Lecture 17

differential equations w/ more than one variable

Suppose we would like to solve

$$\frac{dx}{dt} = xy - x \quad , \quad \frac{dy}{dt} = y - xy + \sin^2 \omega t$$

still ordinary diff. eq.'s

more general form is

$$\frac{dx}{dt} = f_x(x,y,t) \quad + \quad \frac{dy}{dt} = f_y(x,y,t)$$

For more variables, write as

$$\frac{d\underline{r}}{dt} = \underline{f}(\underline{r},t)$$

$\underline{f}$ is a vector of functions

Can Taylor expand $\underline{r}$ as

$$\underline{r}(t+h) = \underline{r}(t) + h\frac{dr}{dt} + o(h^2)$$
$$= \underline{r}(t) + h\,\underline{f}(\underline{r},t) + o(h^2)$$

Dropping order h^2 term gives the vector Euler method:

$$\underline{r}(t+h) = \underline{r}(t) + h\,\underline{f}(\underline{r},t)$$

can also generalize the Taylor expansions ~~for~~ Runge-Kutta method ~~to~~ get vector case

$$\underline{k_1} = h\,\underline{f}(\underline{r},t)$$
$$\underline{k_2} = h\,\underline{f}(\underline{r} + \tfrac{1}{2}\underline{k_1},\ t+\tfrac{1}{2}h)$$
$$\underline{k_3} = h\,\underline{f}(\underline{r} + \tfrac{1}{2}\underline{k_2},\ t+\tfrac{1}{2}h)$$
$$\underline{k_4} = h\,\underline{f}(\underline{r} + \underline{k_3},\ t+h)$$
$$\underline{r}(t+h) = \underline{r}(t) + \tfrac{1}{6}(\underline{k_1} + 2\underline{k_2} + 2\underline{k_3} + \underline{k_4})$$

Bring up 1-odesim.py

2^{nd} order & higher diff. eq's.

can use a simple trick to reduce to 1^{st} order.

general form for a 2^{nd} order diff. eq. w/ one dependent variable is

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}, t\right)$$

where f is an arbitrary function.

E.g., (#) $$\frac{d^2x}{dt^2} = \frac{1}{x}\left(\frac{dx}{dt}\right)^2 + 2\frac{dx}{dt} - x^3 e^{-4t}$$

Define $y = \frac{dx}{dt}$

& then we can write (*) as

$$\frac{dy}{dt} = f(x, y, t)$$

so these two eq's are equivalent to the 1st.

Idea was to reduce a single 2nd-order diff. eq. to 2 1st-order ones

can do the same trick for higher order

e.g. $$\frac{d^3x}{dt^3} = f\left(x, \frac{dx}{dt}, \frac{d^2x}{dt}, t\right)$$

then set $y = \frac{dx}{dt}$, $z = \frac{dy}{dt}$

& we have

$$\frac{dz}{dt} = f(x, y, z, t)$$

can generalize to vector case

Suppose we have

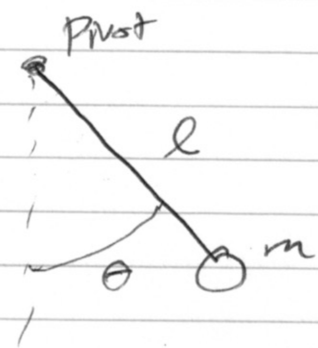
$$\frac{d^2 \underline{r}}{dt^2} = \underline{f}\left(\underline{r}, \frac{d\underline{r}}{dt}, t\right)$$

this is equivalent to

$$\frac{d\underline{r}}{dt} = \underline{s} \qquad \frac{d\underline{s}}{dt} = \underline{f}(\underline{r}, \underline{s}, t)$$

If we have n equations of mth order, then this method gives n x m simultaneous 1st order equations.

Consider a nonlinear pendulum. Standard treatment is to make this linear, but we can treat it in full generality.



acceleration of the mass is
$l\dfrac{d^2\theta}{dt^2}$ in tangential direction
+ force on mass
is vertically downward + $= mg$.
component in tangential direction
is $mg\sin\theta$.

Newton 2nd law gives equation of motion

$$ml\frac{d^2\theta}{dt^2} = -mg\sin\theta$$

$$\Rightarrow \quad l\frac{d^2\theta}{dt^2} = -g\sin\theta$$

solve numerically using

$$\frac{d\theta}{dt} = \omega$$

$$\frac{d\omega}{dt} = -\frac{g}{\ell}\sin\theta$$

Bring up 2-pendulum.py

Varying the step size when solving diff. eq's.

So far we've used repeated steps of the same size, but we can do better by varying the step size.
- worthwhile to analyze this.

Suppose function looks like

then in slowly varying regions, take larger stepsizes but in quickly varying, take smaller would like to vary the step size such that the error introduced per ~~step~~ unit interval in t is constant.

E.g., might want an error of .001 per unit time so that from $t=0$ to $t=10$ error is .01

Adaptive step size method has two parts:

1) estimate error
2) compare error to desired accuracy & increase or decrease step size accordingly

Idea: choose some initial value of h & use ordinary Runge-Kutta to do two steps of algorithm, each of size h & after 2 time steps we estimate $x(t+2h)$

Then we go back to time step t & do Runge-Kutta of size 2h. Estimate will generally be different from the previous one.

How does this help? Recall that RK is accurate to 4th order but w/ 5th order error.

So size of error is ch^5 for some c.

Starting @ time t & doing two steps, the error will be roughly $2ch^5$, i.e.,

$$x(t+2h) = \underbrace{x_1}_{\text{estimate}} + 2ch^5$$

For a single large step of size 2h, the error is

$$c(2h)^5 = c\,32h^5$$

& $x(t+2h) = x_2 + 32ch^5$

So the per-step error $\varepsilon = ch^5$ is

$$\varepsilon = ch^5 = \frac{1}{30}(x_1 - x_2)$$

we want ε to be some target accuracy. It might be better or worse, & we can adapt to make it closer to desired accuracy.

So we ask "what is step size necessary to make error size equal to target accuracy?"

Let h' denote perfect step size.

Taking steps of this size gives error ε'

$$\varepsilon' = c(h')^5 = ch^5\left(\frac{h'}{h}\right)^5$$

$$= \frac{1}{30}(x_1 - x_2)\left(\frac{h'}{h}\right)^5$$

Suppose desired overall accuracy per unit time is δ. Then desired accuracy for a single step of size h' is $h'\delta$

So solve for h' in

$$h'\delta = \frac{1}{30}|x_1 - x_2|\left(\frac{h'}{h}\right)^5$$

$$\Rightarrow \quad h' = h\left(\frac{30h\delta}{|x_1 - x_2|}\right)^{1/4}$$

$$\equiv h\,\rho^{1/4}$$

ratio of ~~target~~ accuracy to desired accuracy

Complete Method:

1) perform two RK steps of size h
   perform one RK step of size $2h$
   Gives estimates x_1 & x_2 for $x(t+2h)$
2) If $\rho > 1$ then target accuracy is ~~better~~ larger than actual accuracy. Make step bigger for next time: Set $h' = h\rho^{1/4}$

3) If $\rho < 1$ then actual accuracy larger than target accuracy. repeat calculation w/ smaller step size:

Pick $h' = h\rho^{1/4}$

- Method involves more work but often results in decreased computational time
- could happen that x_1 & x_2 are very close, causing h' to be unusually large. Fix: place an upper limit on how large h' can be.
- Important to repeat steps that miss target accuracy. Otherwise, errors build up.

can improve adaptive method slightly

Consider that

$$x(t+2h) = x_1 + 2ch^5 \; o(h^6)$$

estimated ch^5 as $\frac{1}{30}(x_1 - x_2)$

Then add this to get

$$x(t+2h) = x_1 + \frac{1}{15}(x_1 - x_2) + o(h^6)$$

accurate to order h^5 w/

error order h^6

called <u>local extrapolation</u> &

comes for free

can continue doing this, similar

to what we did w/ Romberg integration

will return to this later...

<u>other variations</u>: leap-frog method

Consider $\frac{dx}{dt} = f(x,t)$

2nd order RK - given $x(t)$, estimate value @ ~~t+h~~ h using slope @ midpoint $f(x(t+\tfrac{1}{2}h), t+\tfrac{1}{2}h)$

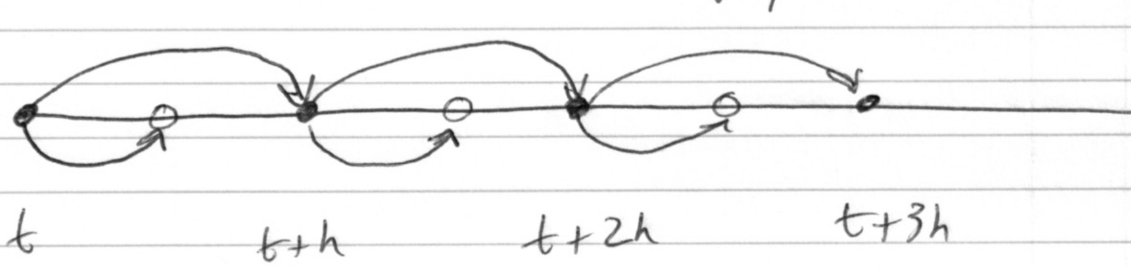
but estimate $x(t+\tfrac{1}{2}h)$ using Euler method.

can write equations as

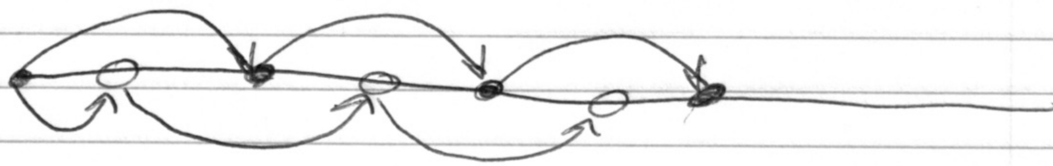
$$x(t+\tfrac{1}{2}h) = x(t) + \tfrac{1}{2}h\,f(x,t)$$

$$x(t+h) = x(t) + h\,f(x(t+\tfrac{1}{2}h), t+\tfrac{1}{2}h)$$

RK method looks like
estimate value @ midpoint & use that to get $x(t+h)$



leap frog method is the following variation



to get next midpoint use previous midpoint rather than endpoint.

Starts out the same as RK but then changes the way that midpoints are calculated.

i.e.,

$$x(t+\tfrac{3}{2}h) = x(t+\tfrac{1}{2}h) + h f(x(t+h), \cancel{t}+h)$$

then get next full step via

$$x(t+2h) = x(t+h) + h f(x(t+\tfrac{3}{2}h), t+\tfrac{3}{2}h)$$

Amounts to repeatedly applying the equations

$$x(t+h) = x(t) + h\,f(x(t+\tfrac{1}{2}h), t+\tfrac{1}{2}h)$$

$$x(t+\tfrac{3}{2}h) = x(t+\tfrac{1}{2}h) + h\,f(x(t+h), t+h)$$

each step "leaps over" previously calculated value.

each step is accurate to order h^2 & has error h^3

Advantage of this method is that it is time-reversal symmetric.