1. Ordinary Differential Equations

Differential equations have the form:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(x,t) \tag{1}$$

and can come both in systems (i.e. multiple equations that have to remain all true) and can contain higher orders in the derivatives. In the case above, we will want to solve for $x(t)$, and x is referred to as the dependent variable, and t is referred to as the independent variable.

What is the distinction between ordinary and partial differential equations?

Ordinary differential equations have a single independent variable, whereas *partial differential equations* require multiple independent variables.

For example, Poisson's equation:

$$\nabla^2\rho = 4\pi G\rho(x,y,z) \tag{2}$$

involves derivatives in all three coordinates, and they are all required in general. You can reduce Poisson's equation to an ODE in spherical symmetry.

What is the distinction between homogeneous and nonhomogeneous ODEs?

Homogeneous ODEs have no terms without the dependent variable. Nonhomogeneous ODEs have "driving" terms, without the dependent variable.

For example, a nonhomogeneous equation is:

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = f(t) \tag{3}$$

What is the distinction between linear and nonlinear ODEs?

Linear ODEs are *linear* in the dependent variable (but not necessarily in the independent variable).

Thus, the following equation is linear:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = t^3x(t) \tag{4}$$

but this equation is *nonlinear*:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = tx^3(t) \tag{5}$$

While solutions of homogeneous, linear ODEs may be superposed to find new solutions, the same is not true of nonlinear equations. Any two solutions to the top equation x_1 and x_2 can be

linearly combined into a new equation, but that's not going to be necessarily true of (nontrivial) solutions of the bottom equation.

With ODEs, you need to set *boundary conditions*. For example, in the simplest case, you might set a set of conditions at $t = 0$ (i.e. *initial conditions*). For each dependent variable, the number of independent boundary conditions you need to set are equal to the order of the problem.

You can in principle set conditions at multiple t, as long as you have the right number of conditions. In such cases, it is not necessarily the case that your conditions can be met under your equations of course. It definitely complicates the finding of a solution.

ODEs of any order can be reduced to systems of linear equations. Specifically they can be reduced to a form like:

$$\begin{aligned} \frac{\mathrm{d}w_0}{\mathrm{d}t} &= f_0(\vec{w}, t) \\ \frac{\mathrm{d}w_1}{\mathrm{d}t} &= f_1(\vec{w}, t) \\ &\cdots \\ \frac{\mathrm{d}w_{N-1}}{\mathrm{d}t} &= f_{N-1}(\vec{w}, t) \end{aligned} \tag{6}$$

where no terms appear on the right-hand side that are derivatives of $\vec{w}$ with t. In vector form this appears as:

$$\frac{\mathrm{d}\vec{w}}{\mathrm{d}t} = \vec{f}(\vec{w}, t) \tag{7}$$

But do not confuse these vectors with three-dimensional vectors in configuration space. More often (usually) they involve vectors in phase space.

For example, take the second-order equation in three-dimensional space:

$$\frac{\mathrm{d}^2\vec{x}}{\mathrm{d}t^2} = -\vec{x} + \vec{f}(t) \tag{8}$$

which represents a spring under some time-dependent forcing. This may be reduced to one dimension as follows:

$$\begin{aligned} \frac{\mathrm{d}\vec{x}}{\mathrm{d}t} &= \vec{v} \\ \frac{\mathrm{d}\vec{v}}{\mathrm{d}t} &= -\vec{x} + \vec{f}(t) \end{aligned} \tag{9}$$

which represents six first-order equations in the 6-D phase space vector $\vec{w} = (\vec{x}, \vec{v})$.

2. Algorithms for ODEs

For a general nonlinear ODE, given initial conditions, we will need to integrate (usually) forward in time numerically. There are a number of well-developed algorithms for doing this. As in

other applications we have done this semester, the more complex algorithm tend to take the form of higher-order approximations to the solutions of the problem.

The simplest ODE integrator that exists is *Euler's algorithm*. Like all the simplest examples in this course, you should probably never use it, but it is illustrative.

This algorithm takes equal time steps Δt, starting from $\vec{w}(t=0)$, just says:

$$\vec{w}(t+\Delta t) = \Delta t \dot{\vec{w}}(t) = \Delta t \vec{f}(\vec{w}, t) \tag{10}$$

Perhaps not surprisingly given what we have learned about similar methods for forward differencing, the error in this method is $\mathcal{O}(\Delta t^2)$. This means it is accurate to first-order in Δt.

Second-order Runga-Kutta is simple second-order algorithm evaluates the function at $t+\Delta t/2$, and then takes the full step based on the values at the mid-point:

$$\begin{aligned} k_1 &= \Delta t f(\vec{w}_n, t_n) \\ k_2 &= \Delta t f\left(\vec{w}_n + \frac{1}{2}k_1, t_n + \frac{1}{2}\Delta t\right) \\ \vec{w}_{n+1} &= \vec{w}_n + k_2 \end{aligned} \tag{11}$$

We can see why this is second-order fairly easily. In the case that we are integrating just one dependent variable, we are trying to approximate the integral:

$$\Delta w = \int_0^{\Delta t} \mathrm{d}t f(w(t), t) \tag{12}$$

We can Taylor expand $f(w(t), t)$ in t:

$$f(w,t) \approx f\left(\Delta t/2\right) + \left(t - \frac{\Delta t}{2}\right)\left[\frac{\mathrm{d}f}{\mathrm{d}t}\right]_{\Delta t/2} + \frac{1}{2}\left(t - \frac{\Delta t}{2}\right)^2 \left[\frac{\mathrm{d}^2 f}{\mathrm{d}t^2}\right]_{\Delta t/2} + \ldots \tag{13}$$

When we insert this into the integral we find the second term vanishes:

$$\Delta w \approx \Delta t f(w(\Delta t/2), \Delta t/2) + \mathcal{O}(\Delta t^3) \tag{14}$$

This assumes that we have an estimate of $f(\Delta t/2)$. However, since this is multiplied by Δt, to remain second-order we only need this estimate to be good to first-order. Thus, the Euler algorithm can be used to estimate the midpoint and we get the procedure above.

This second-order Runge-Kutta algorithm requires two determinations of f, not one. So it will only be useful when the Δt^2 term is sufficiently small (i.e. the function is sufficiently smooth), so that the decrease in time step is worth it.

Harder to prove, but more useful generally, is the fourth-order Runge-Kutta. This is given by evaluating the derivative at the start, using that to estimate the derivative at the mid-point,

then using the mid-point derivative to estimate the mid-point *again*, and and using those results to estimate a trial end point, before reaching the final answer.

$$\begin{aligned}
\vec{k}_1 &= \Delta t \vec{f}(\vec{w}_n, t_n) \\
\vec{k}_2 &= \Delta t \vec{f}(\vec{w}_n + \frac{1}{2}\vec{k}_1, t_n + \frac{1}{2}\Delta t) \\
\vec{k}_3 &= \Delta t \vec{f}(\vec{w}_n + \frac{1}{2}\vec{k}_2, t_n + \frac{1}{2}\Delta t) \\
\vec{k}_4 &= \Delta t \vec{f}(\vec{w}_n + \vec{k}_3, t_n + \Delta t) \\
\vec{w}_{n+1} &= \vec{w}_n + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4
\end{aligned} \tag{15}$$

This, it turns out, has error terms $\mathcal{O}(\Delta t^5)$.

3. Step sizes

In integrating differential equations, setting the step size is critical. Generally you don't know this in advance. ODE integrators essentially always use an adaptive step size designed to keep the integration accuracy within some tolerance.

Specifically, this is typically done by performing the integration in pairs of steps, and then comparing the answer using two steps to using one step twice as big. This adds at most 50% more function evaluations.

The two answers will be:

$$\begin{aligned}
w(t + 2\Delta t) &= w_1 + (2\Delta t)^5\phi + \mathcal{O}(\Delta t^6) \\
w(t + 2\Delta t) &= w_2 + 2(\Delta t)^5\phi + \mathcal{O}(\Delta t^6)
\end{aligned} \tag{16}$$

And you can adjust Δt based on:

$$\delta = w_2 - w_1 \approx -30(\Delta t)^5\phi \tag{17}$$

Note that you usually should consider both the absolute and relative errors.

Note that these can be combined to:

$$w(t + 2\Delta t) = w_2 + \frac{\delta}{15} \tag{18}$$

for a fifth order method.

4. Symplectic Integrators

Sometimes what we want is not just high order accuracy, but we want certain facts about the original equations to remain true, perhaps even at the expense of accuracy. For example, many

sets of equations are required to obey conservation laws, such as momentum, energy, etc. It is also sometimes nice to guarantee that the equations you are using are time-reversible, just like the actual equations in physics can be.

These classes of methods are sometimes known as "symplectic," because of its relationship to symplectic transforms in mathematics, which we will not get into here. However, it is this property that leads these finite difference expressions to have the useful properties that they do have.

The simplest version of a symplectic integrator is the leapfrog algorithm. For example, for a one-dimensional equation of motion:

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= F(x) \end{aligned} \tag{19}$$

we can write:

$$\begin{aligned} x_1 &= x_0 + \Delta v_{1/2} \\ v_{3/2} &= v_{1/2} + \Delta F(x_1) \end{aligned} \tag{20}$$

This will be accurate with remainders of order Δ^3 at each time step. Note that eventually you want v_1, so:

$$v_1 = v_{1/2} + \Delta F(x_1)/2 \tag{21}$$

What are the virtues of this? First is time-reversibility. If I have $v_{3/2}$ and x_1, I can integrate backwards with the same equations to get the original $v_{1/2}$ and x_0. Believe it or, this is not true of Runge-Kutta!

Another less generally useful fact, but still useful, is that in two or more dimensions in a spherically symmetric potential, it conserves angular momentum.

Finally, it conserves phase space density. It is area preserving in phase space.