Numerical Solutions to ODEs

$$\frac{dy}{dt} = g(y) + f(t) \quad w/ \quad y(0) = y_0$$

Forward / Explicit Euler: $y_{n+1} = y_n + \Delta t\, g(y_n) + \Delta t\, f(t_n)$

Backward / Implicit Euler: $y_{n+1} = y_n + \Delta t\, g(y_{n+1}) + \Delta t\, f(t_{n+1})$

ex, 1) $\dfrac{dy}{dt} + 2y = 0 \quad w/ \quad y(0) = 1$

Analytic: $y(t) = ae^{\lambda t} \Rightarrow y' = \lambda a e^{\lambda t}$

$$\lambda a e^{\lambda t} + 2ae^{\lambda t} = 0 \Rightarrow (\lambda + 2) ae^{\lambda t} = 0$$

$$\Rightarrow \lambda = -2$$

$$\Rightarrow y(t) = ae^{-2t} \qquad y(0) = a = 1 \Rightarrow y(t) = e^{-2t}$$

Now, use Forward & Explicit Euler to get $y(1)$, exact answer is $y(1) = \underline{0.1353}$

Forward Euler w/ $\Delta t_1 = 0.25$

$$y_{n+1} = y_n - 2\Delta t\, y_n \Rightarrow (1 - 2\Delta t) y_n$$

n	t	y_n
0	0	1
1	0.25	$(1-2(0.25))(1) = 0.5$
2	0.5	0.25
3	0.75	0.125
4	1	0.0625

$\rightarrow$ Error is $0.0728 = e_1$

Now try $\Delta t_2 = 0.125$, $y(1) = 0.1001$

error $= 0.035 = e_2$

$$\frac{e_2}{e_1} = \frac{0.035}{0.0728} = 0.48 \sim \frac{\Delta t_2}{\Delta t_1} = 0.5$$

$\Rightarrow O(\Delta t)$

Now look at Backward Euler!

$$y_{n+1} = y - 2\Delta t\, y_{n+1} \Rightarrow y_{n+1} = \frac{y_n}{1+2\Delta t}$$

$\Rightarrow$ for $\Delta t_1 = 0.25$, $y(t=1) = 0.1975$, $e_1 = 0.062$
for $\Delta t_2 = 0.125$, $y(t=1) = 0.1677$, $e_2 = 0.0324$

$$\frac{e_2}{e_1} = 0.52 \sim \frac{\Delta t_2}{\Delta t_1} = 0.5$$

In general, the errors of the implicit & explicit schemes of the same order will be comparible,

So why do implicit schemes?

Stability.

Again, look at $\frac{dy}{dt} + 2y = 0$, $y(t) = e^{-2t}$

Clearly, $\lim_{t \to \infty} y(t) = 0$

Look at explicit Euler: $y_{n+1} = y_n - 2y_n \Delta t$
$= (1-2\Delta t) y_n$

Given y_0

$y_1 = (1-2\Delta t) y_0$

$y_2 = (1-2\Delta t) y_1 = (1-2\Delta t)^2 y_0$

$\vdots$

$y_n = (1-2\Delta t)^n y_0$

To have $y_n \to 0$ as $n \to \infty$, then

$|1-2\Delta t| < 1$

$\Delta t = 0.25$: $|1-2(0.25)| = 0.5 < 1$

$\Rightarrow$ 0.25 is stable.

What about $\Delta t = 1.5$?

$|1-2(1.5)| = 2 > 1$ $\leftarrow$ Not stable

$y_n \neq 0$ at $n = \infty$

Backward Euler: $y_{n+1} = \dfrac{y_n}{1+2\Delta t} = \left(\dfrac{1}{1+2\Delta t}\right) y_n$

$y_1 = (1+2\Delta t)^{-1} y_0$

$y_2 = (1+2\Delta t)^{-1} y_1 = (1+2\Delta t)^{-2} y_0$

$\vdots$

$y_n = (1+2\Delta t)^{-n} y_0$

$\Rightarrow$ Need $\left(\dfrac{1}{1+2\Delta t}\right)^n$ to be less than 1.

$\Rightarrow$ Need $|1+2\Delta t| > 1$

$\Rightarrow$ true for any $\Delta t > 0$

$\Rightarrow$ Unconditionally stable.

$\Rightarrow$ Forward Euler is Conditionally Stable,
Backward Euler is Unconditionally Stable.

Formally: let $\frac{dy}{dt} = \lambda y$ w/ $y(0) = y_0$ & $\lambda < 0$

Forward: $y_n = (1+\lambda \Delta t)^n y_0$

$A = 1 + \lambda \Delta t$ = Amplification factor

Stable if $|A| = |1 + \lambda \Delta t| < 1$

$$\Rightarrow \quad \Delta t < \frac{2}{|\lambda|}$$

Backward: $y_n = \dfrac{y_0}{(1-\lambda \Delta t)^n}$, $A = \dfrac{1}{1-\lambda \Delta t}$

$|A| < 1$ for any $\Delta t > 0$ if $\lambda < 0$

Notes: ① Not all implicit schemes are unconditionally stable. But, they are always more stable than explicit schemes.

② Just because you can use large time steps does not mean you should. Some time Δt is determined by the physics & fast dynamics.

Implicit Scheme Disadvantage : Cost

At best, a linear system needs to be solved (i.e, $y_{n+1} = y_n - 2\Delta t\, y_{n+1}$)

At worst : Solve a non-linear equation.

ex.) $\frac{dy}{dt} + \sinh(y) = 0$

$$y_{n+1} + \Delta t \sinh(y_{n+1}) = y_n$$

Despite this cost, implicit scheme can be cheaper overall for stiff problems w/ stringent Δt restrictions.

Ex.) if $\Delta t < h^4$

A modification instead of fully implicit on non-linear system :

Semi-Implicit Scheme.

Let $\frac{dy}{dt} + g(y) = 0$ $\quad g(y)$ = Any function.

If possible, split $g(y)$ in linear & non-linear part:

$$g(y) = L(y) + N(y)$$

$$\frac{dy}{dt} + L(y) + N(y) = 0$$

Then, $\frac{y_{n+1} - y_n}{\Delta t} + L(y_{n+1}) + N(y_n) = 0$

$$y_{n+1} + \Delta t\, L(y_{n+1}) = y_n - \Delta t\, N(y_n)$$

This will be less stable than fully implicit but more stable than explicit.

Usually <u>much</u> cheaper than fully implicit.

Multistage Methods.

These are schemes which take multiple "mini" steps between t_n & $t_n + \Delta t = t_{n+1}$ to get higher order schemes.

Also called Predictor - Corrector.

Focus on explicit schemes in the family called Runge-Kutta.

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

$O(\Delta t)$: Let $k_1 = f(t_n, y_n)$ ← the derivative y_t

then $y_{n+1} = y_n + \Delta t\, k_1$ ← forward Euler

$O(\Delta t^2)$ Let $k_1 = f(t_n, y_n)$

$k_2 = f(t_n + c_1 \Delta t, y_n + a_1 \Delta t\, k_1)$

for $c_1 \in [0, 1]$, $a_1 \in [0, 1]$

Given $y_n \Rightarrow k_1$ is dy/dt at t_n.

k_2 = Derivative at some time between t_n & t_{n+1}

then $y_{n+1} = y_n + b_1 \Delta t\, k_1 + b_2 \Delta t\, k_2$

$= y_n + \Delta t\,(b_1 k_1 + b_2 k_2)$

We need a_1, b_1, b_2 & c_2 to make it "$O(\Delta t^2)$"

Find y_{n+1} as Taylor Series of y_n about t_n,

$$y_{n+1} = y_n + \Delta t\, y_n' + \tfrac{1}{2}\Delta t^2\, y_n''$$

$$y_n' = f(t_n, y_n)$$

$$y_n'' = \frac{df}{dt} = f_t + f_y\, y_n'$$

(1) $\Rightarrow y_{n+1} = y_n + \Delta t\, f(t_n, y_n) + \tfrac{1}{2}\Delta t^2 f_t(y_n, t_n)$
$+ \tfrac{1}{2}\Delta t^2 f_y(t_n, y_n)\, f(y_n, t_n)$

Now expand k_2:

$k_2 = f(t_n + c_1 \Delta t,\ y_n + a_1 k_1 \Delta t) = f(t_n, y_n)$
$+ c_1 \Delta t\, f_t(t_n, y_n)$
$+ a_1 k_1 \Delta t\, f_y(t_n, y_n) + \text{H.O.T.}$

Use in $y_{n+1} = y_n + b_1 \Delta t\, k_1 + b_2 \Delta t\, k_2$

(2) $$y_{n+1} = y_n + b_1 \Delta t\, f(t_n, y_n) + b_2 \Delta t\, f(t_n, y_n)$$
$$+ b_2 c_1 \Delta t^2 f_t(t_n, y_n)$$
$$+ b_2 a_1 \Delta t^2 f(t_n, y_n)\, f_y(t_n, y_n)$$

(1) $$\Rightarrow y_{n+1} = y_n + \Delta t\, f(t_n, y_n) + \tfrac{1}{2} \Delta t^2 f_t(y_n, t_n)$$
$$+ \tfrac{1}{2} \Delta t^2 f_y(t_n, y_n)\, f(y_n, t_n)$$

Now compare (1) & (2)

$$\Rightarrow \left.\begin{aligned} b_1 + b_2 &= 1 \\ b_2 c_1 &= 1/2 \\ b_2 a_1 &= 1/2 \end{aligned}\right\} \text{4 unknowns but 3 equations} \Rightarrow \infty \text{ solutions.}$$

$\Rightarrow$ Infinite # of $O(\Delta t^2)$ RK schemes.

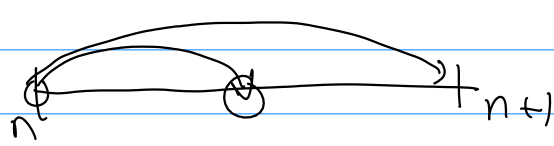
Choose one unknown (typically b_2) & solve for others.

a) Midpoint: $b_2 = 1 \Rightarrow b_1 = 0,\ c_1 = a_1 = 1/2$

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + 1/2\, \Delta t,\ y_n + 1/2\, \Delta t)$$

$$y_{n+1} = y_n + \Delta t\, k_2$$



b.) Ralston's Method

$b_2 = 3/4 \Rightarrow b_1 = 1/4 \quad , c_1 = a_1 = 2/3$

$k_1 = f(t_n, y_n)$

$k_2 = f(t_n + 2/3\,\Delta t,\ y_n + 2/3\,\Delta t\, k_1)$

$y_{n+1} = y_n + 1/4\,\Delta t\, k_1 + 3/4\,\Delta t\, k_2$

c.) Heun's Method (Improved Euler)

$b_2 = 1/2 \Rightarrow b_1 = 1/2 \quad , c_1 = a_1 = 1$

$k_1 = f(t_n, y_n)$

$k_2 = f(t_n + \Delta t,\ y_n + \Delta t\, k_1)$

$y_{n+1} = y_n + 1/2\,\Delta t\, k_1 + 1/2\,\Delta t\, k_2$

$= y_n + \Delta t\,(k_1 + k_2)/2$

$O(\Delta t^4)$: Using similar Taylor series analysis, you can get 4^{th}-order schemes.

The most well-known is simply called RK4.

$$\frac{dy}{dt} = f(t, y) \quad , \quad \text{given } y_n \text{ \& } \Delta t$$

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + 1/2\,\Delta t,\ y_n + 1/2\,\Delta t\, k_1)$$

$$k_3 = f(t_n + 1/2\,\Delta t,\ y_n + 1/2\,\Delta t\, k_2)$$

$$k_4 = f(t_n + \Delta t,\ y_n + \Delta t\, k_3)$$

$$y_{n+1} = y_n + 1/6\,(k_1 + 2k_2 + 2k_3 + k_4)$$

Generic RK Schemes

A compact way to write RK schemes is Butcher Tables / Tableau

Let a generic RK scheme of order s be:

$$y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + c_2 \Delta t, y_n + \Delta t\, a_{21} k_1)$$

$$k_3 = f(t_n + c_3 \Delta t, y_n + \Delta t\, a_{31} k_1 + \Delta t\, a_{32} k_2)$$

$$\vdots$$

$$k_i = f\left(t_n + c_i \Delta t, y_n + \Delta t \sum_{j=1}^{i-1} a_{ij} k_j\right)$$

$$\vdots$$

$$k_s = \cdots$$

Write in Table form:

$$\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\
c_2 & a_{21} & a_{22} & \cdots & a_{2s} \\
\vdots & \vdots & & & \\
c_s & a_{s1} & \cdots & \cdots & a_{ss} \\
\hline
 & b_1 & b_2 & \cdots & b_s
\end{array}$$

ex.) Forward Euler:

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

Midpoint:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \end{array}$$

Heun's Method:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array}$$

RK4:

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

Notice, In each of these the diagonal of a_{ij} is zero because they are explicit.

Matlab + RK Schemes

Matlab has many built-in ODE solvers that need:

$$f(t, y)$$
$$[t_0, t_{final}]$$
$$y(t_0)$$

Most used: ode45 $\rightarrow$ A $O(\Delta t^5)$ scheme that uses an $O(\Delta t^4)$ scheme to estimate error & vary Δt as needed.

Others: ode23, ode113, ...

Stiff problems: ode23s, ode23t, ...