Ordinary Differential Equations - ODEs

Differential eq$^{\underline{n}}$s are super important in physics, e.g. Newton's 2^{nd} Law, Schrödinger eq$^{\underline{n}}$, heat eq$^{\underline{n}}$ etc

ODE - single independent variable (time or position)
PDE - multiple independent variables (time and position, say, or two spatial variables)

Three main types of ODE's

1) Initial value problem (IVP): time-dependent eq$^{\underline{n}}$s
need knowledge of function at specified time
e.g. $F = ma$

2) Boundary value problem: require knowledge of boundary conditions e.g.
Laplace eq$^{\underline{n}}$ in 1-D

3) Eigenvalue problem: sol$^{\underline{n}}$ exists only for selected parameter values. e.g. Schrödinger eq$^{\underline{n}}$

Systems of ODEs and IVPs

E.g. Consider the simple harmonic oscillator (SHO)

$$F = ma$$

$$-kx = m\frac{d^2x}{dt^2} \qquad m\ddot{x} = -kx$$

Can rewrite this 2^{nd} order ODE as
two first-order ODEs

position $\to$ $y_1 \equiv x$ $\qquad\qquad$ $y_2 \equiv \frac{dx}{dt} = \dot{x} = \dot{y}_1$ $\to$ velocity

$$-\frac{k}{m}x = -\frac{k}{m}y_1 = \frac{d^2x}{dt^2} = \frac{d}{dt}y_2 = \dot{y}_2$$

We get a system of equations

$$\dot{y}_1 = y_2$$
$$\dot{y}_2 = -\frac{k}{m}y_1$$

that are solved given initial ($t=0$) position $y_1(0)$
and velocity $y_2(0)$.

In general, ODE of any order n (involving $\frac{d^n y}{dt^n}$)
can be written as a set of n
first-order ODEs.

often written in vector form

$$\frac{d\vec{y}}{dt} = \vec{f}(\vec{y}, t)$$

where $\vec{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$ and $\vec{f} = \begin{pmatrix} f_1(\vec{y},t) = f_1(y_1, y_2, \dots y_n, t) \\ f_2(\vec{y},t) \\ \vdots \\ f_n(\vec{y},t) \end{pmatrix}$

For SHO example

$$y_1 = x \qquad f_1 = y_2$$
$$y_2 = \frac{dy_1}{dt} \qquad f_2 = -\frac{k}{m}\, y_1$$

(Note: independent variable does not have to be time)

Formally, we can write sol^{n} as

$$\vec{y}(t) = \vec{y}(t_0) + \int_{t_0}^{t} dt' \, \vec{f}(\vec{y}(t'), t')$$

Problem is RHS requires $\vec{y}$ for all t, but that's precisely what we need to find, and so we approximate the integral.

Euler Method

Consider discretization of indep. var.

$$\Delta t = h = t_{i+1} - t_i \qquad \text{so that}$$

$$\int_{t_i}^{t_{i+1}} dt' \, \vec{f}(\vec{y}(t'), t') \simeq \Delta t \, \vec{f}(\vec{y}(t_i), t_i)$$

- equivalent to left-side rectangle rule

then

$$\vec{y}(t_{i+1}) = \vec{y}(t_i) + \Delta t \, \vec{f}(\vec{y}(t_i), t_i)$$

or $\vec{y}_{i+1} = \vec{y}_i + h\vec{f}_i$ or just $y_{i+1} = y_i + hf_i$

Can rewrite $\dfrac{\vec{y}_{i+1} - \vec{y}_i}{h} = \vec{f}_i = \dfrac{dy_i}{dt}$ } from ODE itself

which is just the forward difference approximation

Example: SHO

$$\frac{dx}{dt} = v \qquad \frac{d}{dt}\underbrace{\begin{pmatrix} x \\ v \end{pmatrix}}_{\vec{y}} = \underbrace{\begin{pmatrix} v \\ -\frac{k}{m} x \end{pmatrix}}_{\vec{f}} \qquad f_1 = v = y_2$$

$$\frac{dv}{dt} = -\frac{k}{m} x \qquad f_2 = -\frac{k}{m} x = -\frac{k}{m} y_1$$

Euler: $x(t+\Delta t) = x(t) + \Delta t \; v(t)$

$v(t+\Delta t) = v(t) - \Delta t \; \frac{k}{m} x$

or

$$x_{i+1} = x_i + h v_i$$
$$v_{i+1} = v_i - h \frac{k}{m} x_i$$

show spring_euler.cpp results gnuplot spring.gnu

→ sol$^{\underline{n}}$ not good, as amplitude and energy grow in time

$E = \frac{1}{2} m v^2 + \frac{1}{2} k x^2$ should be constant

$$\frac{dE}{dt} = \dot{E} = m v \dot{v} + k x \dot{x} = v(\underbrace{ma + kx}_{\text{ODE} \quad ma = -kx}) = 0$$

$\dot{E} = 0 \rightarrow E = \text{constant}$ (for exact sol$^{\underline{n}}$)

Truncation error - recall forward difference

$$\frac{dy_i}{dt} = \frac{y_{i+1} - y_i}{h} + O(h)$$

So for Euler we get (note $\frac{dy_i}{dt} = f_i$)

$$y_{i+1} = y_i + hf_i + hO(h)$$
$$= y_i + hf_i + O(h^2)$$

For $N = \frac{t_f - t_0}{h}$ steps to get from initial to final time,

total error is $N O(h^2) = O(h)$

"accurate to first order"
"first-order method"

Midpoint Method or Modified Euler

Recall centred difference $\dot{y}(t) = \frac{y(t+\Delta t) - y(t-\Delta t)}{2\Delta t} + \mathcal{O}(\Delta t^2)$

or $\dot{y}(t) = \frac{y(t+\Delta t/2) - y(t-\Delta t/2)}{\Delta t} + \mathcal{O}(\Delta t^2)$

or $\dot{y}(t+\frac{\Delta t}{2}) = \frac{y(t+\Delta t) - y(t)}{\Delta t} + \mathcal{O}(\Delta t^2)$

solve for $y(t+\Delta t)$

$$y(t+\Delta t) = y(t) + \Delta t\, \dot{y}(t+\Delta t/2) + \mathcal{O}(\Delta t^3)$$

$$= y(t) + \Delta t\, f(t+\Delta t/2) + \mathcal{O}(\Delta t^3) \quad \star$$

- gain an order of accuracy if we can evaluate f at the midpoint
- Use Taylor expansion of f to estimate midpoint value

$$y(t+\Delta t) = y(t) + \Delta t \Big[\underbrace{f(t)}_{\dot{y}(t)} + \frac{\Delta t}{2} \underbrace{\frac{df}{dt}(t)}_{\ddot{y}(t)} + \mathcal{O}(\Delta t^2) \Big] + \mathcal{O}(\Delta t^3)$$

$$= y(t) + \Delta t\, \dot{y}(t) + \frac{\Delta t^2}{2} \ddot{y}(t) + \mathcal{O}(\Delta t^3)$$

→ Need only a first-order approximation to $f(t+\frac{\Delta t}{2})$

$\star\ y(t+\Delta t) = y(t) + \Delta t\, f\big(y(t+\Delta t/2), t+\Delta t/2\big) + \mathcal{O}(\Delta t^3)$

try $y(t+\Delta t/2) = y(t) + \frac{\Delta t}{2} \underbrace{f(y(t),t)}_{\frac{dy(t)}{dt}\ (\text{ODE})} + \mathcal{O}(\Delta t^2)$

This is an Euler step

call $\Delta y = \Delta t\, f(y(t), t)$

$$f\big(y(t) + \Delta y/2, t+\Delta t/2\big) = f(y(t),t) + \frac{\Delta y}{2}\frac{\partial f}{\partial y} + \frac{\Delta t}{2}\frac{\partial f}{\partial t} + \mathcal{O}(\Delta t^2)$$

$$= f(y(t),t) + \frac{\Delta t}{2}\left[\underbrace{f(y(t),t)}_{\text{ODE}}\frac{\partial f}{\partial y} + \frac{\partial f}{\partial t}\right] + \mathcal{O}(\Delta t^2)$$

$$= f(y(t),t) + \frac{\Delta t}{2}\left[\underbrace{\frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial t}}_{\frac{df}{dt}}\right] + \mathcal{O}(\Delta t^2)$$

It works! $= f + \frac{\Delta t}{2}\frac{df}{dt} + \mathcal{O}(\Delta t^2)$

Write scheme as

$$\Delta y = \Delta t\, f(y_i, t_i)$$

$$y_{i+1} = y_i + \Delta t\, f\big(y_i + \tfrac{1}{2}\Delta y, t_i + \tfrac{1}{2}\Delta t\big)$$

E.g. SHO

$$\dot{x} = v$$

$$\dot{v} = -\frac{k}{m}x$$

$\}$ $k/m = 1$ in spring-midpoint.cpp

Euler

$$x_{n+1} = x_n + \Delta t\, v_n$$

$$v_{n+1} = v_n - \Delta t\, x_n$$

Show code
- two extra lines, more stable
- still unstable at large times

Midpoint

$$x_{n+1} = x_n + \Delta t\, v_{mid}$$

$$v_{n+1} = v_n - \Delta t\, x_{mid}$$

where

$$x_{mid} = x_n + \Delta t/2\, v_n$$

$$v_{mid} = v_n - \Delta t/2\, x_n$$

Another way of writing out the midpoint scheme, closer to how it would appear in code is

$$y_{mid} = y_i + \frac{h}{2} f_i$$

$$y_{i+1} = y_i + h f(y_{mid}, t_i + \Delta t/2)$$