $\omega_1 = \omega_2 = 1/2$.