§ Ordinary differential equations

first order ODEs $\quad \dot{y} = f(t, y) \qquad t = \text{time}$

$\frac{dy}{dt}$ $\qquad$ continuous $\quad f : I \times D \to \mathbb{R}^d ; \; D \subset \mathbb{R}^d \ni y(t)$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_d \end{bmatrix} \qquad d > 1 \text{ system of ODEs}$$

Examples 1) $\quad \dot{y} = -\lambda y$ with $\lambda > 0$ constant ; $y(t) = y_0 e^{-\lambda t}, \; t \in \mathbb{R}$

2) $\quad \dot{y} = (\alpha - \beta y) y \quad$ logistic differential equation $\quad (\alpha, \beta \in \mathbb{R}$ constants $)$

$A + B \to 2B$ with rate $r = k c_A c_B$

concentrations of A, B

$$\begin{cases} \dot{c}_A = -r \\ \dot{c}_B = r \end{cases} \qquad c_A + c_B = c_A(0) + c_B(0) = D \text{ constant}$$

$\Rightarrow$ decoupled $\begin{cases} \dot{c}_A = -k(D - c_A) c_A \\ \dot{c}_B = k(D - c_B) c_B \end{cases}$

$$y(0) = y_0 > 0 \Rightarrow y(t) = \frac{\alpha y_0}{\beta y_0 + (\alpha - \beta y_0) e^{-\alpha t}}$$

$y_0 = 0 \Rightarrow y(t) = 0$

$y_0 = 1 \Rightarrow y(t) = 1$
$(\alpha, \beta = 1)$

$f(y^*) = 0 \quad$ (here at $0, \frac{\alpha}{\beta}$) $\to$ call $y^* =$ stationary points

if we start at a stationary point
$y(0) = y^*$, then $y(t) = y^*$ for all times t

$[\, \dot{y}(0) = f(y^*) = 0 \Rightarrow y$ constant ! $]$

3) Lotka-Volterra ODE

$$\begin{cases} \dot{u} = (\alpha - \beta v) u \\ \dot{v} = (\delta u - \gamma) v \end{cases} ; \quad y = \begin{bmatrix} u \\ v \end{bmatrix} \text{ and } f(y) = \begin{bmatrix} (\alpha - \beta v) u \\ (\delta u - \gamma) v \end{bmatrix}$$

$u(t)$ = no of prey (rabbits)

$v(t)$ = no of predators (fox)

Initial value problem (IVP): ODE + IV

$$\begin{cases} \dot y = f(t,y) \\ y(t_0) = y_0 \end{cases}$$

<u>Remark</u> We may consider only <u>autonomous</u> ODEs, i.e. f does not depend explicitly on the t

$$\dot y = f(y)$$

Why? Trick: define a new unknown $\boxed{t_1 = t}$

unknown $\underline{v} = \begin{bmatrix} y \\ t_1 \end{bmatrix}$ and write the equation

$\dot y = f(t,y)$ as $\dot{\underline{v}} = \begin{bmatrix} \frac{d}{dt} y \\ \frac{d}{dt} t_1 \end{bmatrix} = \begin{bmatrix} f(t,y) \\ 1 \end{bmatrix} = g(\underline{v})$

Hence: every ODE can be re-written as an autonomous ODE

$\dot{\underline{v}} = f(\underline{v})$ autonomous!

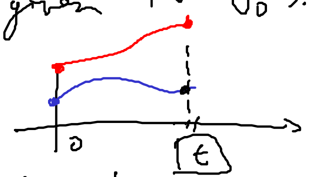
<u>Note</u> Same trick: a higher order ODE (several derivatives of y are involved)

$$\dot y, y'', y^{(3)}, \dots, y^{(k)}$$

can be re-written as a system of ODEs of first order

<u>Note</u> stick mostly on autonomous ODEs of first order

<u>Note</u> If f is smooth enough, we know that solutions of ODEs exist (and are uniques for given IV y_0).



Given $\left.\begin{matrix} \dot y = f(y) \\ y(0) = y_0 \end{matrix}\right\}$

$y_0 \longmapsto y(t)$ at a later time $t > 0$

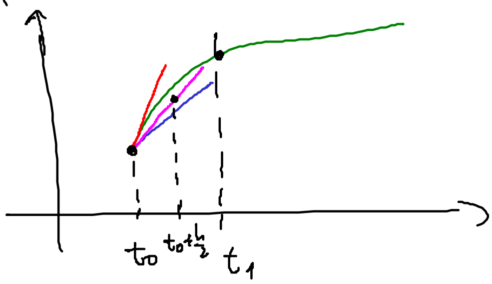
$\Phi^t y_0 := y(t)$ $\qquad \Phi^t : D \to D$

Φ = flux associated with the ODE $\dot y = f(y)$ (continuous flux)

Numerics: construct approximations / discrete versions of Φ

discrete flux Ψ

$$\boxed{\dot y = f(y)}$$



h = small , $t_1 = t_0 + h$

$y(t_1) \approx y_1$

$\dot y(t) = \frac{d}{dt} y(t) = \lim_{h \to 0} \frac{y(t+h) - y(t)}{h}$

$\frac{y_1 - y_0}{h} \approx$
- $f(y_0) \Rightarrow$ explicit Euler (eE)
- $f(y(t_0 + \frac{h}{2})) \Rightarrow$ (iMP)
- $f(y_1) \Rightarrow$ implicit Euler (iE)

(eE) $\quad y_1 = y_0 + h f(y_0)$

(iE) $\quad \underline{\underline{y_1}} = y_0 + h \boxed{f(\underline{\underline{y_1}})}$

$\boxed{\text{implicit}}$: still have to solve an algebraic eq. for y_1

$\Downarrow$

numerically (later) fsolve

(iMP) implicit midpoint rule

$$y_1 = y_0 + h f(\cancel{y(t_0 + \tfrac{h}{2})})$$

$$y_1 = y_0 + h f\left(\tfrac{1}{2}(y_0 + y_1)\right)$$

$f(t, y)$

$\downarrow$

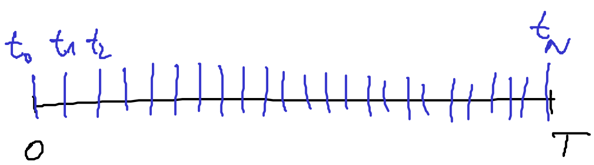
$f\left(\underline{t_0 + \frac{h}{2}}, \frac{1}{2}(y_0 + y_1)\right)$

another simplification

(iTR) $\quad y_1 = y_0 + \underline{\underline{h \frac{1}{2}\left(f(y_0) + f(y_1)\right)}}$ $\quad$ implicit trapezoidal rule

<u>Error?</u> Look at error at a given final time T

$\|y(T) - y_N\|$



N steps , $t_0 = 0$

$h = \frac{T}{N}$ $\qquad y_0, y_1 \approx y(t_1), y_2 \approx y(t_2), \dots, y_N \approx y(t_N) = y(T)$

(eE) $\quad \|y(T) - y_N\| \leq c h^1 \qquad O(h)$ error

(iE) $\quad \|y(T) - y_N\| \leq c h^1 \qquad O(h)$

(iMP) $\quad \|y(T) - y_N\| \leq c h^2 \qquad O(h^2)$

(iTR) $\quad \|y(T) - y_N\| \leq c h^2 \qquad O(h^2)$

only if solution y is smooth enough!

Runge-Kutta methods improve it to $c \cdot h^4$, $c \cdot h^5$, ...

ode45 = combination of RK of order $O(h^4)$ and $O(h^5)$

Note Convergence order is sometimes not so important...

Ex $\dot{y} = -\lambda y$, $y(t) = y_0 e^{-\lambda t} \xrightarrow[t\to\infty]{} 0$

What happens for a large T

$$(eE): \quad y_1 = y_0 + hf(y_0) = y_0 - \lambda h y_0 = (1-\lambda h) y_0 \quad \text{(one step)}$$

$$y_2 = y_1 + hf(y_1) = (1-\lambda h)^2 y_0$$

$$y_N = (1-\lambda h)^N y_0 \xrightarrow[N\to\infty]{} \begin{cases} \infty & \text{if } |1-\lambda h| > 1 \\ 1 & \text{if } |1-\lambda h| = 1 \\ 0 & \text{if } |1-\lambda h| < 1 \end{cases}$$

Stability condition: $|1-\lambda h| < 1 \iff h < \frac{2}{\lambda}$ $(\lambda > 0)$

Large $\lambda \Rightarrow$ need small h in order to have reasonable results.

$$(iE) \quad y_1 = y_0 + hf(y_1) = y_0 - \lambda h y_1 \Rightarrow y_1 + \lambda h y_1 = y_0 \Rightarrow y_1 = \frac{1}{1+\lambda h} y_0$$

$$y_2 = \left(\frac{1}{1+\lambda h}\right)^2 y_0$$

.....

$$y_N = \left(\frac{1}{1+\lambda h}\right)^N y_0 \longrightarrow 0 \quad \text{because } \lambda, h > 0 \Rightarrow 1+\lambda h > 1 \Rightarrow 0 < \frac{1}{1+\lambda h} < 1$$

Note: no restriction on the time step h

typical for implicit methods (stiff problems!)

but has its price: solving an algebraic equation at each time-step!

Note ode45 is explicit! → fast (but sometimes bad!!)

stiff equation

(iMP) ⇝ no need of such time step restriction.

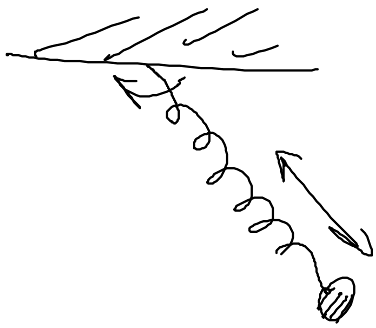
↳ "Stability"

Note More important than precision might be conservation of important quantities "good" methods

→ simplectic integrators

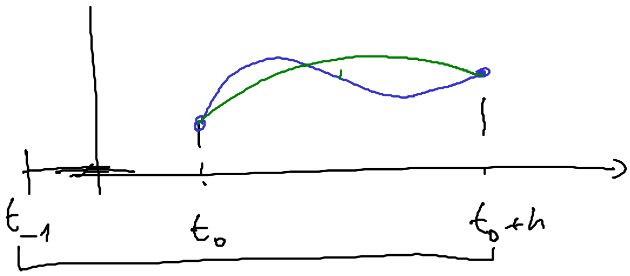
→ geometric integrators

→ (iMP)



Stoermer-Verlet method

$$\begin{cases} \ddot{y} = f(y) \\ y(t_0) = y_0 \\ \dot{y}(t_0) = v_0 \end{cases}$$



given $y_{k-1} \approx y(t_{k-1})$

$y_k \approx y(t_k)$

approximate $y(t)$ on $[t_{k-1}, t_k]$ by a parabola $p(t)$ going through (t_{k-1}, y_{k-1}) and (t_k, y_k) such that

$$\ddot{p}(t_k) = f(y_k)$$

then: $p(t_{k+1}) \approx y(t_{k+1})$

$t_k - t_{k-1} = t_{k+1} - t_k = h$ (time steps of equal length!) $\Rightarrow$

$$\boxed{y_{k+1} = -y_{k-1} + 2y_k + h^2 f(y_k)}$$

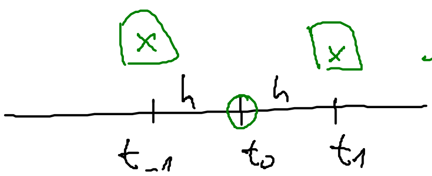
→ explicit method! $O(h^2)$

→ two-step method, needs a second starting point!

$$t_{-1} = t_0 - h, \quad y_1 = -y_{-1} + 2y_0 + h^2 f(y_0)$$

where?

$\dot{y}(t_0) = v_0 \Rightarrow$ put $\dfrac{y_1 - y_{-1}}{2h} = v_0 \Rightarrow y_{-1} = y_1 - 2h v_0$



$$\Rightarrow \boxed{y_1 = y_0 + h v_0 + \frac{h^2}{2} f(y_0)}$$

Note Another possibility:

$$f(t_k, y(t_k)) = \ddot{y}(t_k) \approx \frac{\cancel{\dot{y}(t_k) - \dot{y}(t_{k-1})}}{h} \qquad \frac{y(t_{k+1}) - y(t_k)}{h} \approx \dot{y}(t_k) \qquad \frac{y(t_k) - y(t_{k-1})}{h} \approx \dot{y}(t_{k-1})$$

with $y_k \approx y(t_k) \Rightarrow f(t_k, y_k) = \dfrac{y_{k+1} - 2y_k + y_{k-1}}{h^2} \Rightarrow \dots$ St-V

Starting point: Taylor-Approximation of order 2

$$y_1 \approx y(t_0 + h) = y(t_0) + h\,\dot{y}(t_0) + \frac{h^2}{2}\ddot{y}(t_0) + \boxed{\dots}$$

error $O(h^3)$

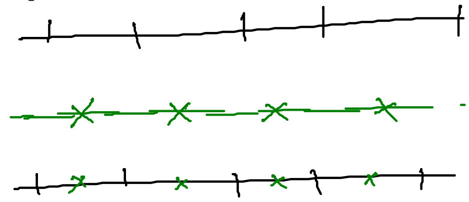
$$\Rightarrow y_1 \approx y_0 + h v_0 + \frac{h^2}{2} f(t_0, y_0)$$

Note The 2-step formulation of St V might be numerically instable (round-off errors accumulate!)

$\downarrow$ one-step reformulation

$$\ddot{y} = f(y) \quad \longleftrightarrow \quad \begin{cases} \dot{y} = v \\ \dot{v} = f(y) \end{cases}$$

$t_0 \quad t_1 \quad t_2 \quad t_3 \quad t_4$



staggered grid

$\downarrow$

$$y_{k+1} - 2y_k + y_{k-1} = h^2 f(y_k)$$

$$\begin{cases} v_{k+\frac{1}{2}} = v_k + \frac{h}{2} f(y_k) \\ y_{k+1} = y_k + h\, v_{k+\frac{1}{2}} \\ v_{k+1} = v_{k+\frac{1}{2}} + \frac{h}{2} f(y_{k+1}) \end{cases}$$

put together

$$\rightarrow \begin{cases} v_{k+\frac{1}{2}} = v_{k-\frac{1}{2}} + h f(y_k) \\ y_{k+1} = y_k + h\, v_{k+\frac{1}{2}} \end{cases} \quad \leftarrow \text{one-step formulation of St V}$$

Even better way to implement St V:

velocity-Verlet:

$$\boxed{\begin{cases} y_{k+1} = y_k + h v_k + \frac{h^2}{2} f(y_k) \\ v_{k+1} = v_k + \frac{h}{2}\left(f(y_k) + f(y_{k+1}) \right) \end{cases}}$$

stable

y_k, v_k in one step

$O(h^2)$

energy cons...

Splitting Methods

$$\begin{cases} \dot y = f(y) = \underbrace{f_a(y)} + \underbrace{f_b(y)} \\ y(0) = y_0 \end{cases}$$

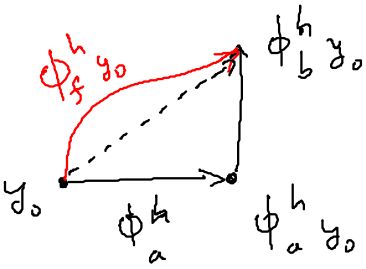
consider 2 ODEs:

$$\dot u = f_a(u) \quad \text{and} \quad \dot v = f_b(v)$$

$$\downarrow \qquad\qquad\qquad \downarrow$$

solution $u(h) = \phi_a^h\, u(0) \qquad v(h) = \phi_b^h\, v(0)$

Idea: recombine the two solutions $u(h)$ and $v(h)$ in clever way:

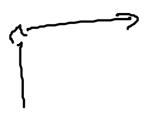


$$\Psi^h y_0 := \phi_b^h \phi_a^h y_0 \approx \phi_f^h y_0$$

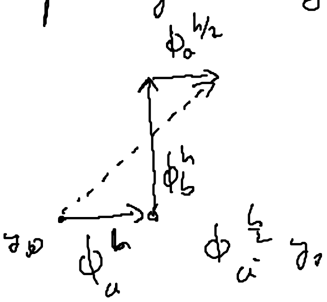
Lie-Trotter-splitting $\quad O(h)$

not energy conserving

$$\phi_a^h \phi_b^h y_0$$



better: put symmetry in method!



$$\Psi^h y_0 := \phi_a^{\frac{h}{2}} \phi_b^h \phi_a^{\frac{h}{2}} y_0$$

Strang splitting $\quad O(h^2)$

energy conserving!

Provide we know the exact solution $\phi_a^h u(0)$, $\phi_b^h v(0)$

then the scheme is simple and explicit!

depends only on r!! not on $\dot r$

Newton: $\ddot r = a(r) \iff \dot y = \begin{bmatrix} \dot r \\ \dot v \end{bmatrix} = \begin{bmatrix} v \\ a(r) \end{bmatrix} = F(y) = \begin{bmatrix} 0 \\ a(r) \end{bmatrix} + \begin{bmatrix} v \\ 0 \end{bmatrix}$

$$\begin{cases} \dot r = 0 \\ \dot v = a(r) \end{cases} \Rightarrow \begin{cases} r(t+h) = r(r) \\ v(t+h) = v(t) + h\, a(r) \end{cases} \quad \begin{cases} r_1 = r_0 \\ v_1 = v_0 + h\, r_0 \end{cases}$$

$$\begin{cases} \dot r = v \\ \dot v = 0 \end{cases} \Rightarrow \begin{cases} r_1 = r_0 + h\, v_0 \\ v_1 = v_0 \end{cases}$$

recombine: Lie-Trotter $\Rightarrow$ $\Psi^h \begin{pmatrix} r \\ v \end{pmatrix} = \begin{bmatrix} r + h(v + h\,a(r)) \\ v + h\,a(r) \end{bmatrix}$

symplectic Euler $O(h)$

Strang-Splitting $\Rightarrow$ velocity-Verlet!! $O(h^2)$, energy conserving etc.

$\longrightarrow$ due to computer problems, second hour on black-board.

$$\begin{cases} \dot{y} = f(t, y) \\ y(0) = y_0 \end{cases}$$

- $\rightarrow$ polygonal lines:  Euler, IMP, ...
- $\rightarrow$ split $f = f_a + f_b$, solve with f_a, f_b, recombine
- $\rightarrow$ $\int_0^h$, quadrature, replace unknown states with simpler formulas $\Rightarrow$ RK

$\hookrightarrow$ adaptivity

Stability and stiff ODEs

logistic $\dot y = \lambda y(1-y)$?

Model problem $\dot y = \lambda y$

Usually: y^* stationary point $f(y^*)=0$

around attractive stat. points : here $y^*=1$ attr. stat. point

$\to$ linearization around attr. stationary point

$$f(y) = \underbrace{f(y^*)}_{\substack{\shortparallel\\ 0}} + (y-y^*)\,f'(y^*) + O(|y-y^*|^2)$$

logistic eq. $f'(y) = \lambda(1-y) + \lambda(-y) = \lambda - 2\lambda y$

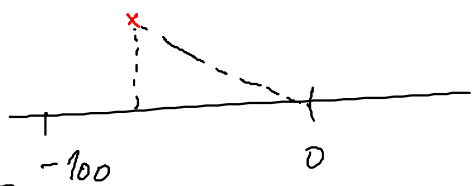
at $y^*=1 \Rightarrow f'(y^*) = -\lambda$

$\Rightarrow$ linearised eq. is : $\dot y = (y-1)(-\lambda)$

Denote $u = y-1 \Rightarrow \dot u = -\lambda u \Rightarrow \dot u = \omega u$

$\omega = -\lambda$

Hence: investigate the stability / stiffness for the model problem $\dot y = \lambda y$ in order to conclude of stability / stiffness near attr. stat. points of general non-linear problems.



e. Euler: $\dot y = \lambda y \Rightarrow h < \dfrac{2}{|\lambda|}$

$y(t) = e^{\lambda t} y(0)$ $\qquad h < \dfrac{2}{|-100|}$

iE : no time-step restriction!

Example explicite trapezoidal rule:

$$\left.\begin{array}{l} k_1 = \lambda y_0 \\ k_2 = \lambda (y_0 + k_1 h) \end{array}\right\} \Rightarrow y_1 = \underbrace{\left(1 + \lambda h + \tfrac{1}{2}(\lambda h)^2\right)}_{S(\lambda h)} y_0$$

$$y_2 = S(\lambda h)\, y_1 = S(\lambda h)^2\, y_0 \Rightarrow \dots$$

$$y_n = \boxed{S(\lambda h)}^{\,n}\, y_0 \longrightarrow 0 \text{ for } n \to \infty, \text{ when } e^{\lambda t} y_0 \to 0$$

$$|S(\lambda h)| < 1$$

must be fullfilled!

$\Rightarrow$ a bound on h!

Def $S(\lambda h) = S(z)$, $z = \lambda h$ is called stability function of the numerical method.

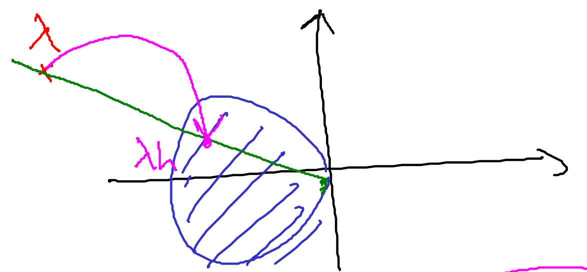
Th stability function of RK:

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

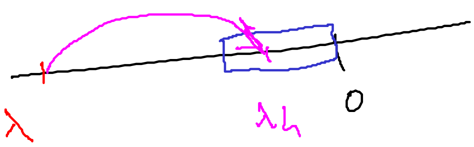
$$S(z) = \frac{\det\left(Id - zA + z\, \mathbb{1}\, b^T\right)}{\det(Id - zA)}$$

<u>Bem</u> expl. RK: $S(z)$ is a polynomial (of degree s)

and $\{z\,;\, |S(z)| < 1\}$ is bounded domain



Stability: $|S(\lambda h)| < 1$



Hence: explicit methods allways have a bound on h!!

implicit RK: $S(z) = \dfrac{P(z)}{Q(z)}$, stability domain is unbounded

Typically, implicit methods do not have a bound on h!

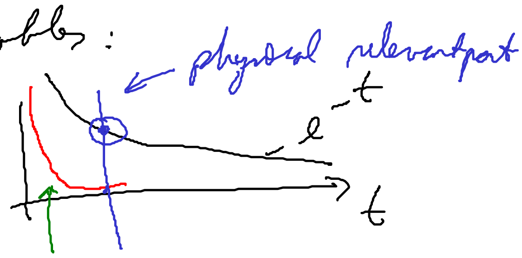
$\Longrightarrow$ they allow for large time steps h!

Def An ODE is called stiff if its stability condition needs a smaller time-step h than the accuracy of the solution requires.

Ex

$$\frac{d}{dt}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -50 & 49 \\ 49 & -50 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad y(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

diagonalization, change of variables:



$$\begin{cases} y_1(t) = e^{-t} + e^{-99t} \\ y_2(t) = e^{-t} - e^{-99t} \end{cases}$$

explicit method will be governed by e^{-99t}

e.g. (eE): $h < \frac{2}{99}$

In practical applications it happens

$\lambda_1, \lambda_2, \dots \lambda_n$

ode45 (explicit methods, time adaptive) will employ very small time steps $\lambda_n \Rightarrow$ small h

see: time integrator advances very slowly $\longrightarrow$

Then: use implicit methods

ode23s, ode15s

$\uparrow$ $\uparrow$ stiff

price to pay: solving nonlinear equations

→ expensive

→ complicated

→ no guarantees!