Big - $\mathcal{O}$ notation

Often, want to classify functions according to growth rate

"$n^2 + n$ grows faster than $20n$ as $n \to \infty$"

or "$h^2 + h^3$ ~~grows~~ decays ~~faster~~ than h as $h \to 0$"

we say "$f(x) \underset{x \to a}{=} \mathcal{O}(g(x))$" ($f(x)$ is order $g(x)$)

if for all x sufficiently close to a, there is a constant $C > 0$ such that

$$|f(x)| \leq C\, g(x)$$

"As x tends to a, $f(x)$ grows no faster than $g(x)$"

Ex: $f(n) = 5n^2 + 2n + 1$, $n \to \infty$

Large: $|f(n)| \leq 5n^2 + 2n^2 + n^2 \leq 8n^2$

$$\Rightarrow \quad f(n) = \mathcal{O}(n^2) \quad \text{for } n \to \infty$$

"largest power wins"

Ex: $f(h) = 2h^2 + 3h + h^3 \quad , h \to 0$

small: $|f(h)| \leq 2h + 3h + h = 6h$

$\Rightarrow \quad f(h) = O(h) \quad \text{as } h \to 0.$

"smallest power wins"

Runge - Kutta methods

Basic idea: sample more points of $f(t,u)$ to get better approximations

Ex: midpoint method

$$s_1^n = f(t_n, u_n)$$

$$s_2^n = f\left(t_n + \frac{h}{2},\ u_n + \frac{h}{2} s_1^n\right)$$

$$t_{n+1} = t_n + h$$

$$u_{n+1} = u_n + s_2^n$$

take an Euler step with size $\frac{h}{2}$ $\longrightarrow$ slope s_2 at the midpoint

move along s_2 from t_0 to t_1

Ex: "classical Runge - Kutta" RK4

(MATLAB's ode 45)

$$s_1^n = f(t_n, u_n)$$

$$s_2^n = f\left(t_n + \frac{h}{2}, u_2 + \frac{h}{2} s_1^n\right)$$

$$s_3^n = f\left(t_n + \frac{h}{2}, u_n + \frac{h}{2} s_2^n\right)$$

$$s_4^n = f\left(t_n + h, u_n + h s_3^n\right)$$

$$t_{n+1} = t_n + h$$

$$u_{n+1} = u_n + \underbrace{\frac{h}{6}\left(s_1^n + 2 s_2^n + 2 s_3^n + s_4^n\right)}_{\text{weighted average}}$$

<u>rror analysis</u>: tedious, but can show

RK4 is order-4 method

neral RKn methods employ weighted averages to
ncel out $O(h^n)$ terms in the Taylor expansion.

Stability

What is a good value of h?

Forward Euler

Consider test equation

$$\frac{du}{dt} = -\lambda u \quad , \quad u(0) = 1$$

$$\rightarrow \quad u(t) = e^{-\lambda t}$$

Euler method: $\quad t_n = t_{n-1} + h = nh$

$$u_n = u_{n-1} + f(t_{n-1}, u_{n-1})$$

$$= u_{n-1} - h\lambda u_{n-1} = (1-\lambda) u_{n-1}$$

$$= (1-h\lambda)^n$$

n general: $\quad u_n = (1-h\lambda)^n u_0$

e know that our test problem has solution that decay to 0 for large t.

A numerical method is stable $\Leftrightarrow$ the approximations also decay for large n.

here: $|(1-h\lambda)^n| \to \infty \iff |1-h\lambda| > 1$

$$\iff 1 - h\lambda < -1$$

$$\iff h\lambda > 2$$

$$\iff \boxed{h > \frac{2}{\lambda}}$$

→ Forward Euler is stable for $h < \frac{2}{\lambda}$

> since FE is not always stable, we call it a conditionally stable method.

Implicit Euler

Instability often cured by backward methods

$$t_{n+1} = t_n + h$$

$$u_{n+1} = u_n + hf(t_{n+1}, u_{n+1}) \Big\} \text{ implicit eqn for } u_{n+1}$$

costly: linear system $\dot{\vec{u}} = A\vec{u}$

$$\rightarrow \quad \vec{u}_{n+1} = (\mathbb{1} - hA)^{-1}\, \vec{u}_n$$

solving a linear system is much more costly than a matrix - vector product!

Error analysis: can show: order - 1 method

Stability: $u_{n+1} = u_n - h\lambda u_{n+1}$ (test problem)

$$\Leftrightarrow \quad u_{n+1} = \frac{1}{1+h\lambda}\, u_n$$

$$= \frac{1}{(1+h\lambda)^n}\, u_0$$

$\rightsquigarrow$ always decays!

$\rightsquigarrow$ Implicit Euler is unconditionally stable.

general stability

$$\frac{d\vec{u}}{dt} = A\vec{u} \quad , \vec{u}(0) = \vec{u}_0$$

$$\rightarrow \quad \vec{u}(t) = c_1 e^{\lambda_1 t}\vec{v}_1 + \dots + c_N e^{\lambda_N t}\vec{v}_N$$

$$A\vec{v}_i = \lambda_i \vec{v}_i$$

Euler method stability : $h < \frac{2}{\lambda_{max}}$,

λ_{max} has minimal real part,

(fastest decaying mode)

but: generally want to know dynamics on the slowest timescale (to see all the dynamics)

$$\rightarrow \quad t_{max} \approx \frac{1}{\lambda_{min}}$$

$\rightarrow$ Number of time steps needed for stable integration with explicit method

$$N = \frac{t_{max}}{h} \sim \frac{\lambda_{max}}{\lambda_{min}}$$

if $\frac{\lambda_{max}}{\lambda_{min}}$ is large the ODE is called

stiff and requires special methods

(such as implicit Euler)

Nonlinear stability

Usual analysis: close to a fixed point

$$\frac{d\vec{u}}{dt} = \vec{f}(\vec{u})$$

$$f(\vec{u}_0) = \vec{0} \rightarrow \vec{u}(t) = \vec{u}_0$$

consider small perturbations

$$\vec{u}(t) = \vec{u}_0 + \Delta\vec{u}(t)$$

$$\rightarrow \frac{d\Delta\vec{u}}{dt} = \vec{f}(\vec{u}_0 + \Delta\vec{u}(t))$$

$$= \underbrace{\vec{f}(\vec{u}_0)}_{=0} + J\Delta\vec{u} + \mathcal{O}(\Delta u^2)$$

Jacobian $J_{ij} = \left.\frac{\partial f_i}{\partial u_j}\right|_{\vec{u}_0}$

$\rightarrow$ linear system, can use standard methods.